Paper 005-2008

Range Formats Made Easy – A CreateRangeFormat Custom Task for SAS® Enterprise Guide®

Petr Husták, Institute for Advanced Analytics, North Carolina State University, Raleigh, NC

ABSTRACT

SAS® formats are often used on numeric variables to assign discrete labels to ranges of values such as “below 20” or “20-40”. SAS Enterprise Guide offers the “Create Format” task, which lets you define ranges and assign labels to these ranges. However building range formats manually can be very laborious. This custom task for SAS Enterprise Guide creates range formats automatically for multiple variables based on their value distributions.

INTRODUCTION

One of the most common uses for SAS formats is to assign distinct labels to ranges of numeric values. The bvariable “age” for example can be formatted to ranges labeled “below 20”, “20-40” and so on. Discrete formatted values can then be easily used in frequency tables or graphs. Creating range formats with the FORMAT procedure can be very laborious. This paper introduces the CreateRangeFormat custom task for SAS Enterprise Guide that creates range formats easily for multiple dataset variables at one time. The task automatically determines value ranges and assigns self descriptive labels to them. The task also offers a wide range of customization options. In addition to the actual formats, the task creates a new dataset with formats applied to variables. By default, the task also produces an HTML report containing definitions and frequency graphs for all the newly created formats.

The first section of this paper describes the CreateRangeFormat task from user’s point of view. It can be used as a reference guide for the task. The following sections describe the underlying SAS macro programs and C# implementation of the task.

USING THE CREATERANGEFORMAT PLUGIN IN SAS ENTERPRISE GUIDE

This section will guide you through the process of installation and use of the “Create Range Format” custom task.

INSTALLATION

Before you can use any custom task in SAS Enterprise Guide, you need to follow a few installation steps. Installation is necessary only once.

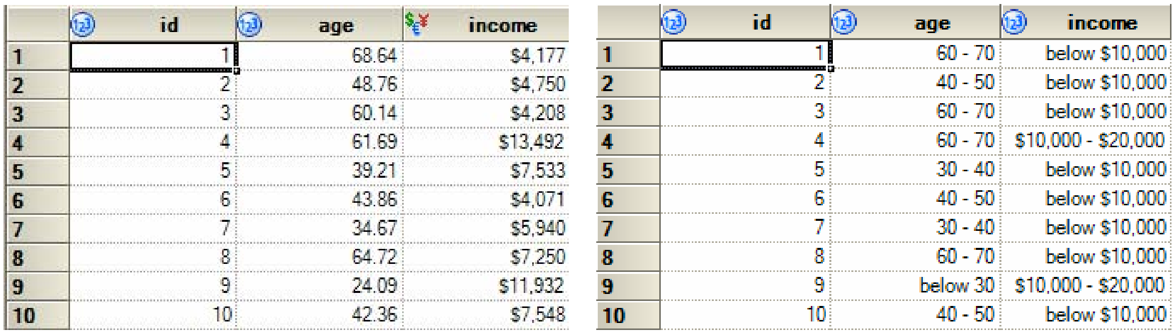
1. Download the CreateRangeFormat custom task dll file from http://www4.ncsu.edu/~phustak/CreateRangeFormatAddIn.dll and save it to a permanent location on your hard drive.
2. Open SAS Enterprise Guide and select “Add-In” and then “Add-In Manager” from the menu bar.
3. In the Add-In manager window click the “Browse” button and locate the “CreateRangeFormatAddIn.dll” file that you downloaded in step 1 and press the Open button.
4. Click OK to close the Add-In manager.

STEP-BY-STEP USE OF THE TASK

In its simplest form, the task can be run in the following basic steps. For further options, please refer to the next section, “Advanced Options”.

1. **Start with a dataset that contains one or more numeric variables.** Insert the dataset in the SAS Enterprise Guide process flow diagram. A sample dataset, CUSTOMERS, is shown in Figure 1. It contains two numeric variables “age” and “income” that are candidates for a range format.
2. **Add the CreateRangeFormat task to your process flow diagram.** This can be achieved either by selecting “Add-In” -> “Create Range Format” from the menu bar or by selecting the “Create Range Format” task from the SAS Enterprise Guide Task List. If you are not able to find the task, please repeat the task installation steps from previous section.

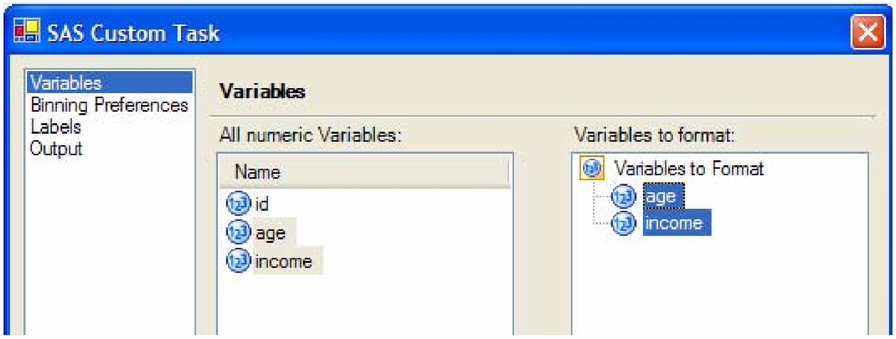
3. **Select variables to format.** After the task window opens, use the variable selector on the “Variables selection” pane to choose the variable or variables to format. In this example it makes sense to select the “age” and “income” variables. This step is illustrated in Figure 2.
4. **Select formatting preferences.** Numerous preferences can be changed in the “Binning Preferences”, “Labels” and “Output” selection panes to suit your needs. Now we will stick with the default settings. Please refer to the next section, “Advanced Options” for explanation of all customizations that are available.
5. **Run the Task.** The task runs automatically after clicking the OK button.
6. **Explore the results.** By default, the task produces two results (see Figure 3) – a dataset with formatted values and an HTML report. Example of a formatted dataset output is in Figure 1. You can compare both source and resulting datasets in Figure 1 to see the effect of running the task. An example of the HTML report can be found in Figure 4. The report contains a table with all created formats. The first column contains format name, the second column contains a complete format definition. This definition can be directly used in the PROC FORMAT value statement. It is possible to recreate the formats in base SAS easily just by copying and pasting this code. Clicking on the link in the third column takes you to a graph with frequencies for the format labels. See Figure 5 for an example for the “age” variable.



	id	age	income
1	1	68.64	$4,177
2	2	48.76	$4,750
3	3	60.14	$4,208
4	4	61.69	$13,492
5	5	39.21	$7,533
6	6	43.86	$4,071
7	7	34.67	$5,940
8	8	64.72	$7,250
9	9	24.09	$11,932
10	10	42.36	$7,548

	id	age	income
1	1	60 - 70	below $10,000
2	2	40 - 50	below $10,000
3	3	60 - 70	below $10,000
4	4	60 - 70	$10,000 - $20,000
5	5	30 - 40	below $10,000
6	6	40 - 50	below $10,000
7	7	30 - 40	below $10,000
8	8	60 - 70	below $10,000
9	9	below 30	$10,000 - $20,000
10	10	40 - 50	below $10,000

Figure 1. *CUSTOMERS - Example of a source (unformatted) and resulting (formatted) datasets*



SAS Custom Task

Variables
Binning Preferences
Labels
Output

Variables

All numeric Variables:

Name
id
age
income

Variables to format:

Variables to Format
age
income

Figure 2. *Selecting variables to format*

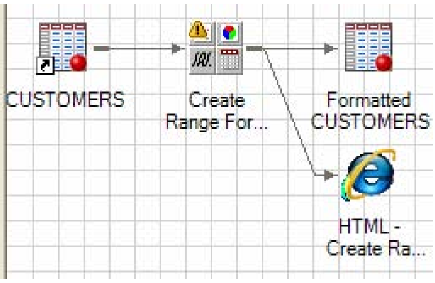
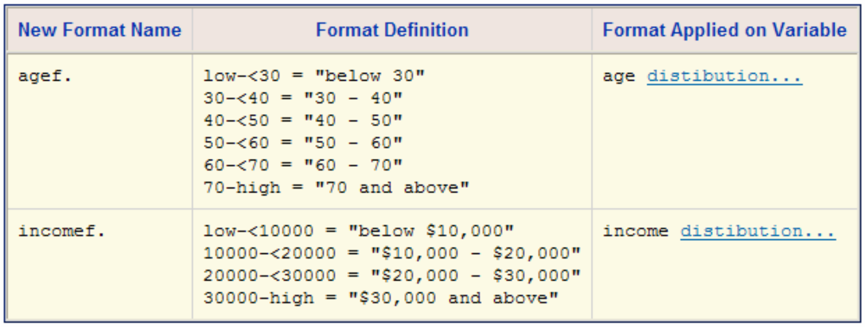


Figure 3. *Default task results – dataset and HTML report*



New Format Name	Format Definition	Format Applied on Variable
agef.	low-<30 = "below 30" 30-<40 = "30 - 40" 40-<50 = "40 - 50" 50-<60 = "50 - 60" 60-<70 = "60 - 70" 70-high = "70 and above"	age distibution...
incomef.	low-<10000 = "below $10,000" 10000-<20000 = "$10,000 - $20,000" 20000-<30000 = "$20,000 - $30,000" 30000-high = "$30,000 and above"	income distibution...

***Figure 4**. Example of an HTML report*

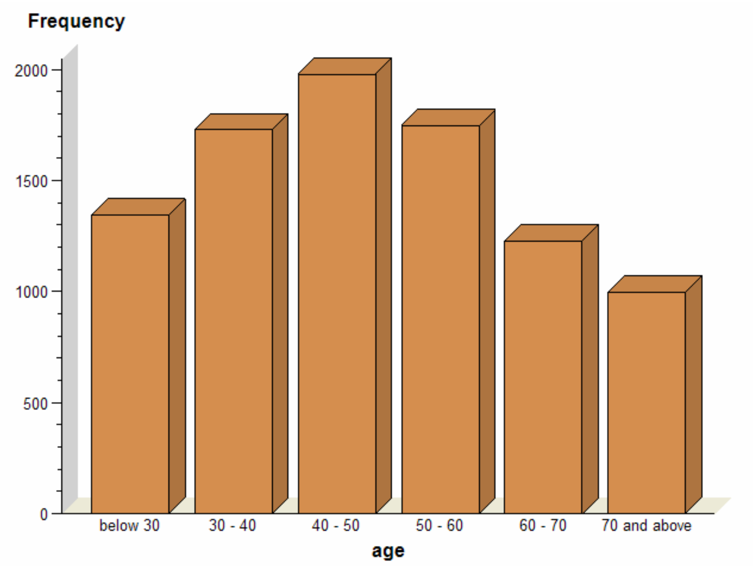
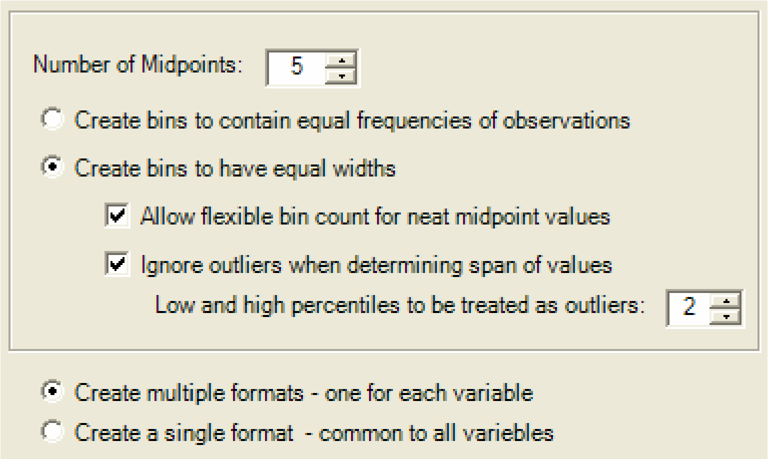


***Figure 5**. Example of label (formatted value) frequency graph*

ADVANCED TASK PREFERENCES

By following the steps in the previous section, you can create formats with minimum effort without having to set the task preferences. This section contains explanations of all the customization possibilities that were skipped in the step 4 of the previous section.



Number of Midpoints: 5

○ Create bins to contain equal frequencies of observations

● Create bins to have equal widths

☑ Allow flexible bin count for neat midpoint values

☑ Ignore outliers when determining span of values

Low and high percentiles to be treated as outliers: 2

● Create multiple formats - one for each variable

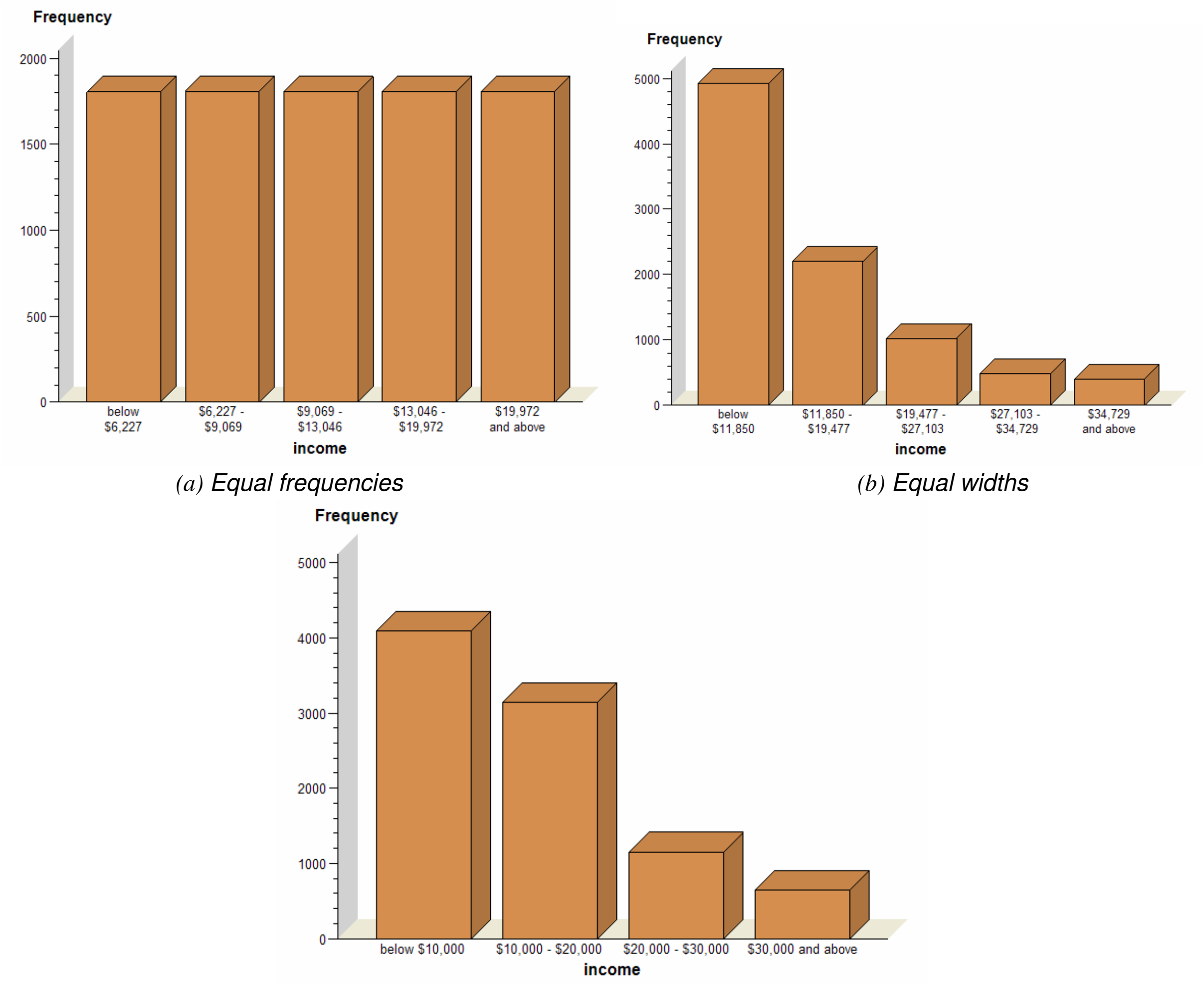
○ Create a single format - common to all varieble

***Figure 6**. Binning Preferences selection pane*

Binning Preferences selection pane

The layout of this control is in Figure 6. It can be used to customize how ranges for the bins will be computed.

- **Number of bins.** Specify how many ranges should be created. If the “Allow flexible bin counts…” option is selected, the resulting number of bins will not be exactly equal to the requested count.
- **Create bins to contain equal frequencies of observations.** When this option is selected, the ranges will not have equal width, but each range will contain equal number of observations. The frequency graph in Figure 7a was created with this option selected. Notice that each column has equal height, but the ranges on x axis are different for each column. Ranges are determined from the quantiles of the source variable distribution.
- **Create bins to have equal widths.** This option requires that all ranges have the same width. By default the ranges are computed by dividing the total span of the variable by the requested number of bins. The effect can be seen on the frequency graph in Figure 7b. Notice that all ranges have equal widths of approximately $7,626. The column heights for the ranges now mirror the distribution of he original variable.



(a) Equal frequencies

(b) Equal widths

(c) Equal widths with flexible bin count

Figure 7. *Formatted value frequency graphs for different binning options*

- **Allow flexible midpoint count for clean midpoint values.** This option preserves equal bin widths, but forces the range limits to have “nice” or “user friendly” values. “Nice” values are defined as multiples of one of the numbers in this series: … 0.1, 0.2, 0.5, 1, 2, 5, 10, 20, 50 …. Examples of ranges can then be: 1-2, 2-3, … or: 0.4–0.6, 0.6-0.8… or 20-30, 30-40, … To fulfill this constraint, the number of midpoints needs to be changed in some cases. Figure 7c displays and example of a frequency distribution utilizing this option.

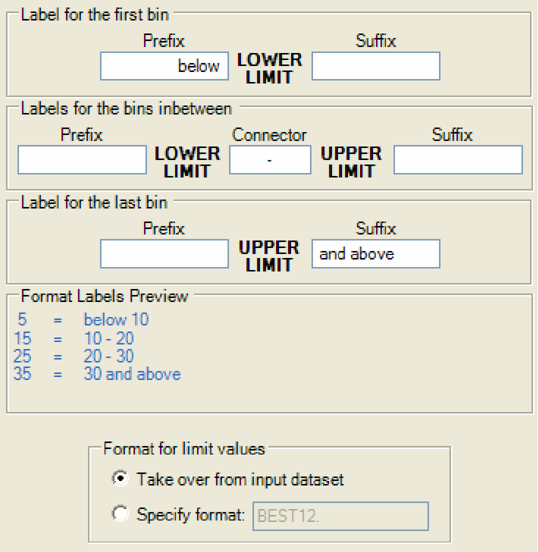
Compare how the ranges are more user friendly compared to 7b. Also notice that there are only 4 columns in the graph although 5 bins were requested.

- **Ignore outliers when determining span of values.** This option applies when bins with equal widths are created. By default, the span of values for bin computation is determined as the interval between minimum and maximum values of the formatted variable. However when the variable contains outliers, the maximum and minimum values don't characterize its distribution very well. For example, if all ages are between 0 and 100 and there is one outlier with value 10,000, all observations except the outlier will end in the very first bin. To avoid this problem, the span of values can be determined as the interval between the n-th and (100-n)-th percentiles. In other words, the highest n% and the lowest n% values are filtered out before the span is computed. The "n" value is set in the following control.
- **Low and high percentiles to be treated as outliers.** Specify the "n" value for the previous option here.
- **Create multiple formats/Create a single format.** This option takes effect when multiple variables are selected. You can either create a different format for each variable or you can create just one format and apply it to all variables. For the dataset in Figure 1, it makes sense to create multiple formats because age and income variables have very different values. On the other hand, if you have 12 variables with incomes for each month of the year, it would be better to create one format that suits all variables.

Labels selection pane

The layout of this control is in Figure 8. Here you can customize how the format labels will be constructed. In most cases you can use the default settings.

- **The label for the first bin** is composed from three components: prefix, upper limit value and suffix. For example the label "younger than 10 years" has these components set to prefix = "younger than " and suffix = " years". Notice the spaces after the "than" word and before the "years" word. If you did not include these spaces, the text would not be separated from the limit value: "younger than10years".
- **An analogous label for the last bin** can be composed from tree components: prefix, lower limit value and suffix. For example "older than 90 years" can be achieved by setting prefix = "older than " and suffix = " years".
- **Labels for all other bins** have both upper and lower limits, prefix, suffix and a connector – text between the limits. An age example for this type of label can be "age between 30 and 40 years".
- **Format labels preview** – In this window, you can always see how your current label settings will look when applied to a sample variable.
- **Format limit values** – This option specifies how the limit values will be printed in the labels. By default, the format from the input dataset will be used. This is useful in most cases. For example, the variable income in Figure 1a has format "dollar10". In the Figure 1b you can see that the label limits are printed with the dollar sign. If you want to override this behavior, you can specify your own format. This can be useful when the input variable is not formatted or when its format is not suitable for printing the label.



Label for the first bin

Prefix: below LOWER LIMIT Suffix:

Labels for the bins inbetween

Prefix: LOWER LIMIT Connector: - UPPER LIMIT Suffix:

Label for the last bin

Prefix: UPPER LIMIT Suffix: and above

Format Labels Preview

```
5   =  below 10
15  =  10 - 20
25  =  20 - 30
35  =  30 and above
```

Format for limit values

● Take over from input dataset

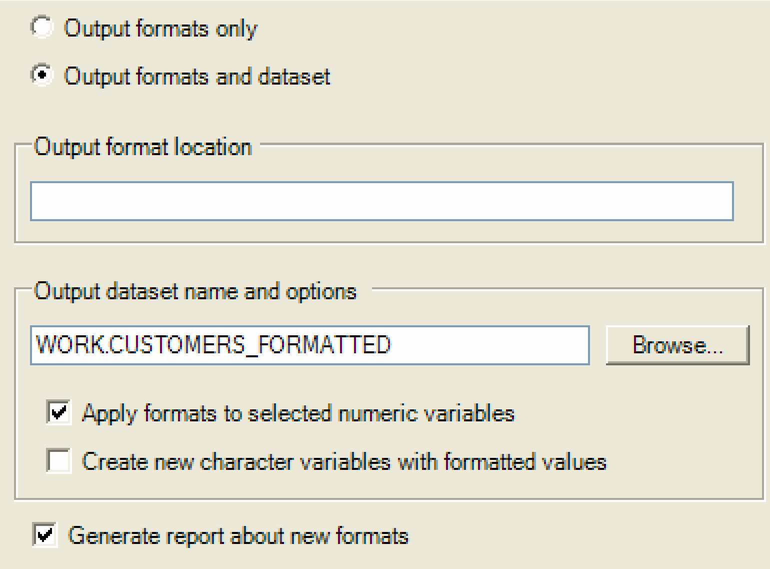
○ Specify format: BEST12.

***Figure 8**. Label selection pane*

Output selection pane

The layout of this control is in Figure 9. You can customize how the format labels will be constructed. In most cases you can use the default settings.

- **Output formats only / Output formats and datasets.** – If you are interested in creating only the formats for later use, you can disable generation of output dataset here.
- **Output formats location** – Specify the library and catalog where formats will be saved. Leave this field blank to store the formats in the default temporary WORK.FORMATS catalog. If one level library name is specified, the new formats will be stored in that library's FORMATS catalog. You can also specify a two level name in the form <library_name>.<catalog_name>.
- **Output dataset name** – Name of the output dataset. This can be either a one-level name for a temporary dataset or a two-level library.dataset name. You can use the browse button to browse available libraries.
- **Apply formats to selected numeric variables** – This option permanently assigns newly crated formats to the corresponding numeric variables in an output dataset. You might want to turn this option off if you are creating temporary formats that might not be available in the future. That might cause problems opening the dataset. In this case, it might be better to create character variables instead of applying formats to numeric variables.
- **Create new character variables with formatted values** – When this option is selected, then one new character variable will be created for each numeric input variable. This character variable will contain the formatted value of the corresponding numeric variable. For example, for the variable age=15 the new character variable will be cage="below 30".
- **Generate report** – Here you can disable or enable the generation of the html report.



○ Output formats only

● Output formats and dataset

Output format location

Output dataset name and options

WORK.CUSTOMERS_FORMATTED

☑ Apply formats to selected numeric variables

☐ Create new character variables with formatted values

☑ Generate report about new formats

***Figure 9**. Output selection pane*

SAS IMPLEMENTATION

This section contains implementation details that are not necessary for a user interested in using the CreateRangeFormat task in SAS Enterprise Guide. You might find this information helpful for insight on why you do not get expected results from the task or if you want to modify the task code to suit your specific needs.

GENERATED SAS PROGRAM STRUCTURE

Every time you run a task in SAS Enterprise Guide, SAS code is generated and executed on SAS server. The code that the CreateRangeFormat generates is complex, because all computations are made by the generated SAS code instead of by the task itself. The generated program consists of a list of macro definitions followed by calls to these macros. The number and types of macros generated depends on what options are selected in the task. The rules for all macros are summarized in Figure 10. The generated program ends with a datastep that creates the output dataset. All macros mentioned in Figure 10 can be downloaded from the following internet location: http://www4.ncsu.edu/~phustak/CreateRangeFormatMacros.sas.

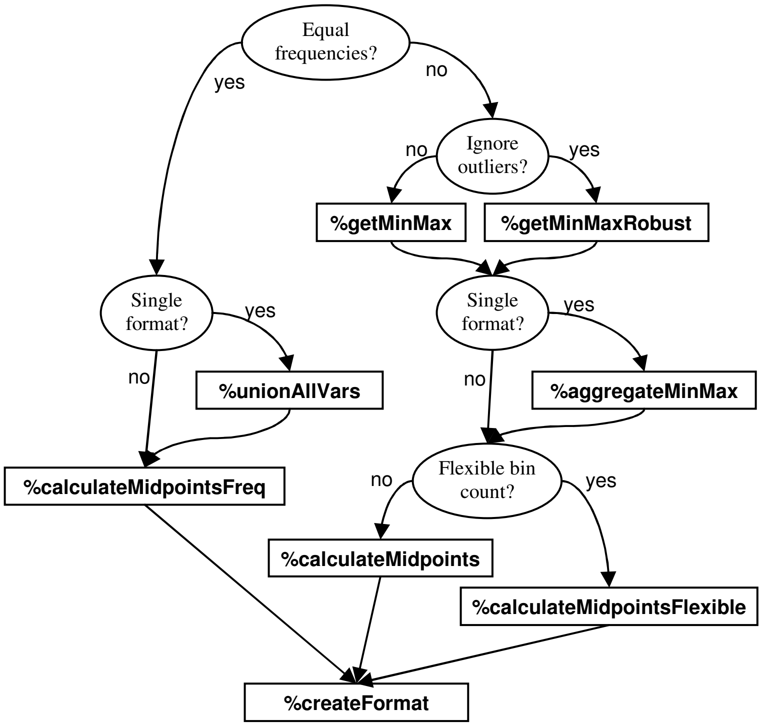


***Figure 10**. Macros generated and used by the task under different task settings*

RANGE COMPUTATION

The central task of the generated SAS code is to compute the ranges used in the new formats. Range computation differs significantly depending on the task options selected. All approaches are briefly described in the following paragraph.

1. **Equal frequencies, multiple formats** (macro %calculateMidpointsFreq) – At first, the required percentile numbers for all ranges are computed as (range number)*100%/(required number of bins). The MEANS procedure is then used to obtain actual percentile values from variable's distribution. These values are then used as range limits in the new format.
2. **Equal frequencies, single format** (macros %unionAllVars, %calculateMidpointsFreq) – Values of all variables selected to be formatted are appended under one variable name. This produces a new dataset with one variable and observation count given by the formula (number of variables)*(original number of observations). The procedure from point 1 is then applied to this new dataset.
3. **Equal widths, multiple formats** (macros %getMinMax, %calculateMidpoints) – This computation starts by obtaining minimum and maximum values of the variable by using the sql procedure. Limits for ranges of the new format are then determined as min+(range number)*(max - min)/(required bin count).
4. **Equal widths, multiple formats, ignore outliers** (macros %getMinMaxRobust, %calculateMidpoints) – The procedure is very similar to point 3. The only difference is that the range of the variable is not determined as min and max values, but as n-th and (100-n)th percentiles using the MEANS procedure.
5. **Equal widths, single format** (macros %getMinMax, %aggregateMinMax, %calculateMidpoints) – Minimum and maximum values are determined for each of the formatted variables. These values are then combined in a global minimum and maximum as $min_g = min(min_1 , min_2 , …)$ and $max_g = max(max_1 , max_2 , …)$. Range limits are computed from these extreme values by the formula described in point 3.
6. **Equal widths, single format, ignore outliers** (macros %getMinMaxRobust, %aggregateMinMax, %calculateMidpoints) – This is combination of points 5 and 4. The procedure from point 5 is slightly modified to create ranges in the n-th and (100-n)th percentiles using the MEANS procedure, instead of the min and max values.
7. **Equal widths, flexible bin count** (macro %calculateMidpointsFlexible) – A flexible bin count is a modification that can be applied to points 3, 4, 5 and 6. This option ensures that the range limits in the new format will have user-friendly values. The value for range width is selected from the approximately exponential series (… 0.1, 0.2, 0.5, 1, 2, 5, 10, 20, 50 …). To determine which member of the series fits best as a range width, the log of the raw range width computed as (max - min)/(required bin count) is

compared with logs of the candidate “nice” widths from the above series. The closest width on the logarithmic scale is then selected for the format. After the width is known, the positions of format ranges are then determined between the variable minimum and maximum.

C# IMPLEMENTATION

The user interface and the code generating procedures of the CreateRangeFormat task have been implemented in Miscrosoft Visual C# 2003. Alternatively, Visual Basic can be also used to build custom tasks. It is not possible though to use newer versions of Microsoft Visual C# (2005 and later), as they use .net framework 2.0, which is not supported by SAS Enterprise Guide 4.1. Some SAS users might be discouraged from developing a custom task for SAS Enterprise Guide because of the need to code in Visual C# or Visual Basic. Although it still requires some effort to make your first custom task, the SAS team has minimized the amount of work by providing a Visual C# project template and multiple sample custom tasks with complete source code.

The following section will concisely present the basic building blocks of the C# program. For your orientation, the location of these blocks in the provided Visual C# 2003 project will be specified.

USER INTERFACE

The visible part of the custom task contains controls in which the user can set preferences that will be used when the task is run. The user interface can be conveniently designed in the standard Visual Studio Form Designer. The Source code for the user interface is in the CreateRangeFormatForm.cs file.

CODE GENERATION

After the task is run, a SAS program is generated by the task and forwarded to SAS Enterprise Guide that runs it in SAS interpreter. The code generating procedure needs to take into account the preferences set by the user in the user interface. In the CreateRangeFormat project, the code is generated in the method CreateRandeFormat.SasCode in the file CreateRangeFormats.

STORAGE AND RETRIEVAL OF PREFERENCES

This functionality is invisible to the user, but it is very important to make it possible to incorporate the custom task in SAS Enterprise Guide projects. The custom task must be able to write its current preferences (previously set by the user in the user interface) in an xml string and load the settings in the same format. This could be a tedious task in particular with a big number of parameters. Fortunately, the Visual C# project template available for download contains a template on how to generate and read from xml documents efficiently. In the CreateRangeFormat project, the parameter storage functionality is in the method CreateRandeFormat.WriteXML and the parameter load is implemented in CreateRandeFormat.ReadXML.

CONCLUSION

The presented custom task can be used in SAS Enterprise Guide to easily create formats for multiple variables. Its simplicity together with flexibility when more advanced options are utilized makes it usable for a wide range of users. If you are a SAS programmer interested in modifying the task functionality you can refer to the SAS implementation section for reference on the algorithms used. C# Programmers interested in building custom tasks for SAS Enterprise Guide might also find the Visual C# project source code useful. It can be downloaded from http://www4.ncsu.edu/~phustak/CreateRangeFormatProject.zip.

RECOMMENDED READING

Excellent documentation on how to build custom task including easy to use Visual C# template and sample projects can be found on the SAS website: http://support.sas.com/documentation/onlinedoc/guide/release30/addins/

CREATERANGEFORMAT TASK RESOURCES:

Compiled task library: http://www4.ncsu.edu/~phustak/CreateRangeFormatAddIn.dll
Task macros: http://www4.ncsu.edu/~phustak/CreateRangeFormatMacros.sas
Task Visual C# 2003 project file: http://www4.ncsu.edu/~phustak/CreateRangeFormatProject.zip

CONTACT INFORMATION

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

Petr Husták
Institute of Advanced Analytics, North Carolina State University
1430-202 Collegeview Avenue
Raleigh, NC 27606
E-mail: phustak@gmail.com