The %MktDups Macro

Introduction

The %MktDups autocall macro detects duplicate choice sets and duplicate alternatives within generic choice sets. To illustrate, consider a simple experiment that uses the following two choice sets. These choice sets are completely different and are not duplicates.

```
a b c       a b c
1 2 1       1 1 1
2 1 2       2 2 2
1 1 2       2 2 1
2 1 1       1 2 2
```

Now consider the following two choice sets:

```
a b c       a b c
1 2 1       2 1 2
2 1 2       1 1 2
1 1 2       2 1 1
2 1 1       1 2 1
```

These two choice sets are the same for a generic study because all the same alternatives are there—just in a different order. However, for a branded study they are different. For a branded study, there would be a different brand for each alternative, so the choice sets would be the same only if all the same alternatives appeared in the same order. For both a branded study and a generic study, the following choice sets are duplicates:

```
a b c       a b c
1 2 1       1 2 1
2 1 2       2 1 2
1 1 2       1 1 2
2 1 1       2 1 1
```

Now consider the following choice sets for a generic study:

```
a b c       a b c
1 2 1       1 2 1
2 1 1       1 2 1
1 1 2       1 1 2
2 1 1       2 1 1
```

First, each of these choice sets has duplicate alternatives (2 1 1 in the first and 1 2 1 in the second). Second, even though they are not exactly the same, these two choice sets are flagged as duplicates because every alternative in choice set 1 is also in choice set 2, and every alternative in choice set 2 is also in choice set 1. In generic studies, two choice sets are considered duplicates unless one choice set has one or more alternatives that are not in the other choice set.

%MktDups Macro Syntax

%MktDups(< *positional-arguments*, > **NALTS=***n* < , *optional arguments* >**)**

Required Argument

NALTS=*n*

specifies the number of alternatives. You must specify this argument for generic or branded choice designs. It is ignored for linear model designs. For generic or branded designs, the DATA= data set must contain NALTS=*n* observations for the first choice set, NALTS=*n* observations for the second choice set, and so on.

Optional Arguments

Positional Arguments

Positional arguments must be the first argument that is specified, and unlike all other arguments, they are not specified after a name and an equal sign.

BRANDED

specifies that because one of the factors is brand, the macro only needs to compare corresponding alternatives in each choice set.

GENERIC

specifies a generic design. Specifying GENERIC indicates that there are no brands and requests that the macro compare each alternative with every other alternative in every choice set. GENERIC is the default specification.

LINEAR

specifies a linear arrangement. Specify LINEAR for a full-profile conjoint design, for an ANOVA design, or for the linear version of a branded choice design.

NOPRINT

suppresses the output display.

Other Arguments

DATA=*SAS-data-set*

specifies the input choice design. By default, the macro uses the last data set that is created.

OUT=*SAS-data-set*

specifies an output data set that contains the design, with duplicate choice sets excluded. By default, no data set is created, and the macro reports only on duplicates.

OUTLIST=*SAS-data-set*
specifies the output data set that contains the list of duplicates. By default, OUTLIST=Outdups.

VARS=*variable-list*
FACTORS=*variable-list*
specifies the factors in the design. By default, all numeric variables are used.

Help Option

You can specify either of the following to display the option names and simple examples of the macro syntax:

```
%mktdups(help)
%mktdups(?)
```

%MktDups Macro Notes

This macro specifies **options nonotes** throughout most of its execution. If you want to see all the notes, submit the following statement before running the macro:

```
%let mktopts = notes;
```

To see the macro version, submit the following statement before running the macro:

```
%let mktopts = version;
```

Examples

In this first example, a design is created by using the %ChoicEff macro choice-set-swapping algorithm for a branded study. Then the %MktDups macro is invoked to check for and eliminate duplicate choice sets. The following statements create the design:

```
%mktex(3 ** 9, n=27, seed=424)

data key;
   input (Brand x1-x3) ($);
   datalines;
Acme   x1 x2 x3
Ajax   x4 x5 x6
Widgit x7 x8 x9
;

%mktroll(design=randomized, key=key, alt=brand, out=cand)

%choiceff(data=cand, model=class(brand x1-x3 / sta), seed=420,
          nsets=18, nalts=3, options=relative, beta=zero)
```

By default, the %ChoicEff macro saves the best design in a data set named Best. The results shown in Figure 1 from running PROC FREQ demonstrate that the design contains some duplicate candidate choice sets.

```
proc freq data=best;
   tables set;
run;
```

Figure 1 PROC FREQ

The FREQ Procedure

Set	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1	3	5.56	3	5.56
2	6	11.11	9	16.67
3	3	5.56	12	22.22
5	6	11.11	18	33.33
8	3	5.56	21	38.89
9	3	5.56	24	44.44
12	3	5.56	27	50.00
13	3	5.56	30	55.56
14	6	11.11	36	66.67
16	6	11.11	42	77.78
19	3	5.56	45	83.33
20	3	5.56	48	88.89
24	3	5.56	51	94.44
27	3	5.56	54	100.00

The following invocation of the %MktDups macro generates a report about the duplicate choice sets in the design that was generated by the %ChoicEff macro and saves in a SAS data set a version of the design that contains the duplicates eliminated:

```
%mktdups(branded, data=best, factors=brand x1-x3, nalts=3, out=nodups)
```

The positional argument BRANDED indicates that only corresponding alternatives in each choice set are to be compared. The DATA= argument specifies that the design to be evaluated is in the data set named Best. The FACTORS= argument specifies the factors that are in the design. The NALTS= argument specifies that each choice set contains three alternatives. The OUT= argument saves in the data set NoDups the design that contains duplicate choice sets excluded.

The %MktDups macro displays the following information in the SAS log:

```
Design:          Branded
Factors:         brand x1-x3
                 Brand
                 x1 x2 x3
Duplicate Sets:  4
```

The first line tells you that this is a branded design, not a generic design. The second line reports the factors that are specified in the FACTORS= argument. These are followed by the actual variable names for the factors. The last line reports the number of duplicates.

The %MktDups macro also generates the table in Figure 2.

Figure 2 %MktDups

+Choice+ Set=1

Choice Set	Duplicate Choice Sets To Delete
1	15

Choice Set	Duplicate Choice Sets To Delete
3	18

Choice Set	Duplicate Choice Sets To Delete
7	14

Choice Set	Duplicate Choice Sets To Delete
10	17

Figure 2 reports that choice set 1 is the same as choice set 15. Similarly, choice sets 3 and 18 are the same, and so on. The OUT=Out data set contains the design, with the duplicate choice set eliminated. The PROC FREQ results in Figure 3 show that there are no duplicate candidate choice sets in the design contained in the data set Out:

```
proc freq data=nodups;
   tables set;
run;
```

Figure 3 PROC FREQ

The FREQ Procedure

Set	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1	3	7.14	3	7.14
2	3	7.14	6	14.29
3	3	7.14	9	21.43
5	3	7.14	12	28.57
8	3	7.14	15	35.71
9	3	7.14	18	42.86
12	3	7.14	21	50.00
13	3	7.14	24	57.14
14	3	7.14	27	64.29
16	3	7.14	30	71.43
19	3	7.14	33	78.57
20	3	7.14	36	85.71
24	3	7.14	39	92.86
27	3	7.14	42	100.00

Now consider an example that has purely generic alternatives. The following statements create and evaluate the design:

```
%mktex(2 ** 5, n=2**5, seed=109)

%choiceff(data=randomized, model=class(x1-x5 / sta), seed=93,
          nsets=42, flags=4, options=relative, beta=zero)
%mktdups(generic, data=best, factors=x1-x5, nalts=4, out=out)
```

Figure 4 shows the %MktDups macro output.

Figure 4 %MktDups Output

+Choice+ Set=2

Choice Set	Duplicate Choice Sets To Delete
2	25

Choice Set	Duplicate Choice Sets To Delete	
39		Alternatives

For each choice set that is listed in the choice set column, the choice sets that it duplicates are listed or the word “Alternatives” is displayed if the problem is duplicate alternatives.

The following statements display the choice sets that have duplication problems:

```
proc print data=best;
   var x1-x5;
   id set;
   by set;
   where set in (2, 25, 39);
run;
```

Figure 5 shows the results.

Figure 5 Choice Sets with Duplicates

Set=2

Set	x1	x2	x3	x4	x5
2	1	2	1	1	1
	2	2	1	1	1
	1	1	2	2	2
	2	1	2	2	2

Set	x1	x2	x3	x4	x5
25	1	1	2	2	2
	2	1	2	2	2
	2	2	1	1	1
	1	2	1	1	1

Figure 5 *continued*

Set	x1	x2	x3	x4	x5
39	1	1	2	1	1
	1	1	2	1	1
	2	2	1	2	2
	2	2	1	2	2

The %MktDups macro detects duplicates even though the alternatives do not always appear in the same order in the different choice sets.