ABSTRACT

SAS® programmers come from a variety of backgrounds in terms of both work experience and education. Some SAS Programmers may have to produce statistical output in spite of having an educational background that does not include advanced statistics.

McNemar’s Test, performed via the FREQ procedure, is an example of a statistical procedure that may require some basic statistical knowledge in order to produce viable results. This can be accomplished with slight manipulation of the source data set that is used to produce the test, along with the use of the WEIGHT/ZEROS statement in the PROC FREQ code. The purpose of this paper is to educate SAS programmers, especially those who may not have strong backgrounds in Statistics, on the accurate, error-free and warning-free production of an output data set that contains results for the McNemar’s test.

MCNEMAR’S TEST – BACKGROUND, IN SIMPLE TERMS

This paper is not a dissertation in statistics, but rather a resource outlining how to perform a statistical procedure in SAS, directed especially to those who may not have an advanced education in statistics. Regardless, a basic background on the McNemar’s statistic helps to set the stage for the implementation of the test via the SAS programming language.

McNemar’s test is a statistical test that is based on a 2x2 classification table, and is used in situations where subjects serve as their own control. In essence, it analyzes a subject going through two different evaluations to see if the outcome of the first evaluation has any bearing on the outcome of the second. The null hypothesis asserts that the marginal probabilities for each outcome are the same. Referring to figure 1.1, this means that the null hypothesis is that the probability of A and the probability of D, denoted as P(A) and P(D), respectively, are equal. The alternative hypothesis states that they are not equal.

Figure 1.1:

<table>
<thead>
<tr>
<th></th>
<th>Test 2 = negative</th>
<th>Test 2 = positive</th>
<th>Row Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test 1 = positive</td>
<td>A</td>
<td>B</td>
<td>A+B</td>
</tr>
<tr>
<td>Test 1 = negative</td>
<td>C</td>
<td>D</td>
<td>C+D</td>
</tr>
<tr>
<td>Column Total</td>
<td>A+C</td>
<td>B+D</td>
<td>N</td>
</tr>
</tbody>
</table>

The actual McNemar’s test statistic shown in figure 1.2. Observe that A and D are the components of the denominator. This will be discussed in more detail subsequently, but based on the nature of the calculation of the statistic, SAS is unable to perform the test without an actual nonzero integer in the denominator. Naturally, any denominator(A+D) value equal to missing or equal to 0 will prevent the statistic from being calculated. Therefore, if either A or D is missing, the statistic will not be able to be calculated.

Figure 1.2:

\[ X^2 = \frac{(D-A)^2}{D+A} \]

The example below, which will be referenced throughout the duration of this paper, is based on an example provided in a Medical Statistics text by Bland (2000). The example has been modified from the original example so that the source data set could be visually displayed within the confines of this paper. In the example portrayed in figure 2.1, 96 schoolchildren were asked whether they had a severe cold while their age was 12, and again while they were age 14.
EXAMPLE (I):

Figure 2.1:

<table>
<thead>
<tr>
<th>Severe Colds at age 12</th>
<th>Severe Colds at age 14</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>6</td>
</tr>
<tr>
<td>No</td>
<td></td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td>54</td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>60</td>
</tr>
</tbody>
</table>

Observe in the table above (figure 2.1), that at age 12, 18.8% (18/96) of the schoolchildren had reported severe colds. At age 14, 37.5% (36/96) of the schoolchildren had reported severe colds.

The data used to create the table in figure 2.1 – data set SEVERECOLDS1, is shown below in figure 2.2. Variable SubjectID is a unique identification variable for a child. Cold_12_YN is equal to 1 if the child reported a cold during age 12, and 0 if he or she did not report a cold during age 12. Likewise, variable Cold_14_YN reports the same information, but for the given child at age 14. Examine the data figure 2.2, and note how it corresponds to the table above. (Note that figure 2.2 represents one data set, although it is displayed in 3 columns).
McNemar’s test is applied in this case in order to answer the following question:

*Was there a significant change in the prevalence of severe colds among schoolchildren from age 12 to age 14?*
APPLICATION

The PROC FREQ code below (figure 2.3) was created by a Statistician to produce a data set containing statistical information produced from McNemar's Test, called "ODSDATA1."

Figure 2.3:

```
proc freq data = SEVERECOLDS;
ods output McNemarsTest=ODSDATA1;
tables Cold_12_YN*Cold_14_YN / agree;
run;
```

When the code is run using the data from figure 2.2, the following data set is created (figure 2.4).

Figure 2.4:

<table>
<thead>
<tr>
<th>Table</th>
<th>Name1</th>
<th>Label1</th>
<th>cValue1</th>
<th>nValue1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table Cold_12_YN * Cold_14_YN</td>
<td><em>MCNEM</em></td>
<td>Statistic (S)</td>
<td>10.8</td>
<td>10.8</td>
</tr>
<tr>
<td>Table Cold_12_YN * Cold_14_YN</td>
<td>DF_MCNEM</td>
<td>DF</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Table Cold_12_YN * Cold_14_YN</td>
<td>P_MCNEM</td>
<td>Pr &gt; S</td>
<td>.0010</td>
<td>.001015</td>
</tr>
</tbody>
</table>

The p-value to use is located in variable nValue1 on the observation where NAME1 = ‘P_MCNEM.’, as per the Statistician’s instruction. Henceforth, the p-value is .001015. This is a relatively small p-value, which indicates that there is likely sufficient evidence to reject the null hypothesis, depending on the level of significance that is employed. Again, the null hypothesis states that there is no relationship between the proportion of severe colds among subjects at ages 12 and at 14. Rejecting the null hypothesis is analogous to stating that there is a significant difference between the two proportions.

WHERE DIFFICULTY MAY ARISE

Suppose the analysis is based on the data and table shown below in figures 3.1 and 3.2, a quasi-utopian case where nobody got colds at age 14:

EXAMPLE (II):

Figure 3.1:

<table>
<thead>
<tr>
<th>Severe Colds at age 14</th>
<th>Severe Colds at age 12</th>
<th>Yes</th>
<th>No</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td></td>
<td>30</td>
<td></td>
<td>30</td>
</tr>
<tr>
<td>No</td>
<td></td>
<td>24</td>
<td></td>
<td>24</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>54</td>
<td></td>
<td>54</td>
</tr>
</tbody>
</table>
The code in figure 3.3 fails to run, and the output data set is not created, as shown in the log (figure 3.4). Why?

Figure 3.3:

```plaintext
proc freq data= SEVERECOLDS2;
    ods output McNemarsTest=ODSDATA2 ;
    tables Cold_12_YN* Cold_14_YN / agree;
run;
```
No statistics were computed, and data set ODSDATA2 was not produced. In order for the McNemar’s test to be executed, the two squares in the contingency table, which contain ‘yes’ for one outcome and ‘no’ for the other outcome (shaded in gray above) must contain non-missing data. The denominator in the McNemar’s Test Statistic is created by finding the sum of the two yes/no squares (values used are 24 and 6 for example (I), and <missing> and 30 for example (II). If one of these values is missing, the statistic cannot be computed by SAS, since the denominator in the statistic will be <missing>.

As a SAS programmer, clearly you want to run error-free code that produces a data set with accurate results. Luckily, there is a way to circumvent the failure outlined in the above example.

**SOLUTION – CREATE A DATA SET WITH ALL POSSIBLE VARIABLE COMBINATIONS THEN USE WEIGHT/ZEROS OPTION IN PROC FREQ**

In order to solve the problem, apply the following routine outlined in figures 3.5 – 3.9. The figures below include all of the code, and the resulting data sets. If you want to copy and paste the code directly into the SAS editor, please refer to Appendix 1.

The first step is to create a data set that will include all combinations of yes/no variables that actually exist in the data. This data set will be called OUTSUMMARY, as shown in Figure 3.5.

**Figure 3.5:**
```
proc means data = SEVERECOLDS2 nway;
class Cold_12_YN Cold_14_YN;
var Cold_12_YN Cold_14_YN;
output out = OUTSUMMARY(drop = _:) N=N;
run;
```

<table>
<thead>
<tr>
<th>OUTSUMMARY</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cold_12_YN</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>1</td>
</tr>
</tbody>
</table>

Then, create a SHELL data set. This data set will contain all possible combinations of yes/no variables that could theoretically exist. Set N equal to 0 as the default for all combinations. Ultimately, N will represent the number of subjects in the study.

**Figure 3.6:**
```
DATA SHELL;
do Cold_12_YN = 0,1;
do Cold_14_YN = 0,1;
   N = 0;
   output;
end;
run;
```
Update the SHELL data set with the information from the SEVERECOLDS2 data set. Any combinations of variables that did not have any information in SEVERECOLDS2 will end up having N equal to 0. The UPDATE statement is similar to a match-merge, as shown in figure 3.7.

**Figure 3.7:**

```
data SEVERECOLDSXXX;
  update SHELL OUTSUMMARY;
  by Cold_12_YN Cold_14_YN;
run;
```

<table>
<thead>
<tr>
<th>SHELL</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Cold_12_YN</td>
<td>Cold_14_YN</td>
<td>N</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SEVERECOLDSXXX</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Cold_12_YN</td>
<td>Cold_14_YN</td>
<td>N</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>24</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>30</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

The final step is essentially the same PROC FREQ code snippet that was developed by the Statistician, as shown already, in figure 3.3. However, the magic fix is an additional line of code in the PROC FREQ code snippet below, in figure 3.8.

**Figure 3.8:**

```
weight N /zeros;.
```

This WEIGHT statement establishes that a given numeric variable will provide a weight for each observation during an analysis. If a WEIGHT statement is not specified, PROC FREQ will assign a weight of 1 to every observation. The sum of the weight variable values represents the total number of observations.

By default, PROC FREQ will ignore the observation in an analysis if a weight variable is either missing or 0. In Example(II), the weight variable is N. In order for the McNemar's test used in the example to work, the value 0 needs to be included in the analysis. This can be accomplished by means of the ZEROS option. This option is a necessity, since PROC FREQ would otherwise ignore the observation by default. Figure 3.9 displays the PROC FREQ code and the end product, a dataset containing the McNemar's test results.

**Figure 3.9:**

```
proc freq data = SEVERECOLDSXXX;
  ods output McNemarsTest = ODSDATAXXX;
  tables Cold_12_YN*Cold_14_YN/agree;
  weight N /zeros;
run;
```
The code described in figures 3.5 – 3.9 results in the ODSDATAXXX data set, which contains the appropriate McNemar’s test results, and an infinitesimal p-value.

CONCLUSION

SAS possesses a vast abundance of statistical procedures, and at times these procedures may be tricky to work with and debug if you are not a statistical expert. It can be very beneficial to understand basic concepts related to the creation of data sets before executing a statistical procedure. Having this knowledge can result in programs that are efficient, accurate, easily reusable, and error free.

REFERENCES


Special thanks to my colleagues Howard Proskin, Dan Hatch and Bill Murphy

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.

Other brand and product names are registered trademarks or trademarks of their respective companies.

CONTACT INFORMATION

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

Stephen W. Bosch
Howard M. Proskin & Associates
300 Red Creek Dr., Suite 330
Rochester, NY 14623
Phone: 585-359-2420
Fax: 585-359-0465
Email: swbosch@gmail.com
Web: www.hmproskin.com
APPENDIX 1

data SEVERECOLDS2;
    input SubjectID $ Cold_12_YN Cold_14_YN;
    datalines;
    B001 1 0
    B002 1 0
    B003 1 0
    B004 1 0
    B005 1 0
    B006 1 0
    B007 1 0
    B008 1 0
    B009 1 0
    B010 1 0
    B011 1 0
    B012 1 0
    B013 1 0
    B014 1 0
    B015 1 0
    B016 1 0
    B017 1 0
    B018 1 0
    B019 1 0
    B020 1 0
    B021 1 0
    B022 1 0
    B023 1 0
    B024 1 0
    B025 1 0
    B026 1 0
    B027 1 0
    B028 1 0
    B029 1 0
    B030 1 0
    B031 0 0
    B032 0 0
    B033 0 0
    B034 0 0
    B035 0 0
    B036 0 0
    B037 0 0
    B038 0 0
    B039 0 0
    B040 0 0
    B041 0 0
    B042 0 0
    B043 0 0
    B044 0 0
    B045 0 0
    B046 0 0
    B047 0 0
    B048 0 0
    B049 0 0
    B050 0 0
    B051 0 0
    B052 0 0
**proc freq data= SEVERECOLDS2;**
   *ods output* McNemarsTest=ODSDATA2 ;
   *tables* Cold_12_YN* Cold_14_YN / *agree*;
**run;**

**proc means data = SEVERECOLDS2 N nway;**
   *class* Cold_12_YN Cold_14_YN;
   *var* Cold_12_YN Cold_14_YN;
   *output* out = OUTSUMMARY(drop = _:) N=N;
**run;**

**DATA SHELL;**
   *do* Cold_12_YN = 0,1;
      *do* Cold_14_YN = 0,1;
       *N* = 0;
       *output*;
   *end*;
*end*;
**run;**

**DATA SEVERECOLDSXXX;**
   *update* SHELL OUTSUMMARY;
   *by* Cold_12_YN Cold_14_YN;
**run;**

**PROC FREQ data = SEVERECOLDSXXX;**
   *ods output* McNemarsTest = ODSDATAXXX;
   *tables* Cold_12_YN*Cold_14_YN/agree; 
   *weight* N /zeros;
**run;**