PharmaSUG China 2014 - Paper CC02

Macro free automation to identify and drop variables with all values as missing

Vamse Goutam, Genpact, India

Amarnath Vijayarangan, Genpact, India

ABSTRACT

Missing values do have a vital role to play in handling data either numeric or character. Especially when a variable is completely missing it taxes the execution speed and also the space that the dataset occupies. However, if we are able to identify these missing variables even before the analysis or study and drop them it would significantly increase the execution speed and would also save a good impactful amount of space in the server. There are several solutions to this problem, but this paper introduces a macro free automation to identify and then drop the completely missing variables, as any use of macros or even a simple proc freq would run into a risk of either a complete failure or slowdown of the process due to insufficient macro length or numerous levels in the dataset.

INTRODUCTION

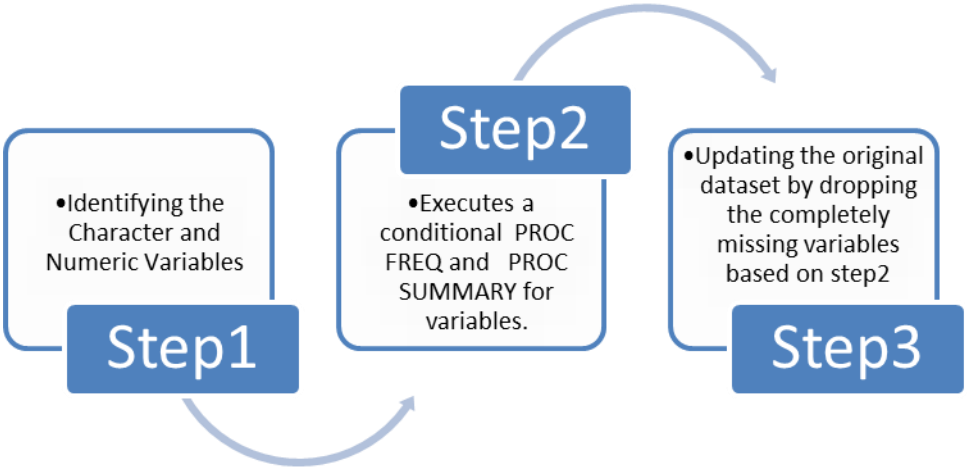
Various SAS® procedures can be used to identify the variables with completely missing values and then an implementation of a data step or proc step will drop these missing variables. But this is a onetime solution for smaller datasets. In case of a repetitive task or larger datasets it is always efficient to execute it through an automated process. Here we propose an automated solution and to make it even more efficient and simple we do not use any macros in it thus making it completely a macro free automated process to identify and drop the variables with completely missing values.

APPROACH

The proposed approach in this paper uses proc format along with file statement and data steps to create various files and treating data steps as an intermediary controlling tool. The process involves execution of proc freq with proc format (to overcome the risk of complete failure or slowdown of the process) for character variables and proc summary for numeric variables. We integrate the list of completely missing variables from these two procedures and perform dataset modifications through conditional execution.

To make the process easily understandable we have used universally available data from SAS® help library and the vcolumn data which is accessible by all the SAS® users.

PROGRAM FLOW



STEP1: IDENTIFYING THE CHARACTER AND NUMERIC VARIABLES

Step1 mainly focusses on identifying the type of the variables in each dataset to be analyzed. Two separate variables (cexist and nexist) are created to help in identifying the presence of the numeric and character variables in each datasets to be analyzed. These variables will have a value greater than 0 if their respective type variables exist in the dataset. For instance cexist will have a value greater than zero, thus helping us to understand that the dataset does contain character variables. These two variables are the input for Step2. This step is also the point where the user needs to list the library and the datasets that needs to be processed.

In addition to the above, step1 also creates a custom format in order to improve the execution time. The custom format created using PROC FORMAT presents all the non – missing values as 1 and the missing values as missing itself. Please refer section 1.1 in the appendix for the code.

STEP2: EXECUTES A CONDITIONAL PROC FREQ AND PROC SUMMARY

Takes the variables created in step1 (cexist and nexist) as input and checks for their value. If cexist > 0 then PROC FREQ is executed for all the character variables with the custom format created in step1. If nexist > 0 then, PROC SUMMARY IS executed for all the numeric variables in the data. The output of the PROCs is converted into a SAS® datasets (CharVarList and NumVarList) only with the variables with completely missing values. This dataset further acts as an input in the final step to update the original datasets. Please refer section 1.2 in the appendix for the code.

STEP3: UPDATING THE ORIGINAL DATASETS

The dataset created in step2 from the PROC FREQ AND PROC SUMMARY are used to extract the exhaustive list of the variables with completely missing values which is used as an input in the drop options of the final data step used to drop the missing variables. Please refer section 1.3 in the appendix for the code.

CONCLUSION

The proposed approach uses the power of data step without using any macros and advanced technique. The approach works faster and smothers on large dataset with many variables either numeric or character with several levels. A process controlled through conditional execution using data step and file statement without any advanced coding makes it easier to use and debug hence efficient among the existing approach.

CONTACT INFORMATION

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

Name: Vamse Goutam
Enterprise: Genpact
Address: Block B Salarpuria Soft Zone Outer Ring Road, Bellandur
City, State ZIP: Bangalore, Karnataka, 560103
E-mail: vamse.kobe09@gmail.com

Name: Amarnath Vijayarangan
Enterprise: Genpact
Address: Block B Salarpuria Soft Zone Outer Ring Road, Bellandur
City, State ZIP: Bangalore, Karnataka, 560103
E-mail: amarnath7@gmail.com

APPENDIX

Macro free automation to identify and drop variables with all values as missing

```
data c1 c2(drop=name sex);
    set sashelp.class;
run;
/*************************      Section 1.1     ****************************/
proc format ;
    value $ chform
        ' ' = ''
      Other = '1';
run;
```

Creates formats for character variables

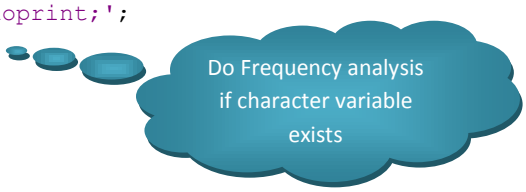
```
data _null_;
    set sashelp.vcolumn(keep=libname  memname type where= (libname = upcase
                ("WORK") and memname in ("C1" "C2"))) end=last;
    by libname memname ;
    if first.memname then do;
         cexist=0;
        nexist=0;
    end;
    file 'RunDropVar';
       if type='char' then cexist+1;
       if type='num' then nexist+1;
       if last.memname then do;
              put 'data CharVarList CharVarList1(drop=dropvarlist)
              NumVarList(keep=DropVarList);';
              put '   length DropVarList $ 32;';
              put "   DropVarList='';";
              put '   nlevels=.;';
              put '   nmisslevels=.;';
              put '   delete;';
              put 'run;';
              put ' ';
```

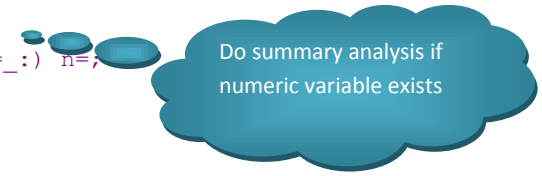
Library and dataset name to be used

Initialize char & Num variable flags for each dataset

Creates temporary dataset for each dataset process

```
/*************************      Section 1.2     ***************************/
              if cexist>0 then do;
                    put 'ods output NLevels=CharVarList;';
                    put 'ods listing close;';
                    put 'proc freq data=' libname '.' memname 'nlevels;';
                    put '    format _character_ $chform.  ;';
                    put '    tables _character_ /noprint;';
                    put 'run;';
                    put 'ods listing;';
                    put 'ods output close;';
                    put ' ';
                    put 'data CharVarList ;';
                    put '    set CharVarList CharVarList1;';
                    put '    if nlevels=1 and nmisslevels=1;';
                    put '    rename tablevar=DropVarList;';
                    put 'run;';
                    put ' ';
              end;
              if nexist>0 then do;
                    put 'proc summary data=' libname '.' memname ';';
                    put '    var _numeric_;';
                    put '    output out=NumVarList(drop=_:) n=;';
                    put 'run;';
                    put ' ';
                    put 'proc transpose data=NumVarList';
                    put '          out=NumVarList(rename=(_name_=DropVarList)';
                    put '          where=(col1=0));';
                    put 'run;';
                    put ' ';
              end;
/*************************      Section 1.3     ***************************/
              put 'data _null_;';
```





```
            put '  set CharVarList(keep=DropVarList)';
            put '       NumVarList(keep=DropVarList) end=last;';
            put "   file 'DropVar';";
            put '   if _n_=1 then do;';
            put "        put 'data " libname '.' memname ";';";
            put "       put     'set " libname '.' memname "(drop=';";
            put '   end;';
            put "   put DropVarList @@;";
            put "   if last then put " "');run;';";
            put 'run;';
            put ' ';
            put "%include 'DropVar';";
            put ' ';
      end;
run;
%include "RunDropVar";
```