A SAS® Application to Identify and Evaluate Outliers
Richard Hendra, MDRC, New York, NY
Paulette W. Staum, Paul Waldron Consulting, West Nyack, NY

ABSTRACT
This paper presents an approach to outlier identification and evaluation that utilizes multiple SAS procedures packaged into a unified application. The output includes reports and plots, with information about extreme values, influence statistics, and the effect of outliers on a model of relationships among variables. It produces a compact, readable report for each variable of interest. The focus is on analyzing outliers within experimental research designs.

INTRODUCTION
This paper describes an application that allows analysts to efficiently identify and evaluate outliers. The application produces one page of output for each variable to be analyzed. Each page contains multiple perspectives on any outliers for a single variable. The application includes techniques for 4 tasks:

1. Identify outliers
2. Combine different types of information for each variable
3. Evaluate outliers
4. Repeat the process for many variables

An outlier is a value or an observation that is quite different from most of the other values or observations in a data set. Sometimes the existence of an outlier is obvious from examining the values for one variable only (i.e., from a univariate perspective). At other times, an outlier is only obvious when examining the values for a combination of variables in an observation (i.e., from a multivariate perspective).

An outlier influences both the mean and the variance of a variable’s distribution. For small samples, this can easily distort the results of an analysis. For large samples, the effect on the mean is smaller, but the increased variance can change the statistical significance of regression estimates. A group of borderline outliers (“fringeliers”) can also have a similar effect on analysis results. Also, the effect of an outlier on conclusions for a sub-group can be larger than the effect for the full sample.

Outliers appear in data for different reasons, including:

- valid natural variation
- measurement errors
- incorrect selection of a sample (i.e., millionaires in a welfare study)
- data entry errors (i.e., monthly or annual wages entered as hourly wages)

What should you do about outliers? Depending on how the data was collected, you might be able to determine whether an outlier is correct or incorrect. If it is incorrect, you might be able to replace it with the correct value, or you might need to delete the faulty value. If it is correct or its correctness cannot be determined, then you might include the value, but you should evaluate how your analysis results would be different without it.

Data from a random assignment study will be used as an example throughout. In the study, each participant was assigned to either the Program or Control group. Data about each individual in the study was collected and stored in one observation per person. The data includes many variables, but we will only use these variables:

- SampleID – identifier for each individual
- E – indicator of whether the individual was in the experimental (program) group or the control group
- YrEarn – earnings for the year before the beginning of the study
- Earn13 and Earn14 - earnings for the 13th and 14th quarters during the study

We will focus on the values for Earn14, the earnings in the fourteenth quarter. See the next page for a sample application output, using data for this variable. This report combines tables and graphs from many different SAS procedures and yields a multi-faceted, integrated perspective on outliers. It takes some effort to produce the report, but it will facilitate improved and more efficient decision making.
1. IDENTIFY OUTLIERS
What is the best way to identify outliers? There is no single ‘best’ method. Both graphical and tabular outputs are useful, as are univariate and multivariate outputs. The distribution of values is a univariate concept that will be presented both graphically and in tables. The list of extreme values is a univariate perspective that will be presented in tables. The distribution across subgroups and the relationship with values of other continuous variables are multivariate perspectives that will be presented graphically. The regression influence statistics will be presented in a table. There are other alternatives that may meet your needs better, but this is an example of what we find useful.

All of these contribute to deciding whether or not a value is an outlier. The output in Figure 1 includes these items:

- Box plots and histograms
- Univariate distribution tables
- Extreme values
- Distribution plots for subgroups (‘inliers’)
- Multivariate scatter plots
- Multivariate influence statistics

1A. UNIVARIATE BOX PLOTS AND HISTOGRAMS
It’s always good to start with the ‘big picture’. Several SAS procedures can produce box plots and histograms. We’ll use the ODS Statistical Graphics procedure, PROC SGPLOT. (Both SAS/GRAPH and SAS/STAT licenses are required to make ODS Statistical Graphics available. At least SAS version 9.2 is required.)

Here are a vertical box plot and a histogram with a density plot. Notice that many individuals have earnings equal to zero because this study included many unemployed people. Most people have earnings under $20,000, but one person has earnings over $40,000 for the quarter. There is a very long right-hand tail in the histogram.

Creating Box Plot
```
proc sgplot data= test.ftp_ar;
  vbox earn14;
run;
```

Creating Histogram
```
proc sgplot data= test.ftp_ar;
  histogram earn14;
  density earn14;
run;
```
1B. UNIVARIATE STATISTICS

Next, let’s get numeric descriptions of the distribution. Running PROC UNIVARIATE yields a variety of descriptive information, including moments, basic statistical measures of location and variability, quantiles and extreme values. (PROC UNIVARIATE can also produce plots, but we chose to use PROC SGPLOT for this purpose.)

The basic output from the procedure includes both the univariate distribution and a list of extreme values. It also includes information that we don’t want. The ODS OUTPUT statement sends the results from any procedure to a series of different data sets each structured to hold the different types of results produced by the procedure. The ODS EXCLUDE statement can eliminate the procedure’s printed output, either selectively by naming each type of output to eliminate, or completely (by using the ODS EXCLUDE ALL statement).

After using the ODS OUTPUT and ODS EXCLUDE statements to redirect the relevant procedure output to data sets, it takes less than a page of code to restructure the basic measures and quantiles for a more compact report. (See the Appendix for the complete code.)

Notice that at least 50% of the values for the EARN14 variable are zero. Also, the maximum is more than 6 times the value at the 99th percentile.

```
 Producing Univariate Distribution Statistics
 ods output basicmeasures = work.UV_basicmeasures
     quantiles = work.UV_quantiles
     extremeobs = work.UV_extremeobs;
 ods exclude moments testsforlocation basicmeasures quantiles extremeobs;
 proc univariate data= test.ftp_ar nextrobs=5;
    id Earn14;
    var Earn14;
 run;
 ods exclude none;
 ods output close;
 * Continue with re-structuring of ODS output data sets, as shown in Appendix;
```
1C. EXTREME VALUES

Now that we have both a graphical and tabular look at the distribution of the data, it’s useful to examine the values at the extremes of the distribution. The PROC UNIVARIATE code shown above produced a list of extreme values in addition to information about the distribution. With a little care in printing the ODS output data set for extreme values, you can produce the tables below. Looking at extreme values in the context of other extreme values can help you form an intuition about whether they are a part of a normal distribution.

Notice that all the low values are zero since there are no negative earnings. Also, the distance between the high value and the next highest value is much larger than the distance between other high values. Finally, the ID column in the table below was assigned consecutively, so for this data the ID and the observation number are the same. In other cases, the ID might be a name or another meaningful identifier.

<table>
<thead>
<tr>
<th>Low Value ID</th>
<th>Obs #</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2813</td>
</tr>
<tr>
<td>0</td>
<td>2810</td>
</tr>
<tr>
<td>0</td>
<td>2809</td>
</tr>
<tr>
<td>0</td>
<td>2807</td>
</tr>
<tr>
<td>0</td>
<td>2805</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>High Value ID</th>
<th>Obs #</th>
</tr>
</thead>
<tbody>
<tr>
<td>45100</td>
<td>2815</td>
</tr>
<tr>
<td>17100</td>
<td>2724</td>
</tr>
<tr>
<td>13900</td>
<td>329</td>
</tr>
<tr>
<td>11700</td>
<td>1222</td>
</tr>
<tr>
<td>10900</td>
<td>1045</td>
</tr>
</tbody>
</table>

1D. SUBGROUPS

In addition to the 3 univariate perspectives described above, multivariate perspectives should be considered. First look at the relationship of outliers with a categorical variable.

If your data contains data for many distinct subgroups, it may be important to look for outliers within subgroups. These are sometimes called “inliers” because they might not be extreme values in the context of the entire sample.

You can create a panel of graphs for a variable, with each graph showing the variable values categorized by age group, gender, socio-economic status or any other categorical variable.

Notice that the one extreme outlier is in the program group. This raises the troubling possibility that the presence of the outlier might lead to an exaggeration of the effect of the program on earnings in the 14th quarter. This question will be discussed in Section 2.

Creating BoxPlot Panel by Subgroup

```sas
proc sgpanel data=ftp_ar;
    panelby e;
    vbox Earn14;
run;
```
1E. MULTIVARIATE SCATTER PLOTS

Now consider the relationship of outliers with another continuous variable. What if 2 continuous variables are closely related? If you expect a relationship between them (e.g., height and weight), then a corresponding outlier for both variables would tend to make you believe that an outlier was a correct, true value.

Graphing relationships among variables can reveal exceptions to general rules. In our example, our interpretation of the outlier value for earnings in quarter 14 will be different if we discover that earnings in quarter 13 were similar. This scatter plot gives an overview of the relationships between the earnings in quarters 13 through 14.

Notice that the quarter 14 outlier does not have a corresponding outlier in quarter 13.

1F. MULTIVARIATE INFLUENCE STATISTICS

Finally, it can be useful to look at outliers in the context of all the information available in an observation. PROC REG or PROC GLM can calculate influence statistics for each observation. These influence statistics (for example, Cook’s Distance) can be used to provide context for evaluating extreme values. If an extreme value for a variable occurs in an observation which is NOT influential, there is far less reason to be concerned about it.

Here we will arbitrarily choose to use Cook’s D instead of any other influence statistic. The conventional cut-off point for Cook’s D is $4 / \text{NObs}$, where NObs is the number of observations in the data set. Cook’s D can be produced by either PROC REG or PROC GLM. The code below assumes that the NObs for observations with non-missing values for relevant variables was already calculated.

R. Dennis Cook described the test known as “Cook’s Distance” in the article “Detection of influential observations in linear regression” published in 1977.

Producing Influence Statistics

* Calculate Cooks D and create data set with LSMeans;
  ods output lsmeans=lsmeans0;
  proc glm data=ftp_ar(keep=sampleid e earn12-earn16) ;
    class e;
    model earn14 = e yearn;
    lsmeans e / pdiff stderr;
    output out=work.cooksd(where=(cooksd>= 4/&nobs)) cookd=cooksd;
  quit;
  ods output close;
2. COMBINE OUTPUT

If we run each of the SAS procedures above for many variables at once, the analyst will need to search through the output, looking for the different bits and pieces that describe the same variable. This is guaranteed to waste time and increase frustration. Instead, the combined results of all procedures can be displayed in a report which is organized by variable.

SAS provides several ways to rearrange output. The ODS Document destination and PROC DOCUMENT are one approach. The document approach is appealing if you have only one document to produce, but its methods for identifying the parts of a document seem cumbersome for any automated processing of multiple documents. SAS macro language is another approach and it will be used in our application to loop through our process for many variables. We will use ODS LAYOUT to place the output from multiple procedures on one page.

The paper by O’Connor and Huntley has a good explanation of the syntax for the ODS LAYOUT and ODS REGION statements. Note that ODS LAYOUT is preproduction in SAS version 9.2. The syntax might change in the future. We had the best results from using ODS LAYOUT with PDF output. We had difficulties with other forms of output.

Our layout uses 2 columns, with a left-hand column that is narrower than the right-hand column. The order in which the regions are populated is top left, top right, second row left, second row right, etc. The position of all regions is determined automatically in our example (but could be explicitly controlled). The size of graphs and regions for graphs is sometimes explicitly controlled. This is not required, but in our case it increased the reliability of the placement of the regions. We also found it necessary sometimes to insert an empty region spanning all columns to force a region to appear in the left-hand column reliably.

The code below places two graphs (from section 1A above) side by side at the top of the page. Then it displays restructured output from PROC UNIVARIATE (from section 1B). The Appendix extends this approach to place additional outputs on the page.

The ODS GRAPHICS statement is used with the HEIGHT and WEIGHT options to control the size of the graphs produced by the SGPLOT and SGPANEL procedures. This allows the production of graphs which are a suitable size for placement on the page.

The ODS PDF statement names the output file (FILE=), suppresses the PDF table of contents (NOTOC), prevents SAS from starting a new page for the output of each procedure (STARTPAGE=NEVER) and sets the style to STATISTICAL.

The ODS LAYOUT START statement begins the process of placing output on the page. It specifies the height and width of the output, and requests a layout with 2 columns.

A series of ODS REGION statements specify areas on the page that will contain output from SAS procedures. You have the choice of using an absolute or gridded (relative) layout, with either explicit or automatic positions for regions. We used a relative layout, so there are no x= or y= options on the ODS REGION statements. We specified sizes for the regions using the WIDTH and HEIGHT options, since this helps produce a consistent placement of regions. Each of the tables and graphs produced in the greyed-out code is placed in a separate region. Also, the COLUMN_SPAN=1 or COLUMN_SPAN=2 option allows you to produce regions which span 1 or 2 columns.

Be sure to check your log for warnings. If a region is not large enough for a graph or table, there will be a warning that information has been truncated.

The ODS PDF TEXT statement can be used to insert text in a region to label or explain the region. You can use inline formatting to control the appearance of the text.

The ODS LAYOUT END statement finishes the layout. The ODS PDF CLOSE statement closes the PDF file.

Note that this application design requires each procedure to run once for each variable. This is not efficient in terms of computing resources, but it is very efficient in terms of human resources.
3. EVALUATE OUTLIERS

Now we have identified potential outliers by looking at them from a wide variety of perspectives. What do you do next? When you see outliers, ask questions.

- First try to confirm the accuracy of the data points. Sometimes multiple sources of information are available. Are the different sources consistent or inconsistent? If they are consistent, or you can otherwise confirm that an outlier is correct, obviously it should remain in the analysis.

- Next, if possible, ask the original source for an outlying data value to find a corrected value. Perhaps there was a measurement or data entry error. Perhaps a survey question was misinterpreted and the answers need to be discarded. There might be mistakes in the identifiers for members of a sample, or dates might be wrong. Maybe there were programming errors that incorrectly derived variables or merged data. If the problem with a data point can be fixed, that is the best solution.

- Sometimes a data point is simply wrong. There might be inherent limits to the range of possible values (minimum or maximum values for ages, dates or payments), so you know a value can't be correct, but you don't know what it should be. Any known erroneous values should be deleted.

What should you do when an outlier cannot be verified or rejected? The fact that the values are outliers does not imply that the values are wrong. Utilize the varied perspectives assembled above to provide context.

- Does the distribution indicate that it is reasonable in the context of other values for the entire sample?
- Is it reasonable in the context of other values for the sub-group?
- Is it reasonable in the context of the values of other variables in the observation?
- Is the outlier part of an observation which is not influential?

A powerful tool for evaluating the effect of outliers is an analysis of the sensitivity of the model results to the outliers. If including or excluding potential outliers has no consequences for the model, then you know the outlier...
is not tainting the analysis. If there is an effect from the outliers, then it is important to communicate that sensitivity to your audience.

One way to check this is to run the model first on the full sample, and then on the sample without the most extreme outlier, then on the sample without the two most extreme outliers, etc. If the results are NOT significantly different when you have removed outliers, any conclusions based on the model are much more robust.

The table below is the result of repeated runs of PROC GLM, each time with an additional "outlier" value removed. It is produced by a series of steps similar to the one above that created the Cooks D statistics. The only difference is that each step contains a WHERE statement that removes additional observations. The series of ODS output data sets are then combined for printing.

Notice that the outlier, even though large, did not greatly affect the results and had minimal consequences for the significance of the result.

<table>
<thead>
<tr>
<th>Value Deleted</th>
<th># Obs with This Value</th>
<th>LSMean_Control</th>
<th>LSMean_Program</th>
<th>LSMeanDiff</th>
<th>Pr &gt;</th>
<th>t</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1,100.51</td>
<td>1,334.90</td>
<td>234.39</td>
<td>0.001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>45100</td>
<td>1</td>
<td>1,100.53</td>
<td>1,303.70</td>
<td>203.17</td>
<td>0.002</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17100</td>
<td>1</td>
<td>1,100.60</td>
<td>1,292.38</td>
<td>191.78</td>
<td>0.003</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13900</td>
<td>1</td>
<td>1,091.41</td>
<td>1,292.48</td>
<td>201.07</td>
<td>0.002</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11700</td>
<td>1</td>
<td>1,091.45</td>
<td>1,285.02</td>
<td>193.57</td>
<td>0.003</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10900</td>
<td>2</td>
<td>1,084.69</td>
<td>1,277.95</td>
<td>193.26</td>
<td>0.002</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

You might expect that erroneous outliers would falsely increase the significance of a result by increasing or decreasing the mean. In fact, simulations indicate that the increase in variance is usually more important and results are often less significant when outliers are included in the analysis.

This type of output can provide analysts with a sense of the range of uncertainty about the results. If the outliers are affecting the results of the analysis in a meaningful way, your options include:

- Report the analysis both with and without the outliers if their inclusion changes the results (perhaps in an appendix).
- Use robust methods of analysis like PROC ROBUSTREG
- Create robust measures (using variables representing distribution categories or ranges, percentages or logs) which are less sensitive to the outliers.

Used by permission of Ben Shabad
http://www.davidmlane.com/ben/cartoons.html
4. PROCESS MANY VARIABLES

Finally, it would be good to be able to produce reports of outliers for many variables in a variety of data sets. Let’s combine our outlier identification, evaluation and reporting code in a SAS macro. The macro will need parameters for the name of the input data set, the variables of interest, the proposed model to be analyzed and the report output destination. It could have additional parameters to provide control over specific processing, e.g., what rules to use for selecting outliers to be omitted in the evaluation process, what type of influence statistic to use, etc. A call to this type of macro is shown here. The Appendix has an example of what the macro might look like. The macro is NOT truly generic, because the ODS LAYOUT technique may require adjusting depending on the input characteristics and the output you want to include.

Calling a Macro for Multiple Variables

```sas
%outlierchk(data = ftp_ar,
            var = earn14 earn15,
            idvar = sampleid,
            classvar = e,
            covar = yrearn,
            scattervar = earn13,
            pdf = c:\nesc2010\outlierid\ftp_earn.pdf);
```

CONCLUSION

SAS provides many procedures that you can use to examine data for outliers, including PROC SGPLOT and SGPANEL, PROC UNIVARIATE and PROC GLM. Consolidating the results of these procedures into a one-page report for each variable allows a data analyst to easily review and integrate many perspectives on outliers.

REFERENCES

Please refer to the online SAS documentation for descriptions of SAS Procedures and other syntax.


ACKNOWLEDGMENTS
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.

Thanks to Aaron Hill and Olga Yurkov for sharing their knowledge about handling outliers.

The data used in this paper are derived from data files made available to researchers by MDRC. The authors remain solely responsible for how the data have been used or interpreted.

Cartoon used by permission of Ben Shabad.

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

Richard Hendra Paulette Staum
MDRC Paul Waldron Consulting
16 East 34th Street 2 Tupper Lane
New York, NY 10016 West Nyack, NY 10994
Email: RichardHendra@mdrc.org Email: staump@optonline.net

APPENDIX: SAMPLE MACRO
%macro outlierchk(data=, var=, idvar=, classvar=, covar=, scattervar=, nextreme=5, style=statistical, pdf= );
/*----------------------------------------------------------
Prepare reports for outlier checks for variables

Parameters:
data    name of input data set - required
var     list of variables to check - required
idvar   name of variable which contains identifier for observations - optional
classvar name of class variable for regressions and plots by category - optional
covar   name of covariate variable(s) for regression - optional
scattervar name of covariate variable for scatter plots - optional
nextreme number of extreme values to display and to exclude from regression - default is 5
style   style for output - default is statistical
pdf     path and name of output PDF file - required

Sample Call: %outlierchk(data=sashelp.class, var=height weight,
idvar=name, classvar=sex, covar=age, scattervar=age,
pdf=c:\nesug2010\outlierid\class.pdf);
Usage Notes:
Model is &ThisVar = &ClassVar &CoVar
Limited to class variables with 2 values
WARNING: This macro will need adapting for your circumstances and data.*/
%local connect cooksnobs iExtreme iVar jExtreme
   NClassVals NVars OutlierValue ThisVar;

%* Check parameters (variable existence tests omitted to save space)
---------------------------------------------------------------------
%* Add more parameter checking if desired;
%if %length(&data)=0 or %length(&var)=0 or %length(&pdf)=0 %then %do;
   %put ERROR: The macro OutlierChk requires values for the data, var and pdf parameters.;
   %goto quit;
%end;
%if %sysfunc(exist(&data))=0 and %sysfunc(exist(&data,view))=0 %then %do;
   %put ERROR: The macro OutlierChk could not find input data set &data.;
   %goto quit;
%end;
%* Get the number of observations in the input data set;
%let dsid nobs rc;
%let dsid = %sysfunc(open(&data, IS));
%let nobs = %sysfunc(attrn(&dsid, NLOBS));
%let rc = %sysfunc(close(&dsid));

%* Check number of values for the class variable;
%if %length(&classvar) %then %do;
   proc sql noprint;
   select count(distinct &classvar) into :NClassVals from &data;
   quit;
%if &NClassVals ^= 2 %then %do;
   %put WARNING: The macro OutlierChk can not do a sensitivity analysis for a class variable with
   more than 2 values;
%end;
%end;
%*---------------------------------------------------------------------

%---------------------------------------------------------------------
Set up PDF
---------------------------------------------------------------------
options nodate nonumber nocenter orientation=portrait;
ods graphics / reset=all height=1.5in width=3in;
ods listing close;
ods pdf file="&pdf" startpage=no notoc style=&style;
ods layout start width=8in height=10in columns=2;
%let NVars=%sysfunc(countw(&var));

%* Loop to process each variable
---------------------------------------------------------------------
%do iVar = 1 %to &NVars;
   %let ThisVar=%scan(&var,&iVar);
   title "%upcase(&ThisVar)   Outlier Evaluation in Data Set %upcase(&data)   (NObs=&NObs)
   &sysdate";
   ods pdf startpage=now;
   ods region column_span=1 width=3in height=1.5in;
   proc sgplot data=&data;
      vbox &ThisVar;
   run;
   ods region column_span=1 width=4.5in height=1.5in;
   proc sgplot data=&data;
      histogram &ThisVar;
      density &ThisVar;
   run;
%end;
%*---------------------------------------------------------------------
Proc Univariate
---------------------------------------------------------------------
ods output basicmeasures = work.UV_basicmeasures
   quantiles  = work.UV_quantiles
   extremeobs = work.UV_extremeobs;
ods exclude moments testsforlocation basicmeasures quantiles extremeobs;
proc univariate data=&data nextrobs=&nextreme;
   id &IDVar;
   var &ThisVar;
run;
ods exclude none;
ods output close;
%*
   ---------------------------------------------------------------------
   Restructure the basic measures output
   The basic measures data set has two measures per observation.
   One locmeasure (location) and one varmeasure (variance).
   ---------------------------------------------------------------------;
data work.UV_basicmeasures;
   set work.UV_basicmeasures;
   by varname;
   retain mean median stddev iqtrrange;
   keep varname mean median stddev iqtrrange;
   if locmeasure='Mean' then mean=LocValue;
   else if locmeasure='Median' then median = LocValue;
   if varmeasure='Std Deviation' then stddev=VarValue;
   else if varmeasure = 'Interquartile Range' then IQtrRange = VarValue;
run;
%*
   ---------------------------------------------------------------------
   Transpose the quantiles
   ---------------------------------------------------------------------;
proc transpose data=work.UV_quantiles out=work.UV_quantiles;
   by varname;
   id quantile;
   var estimate;
run;
%*
   ---------------------------------------------------------------------
   Report basic measures, quantiles and extreme values
   ---------------------------------------------------------------------;
ods rregion column_span=2;
proc print data=work.UV_BasicMeasures noobs label heading=h;
   label varname='Variable' mean='Mean' median='Median' stddev='Std Dev' iqtrrange='Inter Quartile Range';
run;
ods region column_span=2;
proc print data=work.UV_Quantiles(drop=_name_) noobs label heading=h;
   var _0__Min  _1_ _5_ _10__ _25__Q1 _50__Median _75__Q3 _90__ _95__ _99_ _100__Max;
   label varname='Variable'         _100__Max = 'Max'
                 _99_      = '99th %tile'  _95_ = '95th %tile'
                 _90_      = '90th %tile'  _75__Q3 = '75th %tile'
                 _50__Median = '50th %tile' _25__Q1 = '25th %tile'
                 _10_      = '10th %tile'  _5_ = '5th %tile'
                 _1_       = '1st %tile'   _0__Min = 'Min';
run;
ods region column_span=2;
proc print data=work.UV_ExtremeObs(keep=low &idvar._low lowobs) noobs label heading=h;
   label low='Low Value' lowobs='Obs #';
run;
proc sort data=work.UV_ExtremeObs(keep=high &idvar._high highobs) out=work.high;
   by descending high;
run;
ods region column_span=1;
proc print data=work.high noobs label heading=h;

label high='High Value' highobs='Obs #';
run;

%if %length(&classvar) %then %do;
  *---------------------------------------------------------------
  Panel of categorical plots
  *-----------------------------------------------------------------
  %* See SAS Usage Note 39196 - height is required to prevent page break;
  ods region column_span=1 height=1.5in;
  proc sgpanel data=&data;
    panelby &classvar;
    vbox &ThisVar;
  run;
  %end;

%if %length(&scattervar) %then %do;
  *---------------------------------------------------------------
  Scatter plot
  *-----------------------------------------------------------------
  ods region column_span=1 height=1.5in;
  proc sgpanel data=&data;
    %if %length(&classvar) %then %do;
      panelby &classvar;
    %end;
    scatter y=&ThisVar x=&ScatterVar;
  run;
  %end;

%if &nobs < &NExtreme + 10 %then %do;
  %put WARNING: Input data set &data has fewer than %eval(&NExtreme + 10) observations;
  %put WARNING: Regression analysis will be omitted;
  %goto quit;
%end;

%*---------------------------------------------------------------
  Run Proc GLM for all data
  Note assumption that variable to be analyzed is the dependent variable
  Produce Cooks D (or other influence statistics)
  *-----------------------------------------------------------------
  ods exclude all;
  ods output lsmeans=lsmeas0(drop=effect dependent stderr probt);
  proc glm data=&data(keep=&IDVar &ClassVar &ThisVar &CoVar) ;
    %if %length(&ClassVar) %then %do;
      class &ClassVar;
    %end;
    model &ThisVar = &ClassVar &CoVar;
    %if %length(&ClassVar) %then %do;
      lsmeans &ClassVar / pdiff stderr;
    %end;
    %* Limit output to large values of Cooks D - NObs should be (but is not)
    the exact number of observations with no missing values for relevant variables;
    output out=work.cooksd(where=(cooksd>= 4/&nobs)) cookd=cooksd;
  quit;
  ods output close;
  ods select all;
%*---------------------------------------------------------------
  Report Cooks D
  *-----------------------------------------------------------------
  proc sort data=cooksd;
    by descending cooksd;
  run;

%let dsid = %sysfunc(open(cooksd, IS));
%let cooksnobs = %sysfunc(attrn(&dsid, NLOBS));
%let rc = %sysfunc(close(&dsid));
%if &cooksnobs > 0 %then %do;
ods region column_span=1;
proc print data=cooksds(obs=&NExtreme) label;
    id &IDVar;
    var cooks &ThisVar &classvar &covar;
    label cooks="Cook's D";
    format cooks 6.3;
run;
%end;
%else %do;
    data cooksds; Cooks=’No unusual Cooks D’; run;
ods region column_span=1;
proc print data=cooksds label; id cooksds; run;
%end;
%
%if %length(&classvar) and &NClassVals = 2 %then %do;
*----------------------------------------------------
Sensitivity analysis - identify values to exclude
This rule eliminates the values the furthest away from the mean for the variable
-------------------------------------------------------------------------------;
%local MeanVal1 OutlierVal1 NOutlierVal1;
proc sql noprint;
    select mean(&ThisVar) format=best16. into :MeanVal1 from &data;
    select &ThisVar format=best16. into :OutlierVal1 from &data
    having abs(&ThisVar - &MeanVal1) = max(abs(&ThisVar - &MeanVal1));
    select count(*) into :NOutlierVal1 from &data
    where &ThisVar = &OutlierVal1;
%do iExtreme = 2 %to &NExtreme;
%local MeanVal2 to &iExtreme OutlierVal2 to &iExtreme NOutlierVal2 to &iExtreme;
    select mean(&ThisVar) format=best16. into :MeanVal2 to &iExtreme from &data
    where &ThisVar ^= &OutlierVal1
    %if &iExtreme > 2 %then %do jExtreme=2 %to &iExtreme-1;
    and &ThisVar ^= &OutlierVal2 to &jExtreme
    %end;
    select &ThisVar format=best16. into :OutlierVal2 to &iExtreme from &data
    where &ThisVar ^= &OutlierVal1
    %if &iExtreme > 2 %then %do jExtreme=2 %to &iExtreme-1;
    and &ThisVar ^= &OutlierVal2 to &jExtreme
    %end;
    having abs(&ThisVar - &MeanVal2 to &iExtreme) = max(abs(&ThisVar - &MeanVal2 to &iExtreme));
    select count(*) into :NOutlierVal2 to &iExtreme from &data
    where &ThisVar = &OutlierVal2 to &iExtreme
%end;
%end;
quit;

*----------------------------------------------------
Sensitivity analysis - run Proc GLM excluding different numbers of outliers
iExtreme=1 corresponds to the most extreme outlier value
-------------------------------------------------------------------------------;
%do iExtreme = 1 %to &NExtreme;
ods exclude all;
ods output lsmeans=lsmeansiExtreme(drop=effect dependent stderr probt);
proc glm data=&data(keep=&IDVar &ClassVar &ThisVar &CoVar) ;
    where not ( %let connect=; %do jExtreme=1 %to &iExtreme;
    %connect &ThisVar=&&OutlierVal2 to &jExtreme
    %let connect=or;
    %end; 
    class &ClassVar;
    model &ThisVar = &ClassVar &CoVar;
lsmeans &ClassVar / pdiff stderr;
quit;
ods output close;
ods select all;
%end;

%--------------------------------------------------------------------------
% Sensitivity analysis - combine outputs from different Proc GLM runs
%--------------------------------------------------------------------------;
data lsmeansall;
  set lsmeans0(in=in0)
    %do iExtreme = 1 %to &NExtreme;
      lsmeans%iExtreme(in=in%iExtreme)
    %end;
  if in0 then NDelete=0;
  %do iExtreme = 1 %to &NExtreme;
    else if in%iExtreme then NDelete=iExtreme;
  %end;
  label NDelete='# Obs Deleted';
run;
proc transpose data=lsmeansall out=lsmeanst prefix=LSMean_;
  by NDelete;
  id &classvar;
  var LSMean;
run;

data lsmeanstplus;
  merge lsmeanst(drop=_name_ _label_)
    lsmeansall(where=(missing(ProbTDiff)=0) keep=NDelete ProbTDiff);
  by NDelete;
  array lsm {*} lmean_:
  do i = 1 to dim(lsm);
    lsm[i]=round(lsm[i],.01);
  end;
  if dim(lsm)=2 then LSMeanDiff=lsm[2] - lsm[1] ;
  if NDelete > 0 then do;
    ValueDeleted = symgetn('OutlierValue' || left(put(NDelete,best3.)) );
    NValueDeleted = symgetn('NOutlierValue' || left(put(NDelete,best3.)) );
  end;
  ProbTDiff=round(ProbTDiff,.001);
  format lsmean_: comma16.2  probtdiff 6.3;
run;
ods region column_span=1 ;
proc print data=lsmeanstplus label ;
  id ValueDeleted ;
  var NValueDeleted lmean: probtdiff;
  label
    ValueDeleted = "Value Deleted"
    NValueDeleted = "# Obs with This Value";
run;
%end; %* sensitivity analysis if there is a class variable;
ods region column_span=2 height=.1in;
ods pdf text = "Model: %upcase(&ThisVar) = %upcase(&ClassVar) %upcase(&CoVar)"
%end; %* loop for each variable;
%--------------------------------------------------------------------------
Close report
%--------------------------------------------------------------------------;
ods layout end;
ods pdf close;
ods listing;
title;
%quit:
%mend outlierchk;

