Lecture02 - Mostly categorical variables

Steve Simon

7/1/2019

Categorical data

- proc format
- recoding
- proc freq
- barcharts

We're going to look at a different data set, one with mostly categorical variables. I'll introduce proc format, which allows you to attach labels to categorical data, talk about recoding, and show some tables using proc freq. I'll also show you a simple bar chart.

Titanic data set

```
Name     PClass  Age Sex Survived
"Allen, Miss Elisabeth Walton"  1st 29  female  1
"Allison, Miss Helen Loraine"   1st 2   female  0
"Allison, Mr Hudson Joshua Creighton"   1st 30
male    0
"Allison, Mrs Hudson JC (Bessie Waldo Daniels)"
1st 25  female  0
"Allison, Master Hudson Trevor" 1st 0.92    male
1
"Anderson, Mr Harry"    1st 47  male    1
"Andrews, Miss Kornelia Theodosia"  1st 63
female  1
"Andrews, Mr Thomas, jr"    1st 39  male    0
"Appleton, Mrs Edward Dale (Charlotte Lamson)"
1st 58  female  1
```

Here are the first ten rows of the Titanic data set.

1. Output and data locations

```
ods pdf
  file="lecture02.pdf";

filename raw_data
  "../data/titanic_v00.txt";

libname intro
  "../data";
```

This is the first few lines of SAS code, showing where to store the output, where to find the input and where to store the SAS binary data set the program creates.

2. Reading, proc import

```
proc import
    datafile=raw_data
    out=intro.titanic
    dbms=dlm
    replace;
  delimiter='09'x;
  getnames=yes;
run;
```

As a general rule, proc import works best for simple delimited files where the first row contains the variable names. With complicated text files (such as files where the data for an individual extends across more than one line) or files without variable names in the first row are usually better handled by a data step.

3. First ten lines, proc print

```
proc print
    data=intro.titanic(obs=10);
  title1 " ";
run;
```

If you look at the first few rows of data, you will see that the import went reasonably well. It is not always this easy. Do take notice that age is left justified. It is caused by a number of “NA” codes for missing values. You don’t see it here, but if you print a few more observations, you can see the “NA” values. It would have been easier to anticipate these ahead of time, but We’ll fix things up after the fact.

First ten rows of the Titanic data set

Obs	Name	PClass	Age	Sex	Survived
1	Allen, Miss Elisabeth Walton	1st	29	female	1
2	Allison, Miss Helen Loraine	1st	2	female	0
3	Allison, Mr Hudson Joshua Creighton	1st	30	male	0
4	Allison, Mrs Hudson JC (Bessie Waldo Daniels)	1st	25	female	0
5	Allison, Master Hudson Trevor	1st	0.92	male	1
6	Anderson, Mr Harry	1st	47	male	1
7	Andrews, Miss Kornelia Theodosia	1st	63	female	1
8	Andrews, Mr Thomas, jr	1st	39	male	0
9	Appleton, Mrs Edward Dale (Charlotte Lamson)	1st	58	female	1
10	Artagaveytia, Mr Ramon	1st	71	male	0

Output, proc print

At first glance, everything looks fine. But if you look closely, you will see that age is left justified. It is caused by the NA code for missing value, which doesn't appear until about line 14 or 15 of the code.

4. Counts, proc freq

```
proc freq
    data=intro.titanic;
  tables PClass Sex Survived;
run;
```

For any categorical variables, your first step is to get frequency counts.

Counts for categorical data (1/2)

2

The FREQ Procedure

PClass	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1st	322	24.52	322	24.52
2nd	280	21.33	602	45.85
3rd	711	54.15	1313	100.00

Sex	Frequency	Percent	Cumulative Frequency	Cumulative Percent
female	462	35.19	462	35.19
male	851	64.81	1313	100.00

Output, proc freq

Here are the counts for passenger class and sex.

Counts for categorical data (2/2)

The FREQ Procedure

Survived	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	863	65.73	863	65.73
1	450	34.27	1313	100.00

Output, proc freq

Here are the counts for Survived.

5. Convert string to numeric, data step

```
data intro.titanic;
  set intro.titanic;
  age_c = input(age, ?? 8.);
run;

proc means
    n nmiss mean std min max
    data=intro.titanic;
  var age_c;
run;
```

For the one continuous variable (age) you need to convert the code “NA” to the SAS missing value code, which is a dot. The easiest way to do this is to force the data to numeric with a simple arithmetic equation like adding a zero. But you get a warning message for each occurence of NA, which can get tedious. The input function with two question marks avoids this issue.

Means and standard deviations for age

4

The MEANS Procedure

Analysis Variable : age_c					
N	N Miss	Mean	Std Dev	Minimum	Maximum
756	557	30.3979894	14.2590487	0.1700000	71.0000000

Output, proc freq

Here are the descriptive statistics for age. Notice the number of missing values.

6. Using proc format to code categorical data

```
proc format;
  value f_survived
    0 = "No"
    1 = "Yes";
run;

proc freq
    data=intro.titanic;
  tables Survived;
  format Survived f_survived.;
run;
```

For variables like Survived which are numbers, but the numbers represent a particular category, you can document this using a format statement.

Nicely formatted counts for survival

5

The FREQ Procedure

Survived	Frequency	Percent	Cumulative Frequency	Cumulative Percent
No	863	65.73	863	65.73
Yes	450	34.27	1313	100.00

Output, proc freq

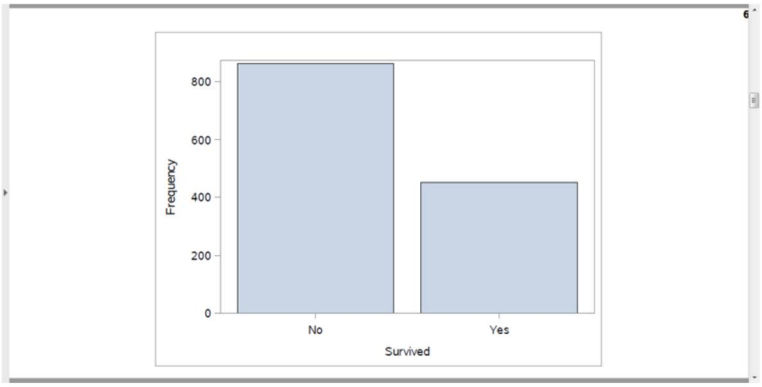
Notice that the format statement replaces the cryptic 0-1 code with the words no and yes.

7. Bar charts, proc sgplot

```
proc sgplot
    data=intro.titanic;
  vbar Survived;
  format Survived f_survived.;
run;
```

I don’t normally like bar charts, but they do have their uses.

Bar chart



Output, proc sgplot

Here are the descriptive statistics for age. Notice the number of missing values.

8. Percentages for bar chart

```
proc freq
    data=intro.titanic;
  tables Survived / noprint out=pct_survived;
run;

proc print
    data=pct_survived;
  format Survived f_survived.;
run;

proc sgplot
    data=pct_survived;
  vbar Survived / response=Percent;
  yaxis max=100;
  format Survived f_survived.;
run;
```

Getting percentages is a bit tricky. You have to run proc freq and output the results to a new data file, pct_survived. I am using the noprint option, because I only want the percentages for internal use. It wouldn't have hurt anything to print out a bit extra, but I want to encourage you to limit the amount of output that you present to a consulting client.

Note the yaxis maxx=100 statement which expands the upper limit of the y axis to 100%.

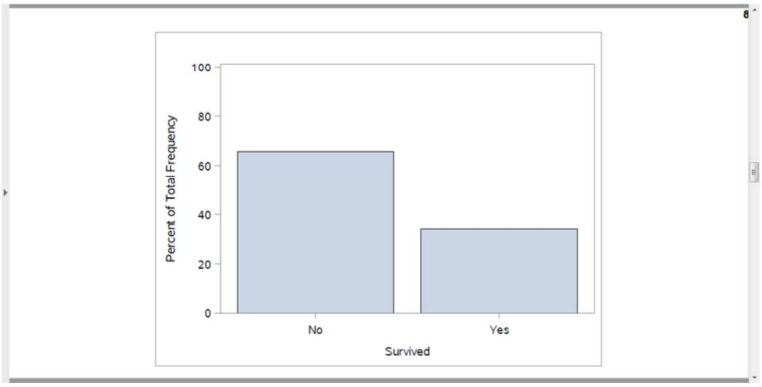
Percentages, proc freq

Obs	Survived	COUNT	PERCENT
1	No	863	65.7273
2	Yes	450	34.2727

Output, proc freq

Here is what the output from proc freq looks like. Just two rows.

Percentages in a bar chart



Output, proc sgplot

Here is what the output from proc freq looks like. Just two rows.

9. Crosstabulation

```
proc freq
    data=intro.titanic;
  tables Sex*Survived / nocol nopercent;
  format Survived f_survived.;
run;
```

To examine relationships among categorical variables use a two dimensional crosstabulation.

Percentages, proc freq

The FREQ Procedure

Frequency Row Pct	Table of Sex by Survived		
	Survived		
Sex	No	Yes	Total
female	154 33.33	308 66.67	462
male	709 83.31	142 16.69	851
Total	863	450	1313

Output, proc freq

Here is what the output from proc freq looks like. Among the males, almost 5/6 died. Among the females only 1/3 died.

10. Converting a continuous variable to categorical

```
data age_categories;
  set intro.titanic;
  if age_c = .
    then age_cat = "missing ";
  else if age_c < 6
    then age_cat = "toddler ";
  else if age_c < 13
    then age_cat = "pre-teen";
  else if age_c < 21
    then age_cat = "teenager";
  else age_cat   = "adult   ";
run;
```

If you want to create categories from a continuous variable, use a series of

if - then - else

statements

11. Quality check

```
proc sort
    data=age_categories;
  by age_cat;
run;

proc means
    min max
    data=age_categories;
  by age_cat;
  var age_c;
run;
```

Always cross check your results against the original variable.

Recoding age (1 / 3)

10

The MEANS Procedure

age_cat=adult

Analysis Variable : age_c	
Minimum	Maximum
21.0000000	71.0000000

age_cat=missing

Analysis Variable : age_c	
Minimum	Maximum
.	.

Output, proc means

Here is the quality check. Notice that adult starts at 21. Should adult start at 18 instead?

Recoding age (2 / 3)

11

The MEANS Procedure

age_cat=pre-teen

Analysis Variable : age_c	
Minimum	Maximum
6.0000000	12.0000000

age_cat=teenager

Analysis Variable : age_c	
Minimum	Maximum
13.0000000	20.0000000

Output, proc means

Are the ranges for pre-teen and teenager reasonable?

Recoding age (3 / 3)

12

The MEANS Procedure

age_cat=toddler

Analysis Variable : age_c	
Minimum	Maximum
0.1700000	5.0000000

Output, proc means

How about the ranges for toddler?

12. Controlling the display order

```
data age_codes;
  set intro.titanic;
  if age_c = .
    then age_cat = 9;
  else if age_c < 6
    then age_cat = 1;
  else if age_c < 13
    then age_cat = 2;
  else if age_c < 21
    then age_cat = 3;
  else age_cat = 4;
run;
```

Notice that the order for age_cat is alphabetical, which is probably not what you want. You can control the order by using number codes and formats.

13. With number codes, use proc format

```
proc format;
  value f_age
    1 = "toddler"
    2 = "pre-teen"
    3 = "teenager"
    4 = "adult"
    9 = "unknown";
run;
```

Once you have the number codes, assign an interpretable label using proc format.

14. Quality check

```
proc sort
    data=age_codes;
  by age_cat;
run;

proc means
    min max
    data=age_codes;
  by age_cat;
  var age_c;
  format age_cat f_age.;
run;
```

Here's the quality check again.

Better age recode (1 /3)

13

The MEANS Procedure

age_cat=toddler

Analysis Variable : age_c	
Minimum	Maximum
0.1700000	5.0000000

age_cat=pre-teen

Analysis Variable : age_c	
Minimum	Maximum
6.0000000	12.0000000

Output, proc means

This shows the age categories starting at the youngest: toddler and pre-teen…

Better age recode (2 /3)

14

The MEANS Procedure

age_cat=teenager

Analysis Variable : age_c	
Minimum	Maximum
13.0000000	20.0000000

age_cat=adult

Analysis Variable : age_c	
Minimum	Maximum
21.0000000	71.0000000

Output, proc means

followed y teenager and adult…

Better age recode (3 /3)

15

The MEANS Procedure

age_cat=unknown

Analysis Variable : age_c	
Minimum	Maximum
.	.

Output, proc means

with missing bring up the rear. This order was the order of the number codes. So if you want to display your results in a non-alphabetical order, use number codes.

15. Modifying a categorical variable

```
data first_class;
  set intro.titanic;
  if PClass = "1st"
    then first_class = "Yes";
    else first_class = "No";
run;

proc freq
    data=first_class;
  table PClass*first_class /
    norow nocol nopercent;
run;
```

Here's another example where you compare First Class passengers to Second and Third class passengers combined.

Quality check

16

The FREQ Procedure

Frequency	Table of PClass by first_class		
	first_class		
PClass	No	Yes	Total
1st	0	322	322
2nd	280	0	280
3rd	711	0	711
Total	991	322	1313

Output, proc means

Here is the quality check. PClass=1st codes to first_class=Yes. PClass=2nd or 3rd codes to first_class=No.