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 possible changes in the characteristics of a population and for diagnosing any problems in model performance. If one or more variables have changed significantly, it usually suggests the need for refreshing or rebuilding a model. 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 by users. The output provides useful statistics and visuals of how much each variable has shifted in distribution.

INTRODUCTION

Population Stability Index (PSI) was developed in risk scorecards for monitoring the changes in distribution of a score between an out-of-time validation sample and a modeling sample. The initial idea was to check "How the current scoring is compared to the predicted probability from training data set". Nowadays its usage 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 is a useful practice to identify the cause. If at least one variable has changed significantly or if several variables have changed at least moderately, the model should be refreshed or rebuilt. A change in a variable can be due to one of the following causes:

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

CALCULATING PSI

The following are steps for calculating PSI for a numeric variable:

1) Split the data into 10 or 20 groups. One can apply PROC RANK to divide the sample into 10 or 20 groups.
2) Calculate % of records in each group based on scoring sample.
3) Obtain the high cap and low cap of each group to be used as variable cuts.
4) Apply the cuts to a different sample, usually a validation sample.
5) Calculate the distributions by group for the modeling sample and the validation sample.
6) Sum up the statistics across all groups from 5) to get PSI.

The following formula is used for Steps 5) and 6) to calculate PSI:

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

where %Expected and %Actual refer to the distribution of a variable in a validation sample and a modeling sample respectively.

For character or ordinal variables[1], we directly apply the distribution across the multiple values (including missing values) from the modeling sample in step 1. Users can also apply existing binning method (such as those commonly seen in defining credit grade) in step 1. The rest of the steps are the same.

[1] Even though an ordinal variables is often expressed as numeric numbers with a monotonic trend, we suggest transforming it into character variables so that multiple adjacent values will be collapsed into one by the SAS process.

Let's use an example for illustration. Suppose a risk score is distributed in a modeling sample (% Expected) and a validation sample (% Actual) as follows:

Score bands	Risk Score	% Expected	% Actual	% Expected	% Actual	Index/PSI
1	42	3,718	154	10.6	6.0	0.0255
2	69	3,795	172	10.8	6.8	0.0191
3	90	3,239	141	9.2	5.5	0.0188
4	111	3,537	195	10.1	7.7	0.0066
5	138	3,320	189	9.5	7.4	0.0049
6	168	3,596	301	10.2	11.8	0.0023
7	201	3,457	298	9.8	11.7	0.0032
8	249	3,515	369	10.0	14.5	0.0165
9	321	3,444	412	9.8	16.2	0.0318
10	444	3,503	317	10.0	12.4	0.0055
Total		35,124	2,548	100.0	100.0	0.1342

Tables 1 - Example of PSI Calculation

Each group is labeled by its median value. As an example, the index for the 2nd score band is computed as following:

$$Index = (10.8\% - 6.8\%) \times ln\frac{10.8\%}{6.8\%} = 0.0191$$

If a variable has exactly the same distribution in a validation sample as in a modeling sample, its PSI is equal to 0. The following is the rule of thumb to determine 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 risk score is 0.1342, which suggest some minor changes in distribution, as exhibited by the following chart:

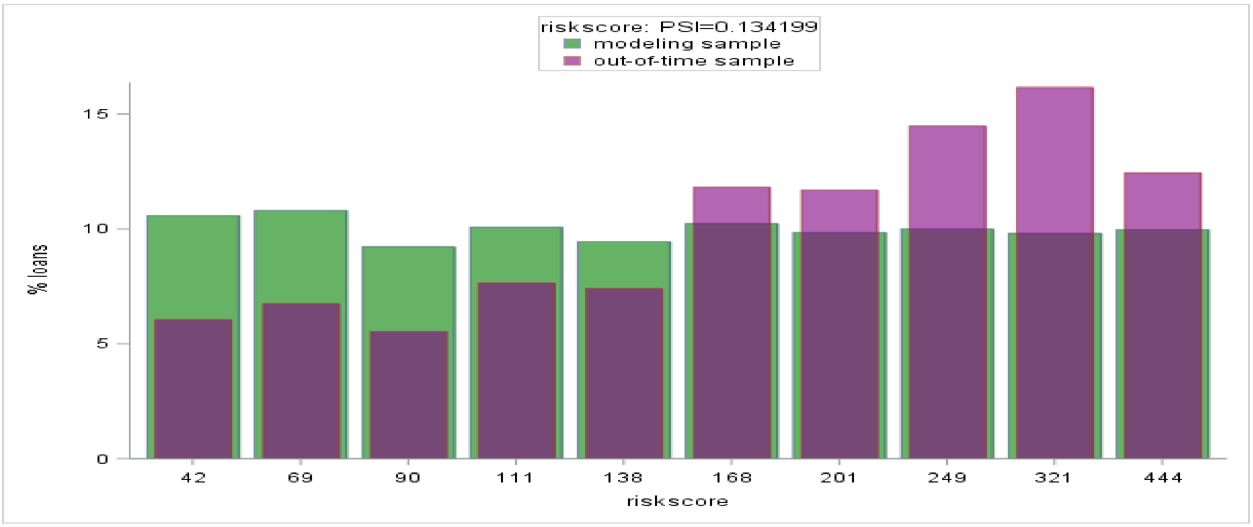

Figure 1 – Example of Distributional Change and PSI

In the chart, the risk score of the modeling sample is distributed quitely evenly, almost 10% for each group. The out-of-time validation sample shows a visibly high percentage of records clustering on the higher bands. This could be due to some internal policy change, such as more strigent measures imposed for risk reduction.

THE SAS PROGRAM

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

```
%let inputset=lib.modeling_sample;                                 /* The modeling sample */
%let compareset=lib.oot_sample;                                  /* The validation sample */
%let varnum=
num1 num2 num3 num4 num5 num6 num7 num8;                     /* 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=_MISSING_;                                /* label for missing character values */
```

(See the complete SAS program in the Appendix.)

There is no need to change the contents in Part II and Part III.

The program will automatically handles the following:

- It will group missing values for a numeric or a character variable into a separate group for calculating PSI. No need to impute the data for missing value.
- If you only have numeric variables to analyze, you can keep the macro values for character variables `vartxt` blank. The program will only analyze numeric variables. Vice versa if you only have character variables to analyze.
- In the case of multiple validation sample to examine, one can plug them into the macro variable `compareset` to do comparison to the modeling sample one by one.

The SAS output will present the table and an associated chart for each variable examined, as illustrated in the previous section.

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.

CONTACT INFORMATION

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

Alec Zhixiao Lin
VP Modeling
Loan Depot
26642 Towne Center Drive
Foothill Ranch, CA 92610
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=lib.modeling_sample;                                 /* The modeling sample */
%let compareset=lib.oot_sample;                                  /* The validation sample */
```

```
%let varnum=
num1 num2 num3 num4 num5 num6 num7 num8;              /* 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=_MISSING_;                          /* label for missing character values */


**********************************************
********* 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 i=1 to dim(fillmiss);
if fillmiss(i)=. then fillmiss(i)=-1;
fillmiss(i)=fillmiss(i)+1;
end;
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;

if freq_mod > freq_oot then PSI=(freq_oot-freq_mod)/100*log(freq_oot/freq_mod);
else PSI=(freq_mod-freq_oot)/100*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;
freq_oot=&totfreqoot;
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;
run;

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

proc sgplot data=modeling_oot_&i subpixel noborder;
vbar &&v&i / response=freq_mod transparency=0.4 FILLATTRS=(color=green);
vbar &&v&i / response=freq_oot transparency=0.4  FILLATTRS=(color=purple)
barwidth=0.7;
```

```
keylegend / down=2 location=outside position=top /* noborder */ title="&&v&i:
PSI=&psi";
label freq_mod="modeling sample";
label freq_oot="out-of-time sample";
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;
set &inputset;

array fillmiss(*) $ &vartxt;
do i=1 to dim(fillmiss);
if missing(fillmiss(i)) then fillmiss(i)="&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;

if freq_mod > freq_oot then PSI=(freq_oot-freq_mod)/100*log(freq_oot/freq_mod);
else PSI=(freq_mod-freq_oot)/100*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;
freq_oot=&totfreqoot;
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;
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=green);
vbar var_label / response=freq_oot transparency=0.4  FILLATTRS=(color=purple)
barwidth=0.7;
keylegend / down=2 location=outside position=top /* noborder */ title="&&v&i:
PSI=&psi";
label freq_mod="modeling sample";
label freq_oot="out-of-time sample";
yaxis label="&yaxislabel";
run;

%end;
%mend freqmodel;
%freqmodel;
%end;
%mend dealtxt;
%dealtxt;
```