Examining Distributional Shifts by Using Population Stability Index (PSI) for Model Validation and Diagnosis

Alec Zhixiao Lin, LoanDepot, Foothill Ranch, CA

ABSTRACT

Population stability index (PSI) is a metric to measure how much a variable has shifted in distribution between two samples or over time. It is widely used for monitoring changes in the characteristics of a population and for diagnosing possible problems in model performance. This paper introduces an efficient process in SAS® that will automatically compare two samples by examining all model-related categorical and numeric variables with minimal manual handling needed from users. The output provides useful statistics and visuals of how much each variable has shifted in distribution.

INTRODUCTION

Population Stability Index (PSI) was originally developed in risk scorecards for monitoring the changes in distribution of a score between an out-of-time validation sample and a modeling sample. Its usage nowadays has expanded to the examination of distributional shifts for all model-related attributes, including both dependent and independent variables. When a model deteriorates in performance, checking distributional changes can help with identifying possible causes. If at least one variable has changed significantly or if several variables have moved to some extent, it might be time to recalibrate or to rebuild the model. A change in a variable distribution can be due to:

- Changes in macroeconomic climate.
- Changes in the source data, e.g., a switch to a different mailing source for marketing campaigns.
- Internal policy changes.
- Issues in data integration.
- Issues in programming, such as model implementation.

Since a distributional change does not necessarily involve a dependent variable, PSI can also be used to examine the similarity/dissimilarity between any samples, e.g, to compare education, income and health status between two or multiple populations in socia-demographic studies.

CALCULATING PSI

The following are steps for calculating PSI for a variable:

1. Divide the numeric attributes into the specified number of intervals or bins by using PROC RANK. For oridinal variables or pre-existing binning schemes, we suggest converting them into character variables so that adjacent values will not be collpased into one by PROC RANK. (This step will be skipped for character variables.)
2. Calculate % of records in each interval based on modeling sample.
3. Obtain the cutoff points for all intervals.
4. Apply the cutoff points to a different sample, usually a validation sample.
5. Generating the distributions for the validation sample.
6. Sum up the statistics across all intervals for both modeling and validation samples to calculate PSI.

Let's illustrate the calculation of PSI with an example. Suppose a revenue score is distributed in a modeling sample (% Expected) and a validation sample (% Actual) as follows:

Revenue Bin	Revenue Score	# Expected	#Actual	% Expected	% Actual	Index/PSI
1	5	1,232	7,372	0.1057	0.2790	0.1682
2	19	1,187	3,589	0.1018	0.1358	0.0098
3	29	1,085	3,095	0.0931	0.1171	0.0055
4	38	1,249	2,967	0.1071	0.1123	0.0002
5	46	1,080	2,461	0.0926	0.0931	0.0000
6	55	1,186	2,347	0.1017	0.0888	0.0018
7	65	1,228	2,069	0.1053	0.0783	0.0080
8	74	1,156	1,282	0.0992	0.0485	0.0362
9	85	1,096	710	0.0940	0.0269	0.0841
10	94	1,159	534	0.0994	0.0202	0.1262
Total		11,658	26,426	1.0000	1.0000	0.4401

Tables 1. Example of PSI Calculation

Each band is labeled by the its median value. The first score band has a median value of 5, with the index computed as follows:

$$Index = (27.90\% - 10.57\%) \times ln\frac{27.90\%}{10.57\%} = 0.1682$$

We can sum up the index across all intervals to get the PSI for this variable:

$$PSI = \sum\left((\%Actual - \%Expected) \times \ln\frac{\%Actual}{\%Expected}\right)$$

If a variable shows exactly the same distribution in a validation sample as in the modeling sample, its PSI is equal to 0. The following is the rule of thumb for determining to what extent a variable has shifted in distribution:

< 0.1: Very slight change.
0.1- 0.2: some minor change.
> 0.2: Significant change.

In our example, PSI for the revenue score is 0.4401, which suggests a significant change in distribution.

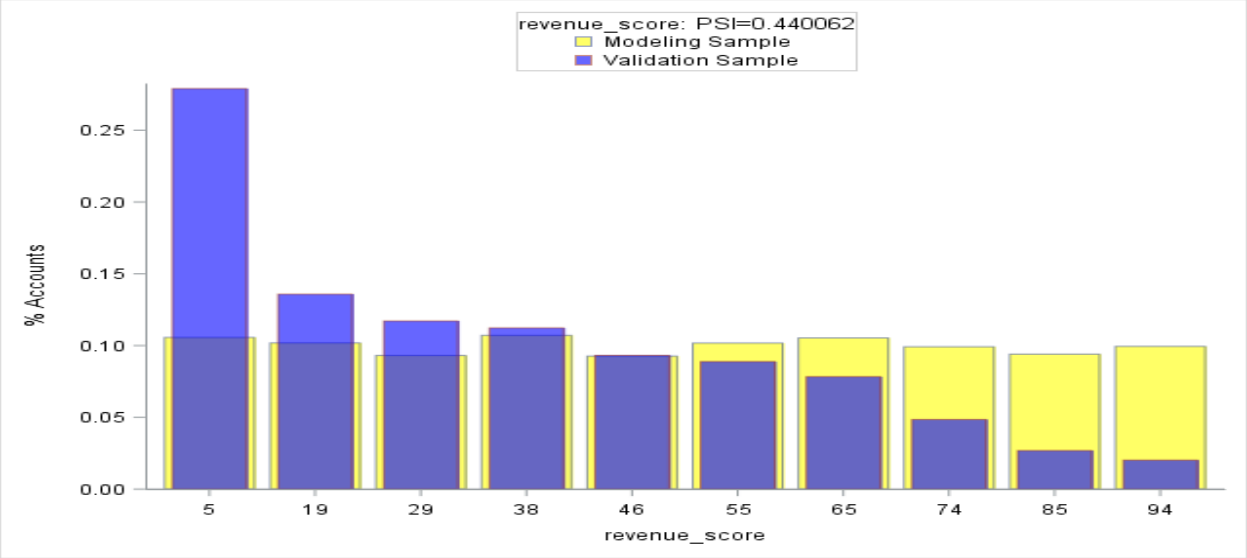

Figure 1. Example of Distributional Change and PSI

THE SAS PROGRAM

This paper introduces a SAS process that will automatically compare the distributions of all numeric and character variables. Users only need to define the data set and variables to be examined in part I to run the program.

```
%let inputset= modeling_sample;                              /* The modeling sample */
%let compareset=validation_sample;                          /* The validation sample */
%let varnum=num1 num2 num3 num4 num5 num6;                /* list of numeric variables */
%let binnum=10;                                  /* number of bins for numeric variables */
%let vartxt=char1 char2 char3 char4;                      /* list of character variables */
%let imtxt=____;                                  /* label for missing character values */
%let missnum=-999999999;                          /* label for missing character values */
%let yaxislabel=% Accounts;                                /* label for  distribution */
%let labelmod=Modeling Sample;                            /* label for modeling sample */
%let labeloot=Validation Sample;                         /* label for validation sample */
```

(See the complete code in the appendix)

The process has taken the following into consideration:

- No need to impute missing values. Missing values are treated as a separate interval in an attribute.

- If no numeric variables or no character variables need to be analyzed, the macro values for `varnum` or `vartxt` can be left blank. The program will automatically skip either set of variables in data processing.

The SAS output will present the table and an associated chart for each variable examined, as illustrated by Table 1 and Figure 1 above.

CONCLUSION

This paper introduces an efficient process in SAS that compares the distributions of all numeric and categorical variables used in a model. It is a useful tool to be incorporated into the practice of model validation. Since the presence of a dependent variable is not required, it can also be used to examine the similarity or dissimilarity between any two samples for key attributes.

CONTACT INFORMATION

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

Alec Zhixiao Lin
VP Modeling
Loan Depot
25500 Towne Center Drive
Foothill Ranch, CA 92630
Email: alecindc@gmail.com
Web: www.linkedin.com/pub/alec-zhixiao-lin/25/708/261/

APPENDIX

```
                    *** Part I: Define Data Set and Variables ****;

%let inputset= modeling_sample;                                  /* The modeling sample */
%let compareset=validation_sample;                             /* The validation sample */
%let varnum=num1 num2 num3 num4 num5 num6;                /* list of numeric variables */
%let binnum=10;                                  /* number of bins for numeric variables */
%let vartxt=char1 char2 char3 char4;                      /* list of character variables */
%let imtxt=____;                                /* label for missing character values */
%let missnum=-999999999;                        /* label for missing character values */
%let yaxislabel=% Accounts;                                /* label for  distribution */
%let labelmod=Modeling Sample;                            /* label for modeling sample */
%let labeloot=Validation Sample;                         /* label for validation sample */

                    ********* Part II: Numeric Variables *********;
%macro dealnum;
%if %sysfunc(countw(&varnum dummyfill)) > 0 %then %do;
data check_contents;
retain &varnum;
set &inputset(keep=&varnum obs=1); run;

proc contents data=check_contents varnum out=check_contents2 noprint; run;
proc sort data=check_contents2(keep=name varnum)
out=checkfreq(rename=(name=tablevar)); by varnum; run;
data varcnt; set checkfreq; varcnt+1; 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 rank data=&inputset group=&binnum out=check_rank ties=low;
var &varnum;
```

```
ranks rank1-rank&vmcnt; run;

data check_rank;
set check_rank;
array fillmiss(*) rank1-rank&vmcnt;
do j=1 to dim(fillmiss);
if fillmiss(j)=. then fillmiss(j)=-1;
fillmiss(j)=fillmiss(j)+1;
end;
drop j;  run;

%macro meannum;
%do i=1 %to &vmcnt;
proc means data=check_rank nway min max median noprint;
class rank&i;
var &&v&i;
output out=check&i(drop=_type_ rename=(_freq_=freq_&i))
       min=min_v&i
          max=max_v&i
       median=&&v&i;run;

data check&i; set check&i; rank_num_&i+1; run;
proc sql noprint; select max(rank_num_&i) into :maxrank from check&i; quit;

data check&i;
length sas_code $ 256.;
set check&i;
if rank_num_&i=1 then sas_code="if &&v&i le "||max_v&i||" then rank_num_&i=1;";
else sas_code="else if &&v&i le "||max_v&i||" then rank_num_&i="||rank_num_&i||";";

if rank_num_&i=&maxrank then sas_code="else rank_num_&i="||rank_num_&i||";";
sas_code=compbl(sas_code); run;

proc sort data=check&i; by rank_num_&i; run;
proc sql noprint; select sas_code into :algnum&i separated by ' ' from check&i ; quit;

data check_mod_sample; set check_rank; &&algnum&i; run;
data check_oot_sample; set &compareset; &&algnum&i; run;

proc freq data=check_mod_sample noprint; tables
rank_num_&i/out=modeling_freq(rename=(count=count_mod percent=freq_mod)); run;
proc freq data=check_oot_sample noprint; tables
rank_num_&i/out=oot_freq(rename=(count=count_oot percent=freq_oot)); run;

proc sort data=modeling_freq; by rank_num_&i; run;
proc sort data=oot_freq; by rank_num_&i; run;
proc sort data=check&i; by rank_num_&i; run;

proc sql noprint; select count(*) into :totcntoot from check_oot_sample; quit;
proc sql noprint; select sum(count_mod) into :totcntmod from modeling_freq; quit;
proc sql noprint; select sum(count_oot) into :totcntoot from oot_freq; quit;
proc sql noprint; select sum(freq_mod) into :totfreqmod from modeling_freq; quit;
proc sql noprint; select sum(freq_oot) into :totfreqoot from oot_freq; quit;

data modeling_oot_freq;
merge modeling_freq oot_freq check&i(keep=rank_num_&i &&v&i sas_code);
by rank_num_&i;

if count_oot=. then count_oot=0;
if freq_oot=. then freq_oot=1/&totcntoot;

freq_mod=freq_mod/100;
freq_oot=freq_oot/100;
```

```
if freq_mod > freq_oot then PSI=(freq_oot-freq_mod)*log(freq_oot/freq_mod);
else PSI=(freq_mod-freq_oot)*log(freq_mod/freq_oot);
order_rank=put(rank_num_&i, 5.); run;

proc sql noprint; select sum(PSI) into :psi from modeling_oot_freq; quit;

data for_total;
order_rank="Total";
PSI=&psi;
count_mod=&totcntmod;
count_oot=&totcntoot;
freq_mod=&totfreqmod/100;
freq_oot=&totfreqoot/100; run;

data modeling_oot_&i;
retain order_rank &&v&i count_mod count_oot freq_mod freq_oot PSI;
set modeling_oot_freq for_total;
drop rank_num_&i;

format freq_mod 6.4;
format freq_oot 6.4;
informat freq_mod 6.4;
format freq_oot 6.4;run;

proc print data=modeling_oot_&i(drop=sas_code) noobs; title "&&v&i"; run;

data modeling_oot_for_graph;
set modeling_oot_&i;
if compress(order_rank)='1' and &&v&i=. then &&v&i=&missnum; run;

proc sgplot data=modeling_oot_for_graph subpixel noborder;
vbar &&v&i / response=freq_mod transparency=0.4 FILLATTRS=(color=yellow);
vbar &&v&i / response=freq_oot transparency=0.4  FILLATTRS=(color=blue) barwidth=0.7;
keylegend / down=2 location=outside position=top /* noborder */ title="&&v&i:
PSI=&psi";
label freq_mod="&labelmod";
label freq_oot="&labeloot";
yaxis label="&yaxislabel"; run;
%end;
%mend meannum;
%meannum;
%end;
%mend dealnum;
%dealnum;

                    ******* Part III: Character Variables ********;

%macro dealtxt;
%if %sysfunc(countw(&vartxt dummyfill)) > 0 %then %do;
data check_contents;
retain &vartxt;
set &inputset(keep=&vartxt obs=1); run;

proc contents data=check_contents varnum out=check_contents2 noprint; run;
proc sort data=check_contents2(keep=name varnum)
out=checkfreq(rename=(name=tablevar)); by varnum; run;

data varcnt; set checkfreq; varcnt+1; 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('vxcnt', vcnt); run;
proc sql noprint; select tablevar into :v1-:v&vxcnt from varcnt; quit;
```

```
data check_rank;
length &vartxt $20.;
set &inputset;

array fillmiss(*) &vartxt;
do j=1 to dim(fillmiss);
if missing(fillmiss(j)) then fillmiss(j)="&imtxt"; end; run;

%macro freqmodel;
%do i=1 %to &vxcnt;
proc freq data=check_rank noprint; tables
&&v&i/out=modeling_freq(rename=(count=count_mod percent=freq_mod &&v&i=var_label));
run;

data check&i; set modeling_freq; rank_txt_&i+1; run;
proc sql noprint; select max(rank_txt_&i) into :maxrank from check&i; quit;

data check&i;
length sas_code $ 256.;
set check&i;
if rank_txt_&i=1 then sas_code="if &&v&i="||strip("'"||var_label||"'")||" then
rank_txt_&i=1;";
else sas_code="else if &&v&i="||strip("'"||var_label||"'")||" then
rank_txt_&i="||rank_txt_&i||";"; run;

data fillallothers;
rank_txt_&i=&maxrank+1;
var_label="z.allothers";
sas_code="else do; rank_txt_&i="||rank_txt_&i||"; var_label='z.allothers'; end;";
run;

data check&i;
set check&i fillallothers;
sas_code=compbl(sas_code); run;

proc sort data=check&i; by rank_txt_&i; run;
proc sql noprint; select sas_code into :algtxt&i separated by ' ' from check&i ; quit;

data check_mod_sample; set check_rank; &&algtxt&i; run;
data check_oot_sample; set &compareset; if &&v&i=' ' then &&v&i="&imtxt"; &&algtxt&i;
run;

proc freq data=check_mod_sample noprint; tables
rank_txt_&i/out=modeling_freq(rename=(count=count_mod percent=freq_mod)); run;
proc freq data=check_oot_sample noprint; tables
rank_txt_&i/out=oot_freq(rename=(count=count_oot percent=freq_oot)); run;

proc sort data=modeling_freq; by rank_txt_&i; run;
proc sort data=oot_freq; by rank_txt_&i; run;
proc sort data=check&i; by rank_txt_&i; run;

proc sql noprint; select count(*) into :totcntoot from check_oot_sample; quit;

data modeling_oot_freq;
merge modeling_freq oot_freq check&i(keep=rank_txt_&i var_label sas_code);
by rank_txt_&i;

if count_oot=. then count_oot=0;
if freq_oot=. then freq_oot=1/&totcntoot;

if count_mod in (0, .) and count_oot > 0 then do;
count_mod=0;
```

```
freq_mod=1/&totcntoot; end;

freq_mod=freq_mod/100;
freq_oot=freq_oot/100;

if freq_mod > freq_oot then PSI=(freq_oot-freq_mod)*log(freq_oot/freq_mod);
else PSI=(freq_mod-freq_oot)*log(freq_mod/freq_oot);
order_rank=put(rank_txt_&i, 5.); run;

proc sql noprint; select sum(PSI) into :psi from modeling_oot_freq; quit;
proc sql noprint; select sum(count_mod) into :totcntmod from modeling_oot_freq; quit;
proc sql noprint; select sum(count_oot) into :totcntoot from modeling_oot_freq; quit;
proc sql noprint; select sum(freq_mod) into :totfreqmod from modeling_oot_freq; quit;
proc sql noprint; select sum(freq_oot) into :totfreqoot from modeling_oot_freq; quit;

data for_total;
order_rank="Total";
PSI=&psi;
count_mod=&totcntmod;
count_oot=&totcntoot;
freq_mod=&totfreqmod/100;
freq_oot=&totfreqoot/100; run;

data modeling_oot_char&i;
retain order_rank var_label count_mod count_oot freq_mod freq_oot PSI;
set modeling_oot_freq for_total;
drop rank_txt_&i; run;

data modeling_oot_char;
set modeling_oot_char&i;
if order_rank ne 'Total' and count_mod in (0, .) and count_oot in (0, .) then delete;

format freq_mod 6.4;
format freq_oot 6.4;
informat freq_mod 6.4;
format freq_oot 6.4; run;

proc print data=modeling_oot_char(drop=sas_code) noobs; title "&&v&i"; run;

proc sgplot data=modeling_oot_char subpixel noborder;
vbar var_label / response=freq_mod transparency=0.4 FILLATTRS=(color=yellow);
vbar var_label / response=freq_oot transparency=0.4  FILLATTRS=(color=blue)
barwidth=0.7;
keylegend / down=2 location=outside position=top /* noborder */ title="&&v&i:
PSI=&psi";
label freq_mod="&labelmod";
label freq_oot="&labeloot";
yaxis label="&yaxislabel"; run;
%end;
%mend freqmodel;
%freqmodel;
%end;
%mend dealtxt;
%dealtxt;
ods pdf close;
```