Using Cardinality Ratio for Fast Data Review
Ronald J. Fehd, Stakana Analytics, Atlanta, GA, USA

Abstract

Description: This paper reviews the database concept Cardinality Ratio. The SAS® frequency procedure can produce an output data set with a list of the values of a variable. The number of observations of that data set is called N-Levels. The quotient of N-Levels divided by the number-of-observations of the data is the variable’s Cardinality Ratio (CR). Its range is in \([0–1]\) and four categories of variables’ CR are obvious: useless, discrete, continuous and unique.

Purpose: Cardinality Ratio provides an important value during data review. The four groups of values are used to create control data sets which can automate data review. Macros are provided that facilitate fast data review.

Audience: data managers and programmers.

Programs: in this paper are available in Fehd [5].

Keywords: continuous, database, dimensionless, discrete, fact, frequency, keys: foreign or primary, nlevels, number of observations (nobs), unique

Quote: Information is the difference that makes a difference. Gregory Bateson 1904–1980 Steps to an Ecology of Mind, 1972 italics added by R.J. Fehd

Contents

Programs

<table>
<thead>
<tr>
<th>Programs</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>cr-calc.sas</td>
<td>4</td>
</tr>
<tr>
<td>listvars.sas</td>
<td>7</td>
</tr>
<tr>
<td>cr-data.sas</td>
<td>8</td>
</tr>
<tr>
<td>cr-freq.sas</td>
<td>9</td>
</tr>
<tr>
<td>cr-mode.sas</td>
<td>10</td>
</tr>
<tr>
<td>cr-smry.sas</td>
<td>10</td>
</tr>
</tbody>
</table>

Unit Tests

<table>
<thead>
<tr>
<th>Unit Tests</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>cr-calc-test-class.sas</td>
<td>12</td>
</tr>
<tr>
<td>cr-calc-test-class-with-format.sas</td>
<td>13</td>
</tr>
<tr>
<td>cr-calc-test-shoes.sas</td>
<td>13</td>
</tr>
<tr>
<td>cr-calc-test-sashelp-all.sas</td>
<td>14</td>
</tr>
</tbody>
</table>

Summary

<table>
<thead>
<tr>
<th>Summary</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>15</td>
</tr>
</tbody>
</table>

Bibliography

<table>
<thead>
<tr>
<th>Bibliography</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>16</td>
</tr>
</tbody>
</table>
Introduction

Overview

When starting a new project a programmer or data detective may use several procedures during data discovery to confirm the relationships between variables in a data set.

The cardinality ratio (CR) of a variable is the quotient of the number of levels of that variable divided by the number of rows of the data set. The dimension: n-rows of numerator and denominator, cancels out leaving a pure number in the range of \(>0\) – one.

CR is similar in concept to the log function: it reduces large numbers to a finite range which makes comparisons easier to grasp.

The Sets of Values

CR can be grouped into four categories.

- continuous: \(\text{CR} \geq 0.5\)
- discrete: \(\text{CR} \leq 0.5\)
- unique: \(\text{CR} = 1\)
- single-valued: \(n\)-levels = 1

Note: One-half (0.5) is an arbitrary separation value.

continuous: information: is a fact variable; if numeric can be summarized

discrete: indicators: character variables have standardized case: either upper or lower; numerics may be integers, or in a small finite range

information: is a classification variable;

todo: locate one-to-one formats, or dimension (lookup) tables

unique: variable is a row-identifier; if numeric and the range is exactly 1:n-observations then it is the row-number

information: is a primary key;

single-valued: values may be:

- character: blank
- numeric: missing
- a single value, indicating data set is a subset of larger data

information: worthless, discard

Task

The task of this article is to show that the mean of the cardinality ratios separates the discrete from the continuous; the former is summarized with the frequency procedure and the latter with the summary procedure.

Continued on next page.
**Algorithm of Calculator**

These are the steps in the cardinality-ratio calculator program.

1. **listvars**: make list of variable names from contents
2. **freq**: make list of n-levels
   - out= print n-levels
3. sort and merge contents and frequency output data sets
   - calculate cardinality ratio
   - out=CR-list
4. calculate mean of CR
5. add mean to CR-list
   - assign categories: unique, discrete, continuous
   - make lists for frequency of discrete,
   - mode of either, and summary of continuous
6. print CR-list with categories
7. for each list, call macros of frequency, mode, summary,
   - out = CR-summaries
8. print CR-summaries

**Note:** This algorithm is from Fehd [7], SmryEachVar and Fehd [9], Cardinality Ratio.

**Algorithm of Procedures**

These are the steps in the cardinality-ratio summarization routines, cr-frequency, cr-mode and cr-summary.

1. procedure, out=out-from-proc
2. data for-append, standardize data structure: cr data
3. append
   - base = CR-summaries
   - data = out-from-proc

**Common Data Structure**

This is the common data structure provided by the subroutine cr-data.

```plaintext
create table WORK.OUT_FREQUENCIES( bufsize=16384 )
libname char(8),
memname char(32),
name char(32),
valu_c char(32),
valu_txt char(32),
valu_n num,
count num format=COMMA. label='N',
percent num format=6.2 label='-%-' 
```

**Notes:** Each procedure renames the variable according to its type. Character variables are renamed to valu_c; numeric variables are renamed to valu_n; the mode procedure and formatted values are written into valu_txt.
Using Cardinality Ratio for Fast Data Review, continued

Programs

Overview

This section lists the suite of programs that produce a report of the cardinality ratios of variables in a data set.

- cardinality ratio calculator
- listvars, make list of variables from contents
- cr-data, standardized data structure for cr-*
- cr-frequency
- cr-mode, proc mode is proc freq order = freq
- cr-summary

Note: The listing of list-processing macro callmacr.sas is in Fehd [11].

---

cr-calc.sas

```sas
/* name: <UNC>\SAS-site\macros\cr_calc.sas
author: Ronald J. Fehd 2012, 2013
-------------------------------------------
Summary : description : cardinality ratio calculator
purpose : data review
-------------------------------------------
Contexts : program group: list processing program
program type: module
SAS type: macro routine
uses routines : callmacr to call:
uses subroutines: cr_freq, cr_mode, cr_smry
-------------------------------------------
Specifications: input : required: libname, memname
process: contents out=
freq levels out=
data: merge contents + freq,
calculate CR
summary: mean of CR
data: add category is_a,
output various
output: save and print: out-lib.out-data
-------------------------------------------
Parameters : libname : actually the libref
memname : data set name
out_lib : work | library
out_data : check the default: cr_&memname
out_smry : kest_lib..smry_ouy_data
-------------------------------------------
Usage Example:
*options mprint source2;
%cr_calc(libname = sashelp,memname = shoes)
Notes: * certain data sets will generate these warnings:
WARNING: The variable nmisslevels in the DROP, KEEP,
or RENAME list has never been referenced.
WARNING: The variable nonmisslevels in the DROP, KEEP,
or RENAME list has never been referenced.
when present, these variables are in:
create table FREQ_N_LEVELS
(name char(14) label='Number of Variable Levels',
TableVarLabel char(28) label='Table Variable Label',
n_levels num format=BEST8. label='Number of Levels',
NMissingLevels num format=BEST8. label='Number of Missing Levels',
NNonMissingLevels num format=BEST8.
-------------------------------------------
```
Using Cardinality Ratio for Fast Data Review, continued

%MACRO cr_calc
   (libname = sashelp
     ,memname = class
     ,out_lib = work
     ,out_data = cr_&memname
     ,out_smry = cr_&memname._smry
     ,mode_show = 3
     ,mode_nlevels = 33 /* fib: 1 1 2 3 4 8 13 21 33 */
     ,testing = 0)
%local out_contents out_freq out_from_procs;
%let out_contents = list_names;
%let out_freq = freq_n_levels;
%let out_from_procs = cr_summaries;
%let testing = %eval( not(0 eq &testing)
or %sysfunc(getoption(mprint)) eq %upcase(mprint));
**** make data set with nlevels;
PROC freq data = &libname..&memname
   nlevels ;
ods exclude onewayfreqs;
ods output nlevels = &out_freq
   (rename = (tablevar = name
   nlevels = n_levels));
title3 "n-levels of &libname..&memname";
%if &testing %then %do;
   PROC sql; describe table &syslast.;
   quit;
%end;
**** prepare for merge;
PROC sort data = &out_contents
   out = &out_contents;
by libname memname name;
PROC sort data = &out_freq
   out = &out_freq;
by name;
DATA &out_data
   (keep = card_ratio libname memname name type
    format label n_levels nobs
    nmisslevels nnonmisslevels);
   if 0 then do;
      attrib card_ratio length = 8
      label = ‘cardinality ratio’;
      set &out_contents (obs = 0);
      set &out_freq (obs = 0);
   end;
do until(endofv); 
merge &out_contents
   &out_freq end = endofv;
by name;
card_ratio = n_levels/nobs;
Using Cardinality Ratio for Fast Data Review, continued

libname = lowcase(libname);
memname = lowcase(memname);
output;
end;
call symputx('nobs',nobs);
stop;
run;

%if &testing %then %do;
PROC sql; describe table &syslast.;
quit;
%end;

**** calculate middle of CR range:
separates foreign keys::discrete and facts::continuous;
PROC summary data = &out_data mean;
var card_ratio;
output
out = mean
(keep = mean)
mean (card_ratio) = mean;
%if &testing %then %do;
PROC sql; describe table &syslast;
quit;
%end;

**** save mean for use in calculation, label and title;
PROC sql noprint;
select mean into :mean
from &syslast;
quit;
%put note: &sysmacroname mean: &mean;
run;
%if &mean eq 1 %then %do;
%put note: &sysmacroname exit: mean eq 1;
%return;
%end;

%let drop_list = drop = is_a card_ratio;
DATA &out_data
cr_list_freq(&drop_list)
cr_list_mode(&drop_list)
cr_list_smry(&drop_list);
keep is_a card_ratio libname memname name type format;
if 0 then do;
attrib is_a length = $ %length(_primary key?!);
set &out_data
(keep = card_ratio n_levels
libname memname name type format);
end;
set mean;
do until(endsof);
set &out_data end = endsof;
if card_ratio eq 1 then
is_a = ' primary key?!';
else if n_levels eq 1 then
is_a = 'foreign key';
else if card_ratio < mean then do;
if n_levels le &mode_nlevels then output cr_list_freq;
else output cr_list_mode;
end;
else if card_ratio > mean then do;
if type = 'c' then output cr_list_mode;
else if type = 'n' then output cr_list_smry;
end;
output &out_data;
end;
stop;
run;
%if &testing %then %do;
PROC sql; describe table &syslast.;
quit;
PROC print data =cr_list_freq/title cr_list_freq;
PROC print data =cr_list_mode/title cr_list_mode;
PROC print data =cr_list_smry/title cr_list_smry;
%end;
PROC sort data = &out_data
   out = &out_lib..&out_data
      (label="&out_lib..&out_data nobs=&nobs mean=&mean");
by   is_a name card_ratio;
PROC print data = &out_lib..&out_data label;
title3 *cardinality ratios of &libname..&memname*
   * nobs=&nobs mean=&mean*;
by   is_o;
run;
*** use routine to call subroutines;
%callmacr(data = cr_list_freq
   ,macro_name = cr_freq
   ,hex16 = 0)
*** note: hex16 = 0 :: pass varnum as integer;
%callmacr(data = cr_list_mode
   ,macro_name = cr_mode
   ,hex16 = 0
   ,macro_parms = %nrstr(mode_hi_lo=&mode_show)
   ,hex16 = 0 )
%callmacr(data = cr_list_smry
   ,macro_name = cr_smry
   ,hex16 = 0
   ,macro_parms = %nrstr(mode_hi_lo=&mode_show)
   ,hex16 = 0 )
%callmacr(data = cr_list_smry
   ,macro_name = cr_univ
   ,hex16 = 0
   ,macro_parms = %nrstr(mode_hi_lo=&mode_show)
   ,hex16 = 0 );
%if not(%sysfunc(exist(&out_from_procs))) %then %do;
   %put note: 0.1 &sysmacroname exit not exist(&out_from_procs);
%return;
%end;
PROC datasets library = work
   nodetails nolist;
   change
   &out_from_procs = &out_smry;
%if %lowcase(&out_lib) ne work %then %do;
   PROC copy in = work
   out = &out_lib;
   select &out_smry;
%end;
proc print data = &out_lib..&out_smry label;
title4 &out_lib..&out_smry;
run;
title3;
%mend cr_calc;

listvars.sas
/*
 * name: <UNC>\SAS-site\macros\listvars.sas
 * Make List of Variable Names
 * description: make list of variable names
 * purpose: prepare list processing data set
 * input : data : _all_ | data set name
 * drop_keep: keep=libname memname name type
 * where : 1 | logical expression
 * process: proc contents
 * output: out =
 * usage:
 * %listvars(data = sashelp.class
 *   ,drop_keep =
 *   keep = libname memname name type format)
 * see also: http://www.sascommunity.org/wiki/Making_Lists */
%MACRO listvars
   (data = sashelp.class
   ,drop_keep =
   keep = type_n formatd formatl
   )
Using Cardinality Ratio for Fast Data Review, continued

PROC contents data = &data
noprint
out = &out
(keep = libname memname memtype varnum
name type length label format
formatd formatl nobs
rename = (type = type_n)
where = (&where
and memtype eq 'DATA') );

DATA &out (&drop_keep
label = "list vars of &data");
attrib libname length =$ 8 label = 'libname'
memname length = $32 label = 'mem name'
varnum length = 8 label = 'var num'
name_lc length = $32 label = 'low (var name)'
name length = $32 label = 'var name'
type length = $1 label = 'type'
length length = 8 label = 'length'
format length = $34 label = 'format'
label length = $256 label = 'label'
nobs length = 8 label = 'n obs';
do until(endofile);
set &out(drop = memtype)
end = endofile;
libname = lowcase(libname);
memname = lowcase(memname);
name_lc = lowcase( name);
* align format, type to sql.dictionary.columns:
1. convert type from numeric to $char1
2. add char.prefix $ and suffix dot to formats;
if format ne ' ' then do;
if formatl then format = cats(format,formatl,'.');
else format = cats(format ,'.');
if formatd then format = cats(format,formatd);
end;
/**************************
if format ne ' ' then
format = catt(format,'.');
if format in ('$','$.' then
format = ' ';*********************/
if type_n eq 1 then type = 'n';
else if type_n eq 2 then type = 'c';
output &out;
end;
stop;
run;
%if &testing %then %do;
proc sql; describe table &out;
quit;
%end;
%mend listvars;

---
cr-data.sas---

/*
name: <UNC>\SAS-site\macros\cr_data.sas
author: Ronald J. Fehd 2013
Summary : description : common data structure of cr_*
purpose : standardization
Contexts: program group: list processing program
program type: subroutine
SAS type: macro subroutine
called by routines cr_* ***** /
%MACRO cr_data(_)
/ des = "site: card. ratio suite: common data structure'\nkeep _libname _memname _name
_valu_c _valu_txt _valu_n _count _percent;
Using Cardinality Ratio for Fast Data Review, continued

```sas
attrib _libname length = $8 label = 'libname'
_memname length = $32 label = 'memname'
_name length = $32 label = 'name'
_valu_c length = $32 label = 'valu c'
_valu_txt length = $32 label = 'valu txt'
_valu_n length = 8 label = 'valu n'
_count length = 8 format = comma label = 'N'
_percent length = 8 format = 6.2 label = '-%-
retain _libname "%lowcase(&libname)"
_memname "%lowcase(&memname)"
_name "&name"
_valu_c '.' _valu_txt '.
_valu_n . _count . _percent .;
%mend cr_data;
```

cr-freq.sas

```sas
/* name: cr_freq.sas
author: Ronald J. Fehd 2013
Summary : description : proc freq
purpose : standardize output
Contexts : program group: list processing program
type: routine
SAS type: macro routine
Specifications: input : libname memname name type format
process: freq out= ***/
%macro cr_freq
(libname =
_memname =
_name =
_type =
_format =
_out_append = cr_summaries
_out_data = out_frequencies
,testing = 0)
/des = 'site: list processing proc freq';
%let testing = %eval( not(0 eq &testing)
or %sysfunc(getoption(mprint)) eq %upcase(mprint));
PROC freq data = &libname..&memname;
format &name; *remove formats;
tables &name
/list missing nopr
out = &out_data
(rename = (&name = _valu_&type
_count = _count
_percent = _percent));
DATA &out_data;
%cr_data(attributes);
do until(endof); set &out_data
  end = endof;
%if &format ne %then %do;
  _valu_txt = put(_valu_&type,&format);
%end;
output;
end;
stop;
run;
%if &testing %then %do;
PROC sql; describe table &syslast;
quit;
%end;
PROC append base = &out_append
  force
  data = &out_data;
PROC delete data = &out_data;
run;
%mend cr_freq;
```
cr-mode.sas

```sas
/* name: cr_mode.sas
author: Ronald J. Fehd 2013
Summary: description: calculate modes, hi and low
purpose: standardize output
Contexts: program group: list processing program
program type: routine
SAS type: macro routine
Specifications: input: libname, memname, name, type
process: freq order=freq out= ***/

%macro cr_mode

(libname = ,memname = ,name = ,type = ,format =
,mode_hi_lo = 3 ,out_append = cr_summary
,out_data = out_modes ,testing = 0)
/des = 'site: card. ratio suite: proc mode';
%let testing = %eval( not(0 eq &testing)
or %sysfunc(getoption(mprint)) eq %upcase(mprint));

PROC freq data = &libname..&memname
  order = freq;
format &name; *remove formats;
tables &name
/list missing noprint
out = &out_data
(rename = (&name = _valu_&type
  count = _count
  percent = _percent));

DATA &out_data;
%cr_data(attributes);

do rownmbr = 1 to &mode_hi_lo;
  set &out_data nobs = n_rows point = rownmbr;
  _valu_txt = "mode: high &mode_hi_lo"
  output;
end;

do rownmbr = n_rows +1 - &mode_hi_lo to n_rows;
  set &out_data nobs = n_rows point = rownmbr;
  _valu_txt = "mode: low &mode_hi_lo"
  output;
end;
stop;
run;

%if &testing %then %do;
PROC sql; describe table &syslast;
quit;
%end;

PROC append base = &out_append
  force
  data = &out_data;
PROC delete data = &out_data;
run;
%mend cr_mode;
```

cr-smry.sas

```sas
/* name: cr_smry.sas
author: Ronald J. Fehd 2013
Summary: description: summarize numeric variable
purpose: standardize output
Contexts: program group: list processing program
program type: routine
SAS type: macro routine
Specifications: input: libname memname name type format
process: summary out= ***/
```
%macro cr_smry
(libname = ,memname = ,name = ,type = ,format = ,out_append = cr_summaries ,out_data = out_summaries ,testing = 0)

/ des = 'site: card. ratio suite: proc summary';
%let testing = %eval( not(0 eq &testing) or %sysfunc(getoption(mprint)) eq %upcase(mprint));

PROC summary data = &libname..&memname;
format &name; *remove formats;
var &name;
output out = summary ( drop = _type_ _freq_)
n (&name) = n
nmiss (&name) = nmiss
min (&name) = min
std (&name) = std
p25 (&name) = p25 %*q1;
mean (&name) = mean %*average;
median(&name) = median %*p50::middle;
p75 (&name) = p75 %*q3;
max (&name) = max ;

PROC transpose data = summary
out = summary_t (keep = col1 _valu_c
(rename =(col1 = _valu_n ))
name = _valu_c ;
DATA &out_data;
%cr_data(attributes);
if 0 then set &libname..&memname(obs=0)
nobs = nobs_data;
do until(endofile);
  set summary_t end = endofile;
  if _valu_c in ('n','nmiss') then
    _percent = 100*(_valu_n/nobs_data);
  else _percent = .;
  %if &format ne %then %do;
    _valu_txt = put(_valu_&type,&format);
  %end;
  output;
  end;
  stop;
  run;
%if &testing %then %do;
  PROC sql; describe table &syslast;
  quit;
%end;

PROC append base = &out_append
force data = &out_data;
PROC delete data = &out_data;
PROC delete data = summary;
PROC delete data = summary_t;
run;
%mend cr_smry;
Using Cardinality Ratio for Fast Data Review, continued

Unit Tests

Overview

This section lists the set of programs used to do unit tests of the cardinality-ratio-calculator suite.

- cr-calc-test-class
- cr-calc-test-class-with-format, shows valu_txt with format
- cr-calc-test-shoes, shows mode in valu_txt
- cr-calc-test-sashelp-all, review all data sets in libref

---

**cr-calc-test-class.sas**

```sas
options mprint source2;
%cr_calc(libname = sashelp ,memname = class)
```

**report**

```
Number of Variable Levels
------- ------
Name    19
Sex     2
Age     6
Height  17
Weight  15

Cardinality ratios of sashelp.class nobs=19 mean=0.621053

<table>
<thead>
<tr>
<th>name</th>
<th>type</th>
<th>format</th>
<th>cardinality ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>c</td>
<td></td>
<td>1.00000 sashelp class Name c</td>
</tr>
<tr>
<td>Height</td>
<td>n</td>
<td></td>
<td>0.89474 sashelp class Height n</td>
</tr>
<tr>
<td>Weight</td>
<td>n</td>
<td></td>
<td>0.78947 sashelp class Weight n</td>
</tr>
</tbody>
</table>

| work.cr_class_smry libname memname name valu_c valu_txt valu_n count percent |
|--------------------|----------|--------|---------|---------|--------|
| sashelp class Age  | .        | 11.000 | 2       | 10.53   |
| sashelp class Age  | .        | 12.000 | 5       | 26.32   |

...  
| sashelp class Height | n       | 19.000 | .       | 100.00  |
| sashelp class Height | nmiss   | 0.000  | .       | 0.00    |
| sashelp class Height | min     | 51.300 | .       |         |
| sashelp class Height | mean    | 62.337 | .       |         |
| sashelp class Height | median  | 62.800 | .       |         |

...  
| sashelp class Weight | max     | 72.000 | .       |         |
| sashelp class Weight | n       | 19.000 | .       | 100.00  |
| sashelp class Weight | nmiss   | 0.000  | .       | 0.00    |

...  
| sashelp class Weight | max     | 150.000| .       |         |
```

Continued on next page.
This test program shows a report of a variable with a format.

```sas
options mprint source2;
proc format;
value $sex 'F' = 'female'
'M' = 'male'
% 'I' = 'intersex' %*;
% other = 'unknown' %*;
;
data class_with_format;
   if 0 then set sashelp.class;
   attrib sex length = $1 format = $sex.;
do until(EndOfFile);
   set sashelp.class end = endofile;
   output;
end;
Name = 'Chris'; Sex = 'I';
output;
stop;
run;
proc sql; describe table &syslast;
quit;
%cr_calc(libname = work
,memname = class_with_format)
```

Formats of values are in variable `valu_txt`.

```sas
| cardinality ratios of work.class_with_format nobs=20 mean= 0.61 |
|-------------------|-----------------|-----------------|
|                  | card_            |...              |
|                  | is_a ratio libname memname name type format |
|                  | foreign key 0.15 work class_with_format sex c $SEX. |
| libname memname name valu_c valu_txt valu_n count percent |
|                  | work class_with_format sex F female . 9 45.00 |
|                  | work class_with_format sex I I . 1 5.00 |
|                  | work class_with_format sex M male . 10 50.00 |
```

This test program shows output from the mode procedure.

```sas
options mprint source2;
%cr_calc(libname = sashelp
,memname = shoes)
```

Variable Subsidiary is a foreign key with n-levels greater than 33, the default value of `cr_calc` parameter `mode_nlevels`; instead of `proc freq`, the mode macro is used. Note the high values in variable `valu_txt`.

```sas
<table>
<thead>
<tr>
<th>Variable</th>
<th>Label</th>
<th>Levels</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subsidiary</td>
<td></td>
<td>53</td>
</tr>
<tr>
<td></td>
<td>card_</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>is_a</td>
<td>ratio libname memname name type format</td>
</tr>
<tr>
<td></td>
<td>foreign key 0.13418 sashelp shoes Subsidiary c</td>
<td></td>
</tr>
<tr>
<td>libname memname name valu_c valu_txt valu_n count percent</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>sashelp shoes Subsidiary Addis Ababa mode: high 3 . 8 2.03</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sashelp shoes Subsidiary Al-Khobar mode: high 3 . 8 2.03</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sashelp shoes Subsidiary Budapest mode: high 3 . 8 2.03</td>
<td></td>
</tr>
</tbody>
</table>
```
This program is a stress test. It runs `cr_calc` on each data set in `libref sashelp`. This program duplicates the `SmryEachVar` suite, Fehd [7].

```sas
%let in_libref = sashelp;

*** make list of variables;
%listvars(data = &in_libref.._all_,
   drop_keep = libname memname name type format nobs,
   where = nobs,
   out = list_vars_from_contents)

*** make list of memnames from list_vars;
proc sort data = &syslast
   (keep = libname memname)
   nodupkey;
by libname memname;
proc print data = &syslast;
   title3 &syslast of &in_libref;
run;
%callmacr(data = &syslast,
   macro_name = cr_calc)
```
Summary

Cardinality Ratio is valuable information to have in data review. Its small finite range is easier to parse for meaning than the constantly changing and larger number-of-observations (nobs) of the data set. The mean of CR offers an easy way to differentiate between discrete and continuous variables, which are processed with frequency for the former and summary for the latter.

Conclusion

Further Reading

Programs: for this paper are in Fehd [5] sco.Cardinality-Ratio

Predecessors: This paper was first published in Fehd [9] sgl2013.299.

Cardinality Ratio is identified in Fehd [8] wuss2008.Database-Vocabulary for which SmryEachVar is the predecessor.

Programs for SmryEachVar which includes calculations for Cardinality Ratio are here: Fehd [6] sco.SmryEachVar.

Call-Macro: The list processing routine Call-Macro is described in Fehd [11].

Theory: The macros in this article illustrate the macro design ideas presented in Fehd [10] sgl2014.1899.


References


Using Cardinality Ratio for Fast Data Review, continued


Closure

Contact Information:

Ronald J. Fehd mailto:Ron.Fehd.macro.maven@gmail.com
http://www.sascommunity.org/wiki/Ronald_J._Fehd

About the author:

education: B.S. Computer Science, U/Hawaii, 1986
SAS User Group conference attendee since 1989
SAS-L reader since 1994

experience: programmer: 25+ years
data manager using SAS: 17+ years
statistical software help desk: 7+ years
author: 30+ SUG papers
sasCommunity.org: 300+ pages

SAS-L: author: 6,000+ messages to SAS-L since 1997

Trademarks

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 trademarks of their respective companies.