Stata Recitation - Week 8 - Chi-squared tests and Labels

McCourt School of Public Policy, Georgetown University

Key Ideas:

- Calculate statistics based on two variables: tab2 (chi2 option), corr, pwcorr
- Create and modify variable and value labels
- Search variable labels

REVIEW PROBLEMS using the `citytemp.dta` example dataset

```
sysuse citytemp.dta, clear
```

- Generate a dummy variable, hightempJan, that is 1 for all cities with average January temperatures above 40 degrees and 0 for all those with average temperatures of 40 degrees or less.

```
gen hightempjan = 0
replace hightempjan = 1 if tempjan > 40
replace hightempjan = . if tempjan == .
```

- How many cities have high January temperatures?

```
tab hightempjan
```

Ans: 353 cities

- Create a bar graph showing mean cooling degree days and mean heating degree days, broken into two categories, for cities with high January temperatures and low January temperatures.

  ```
  graph bar (mean) heatdd (mean) cooldd, over(hightempjan)
  ```

- How many cities are in the "West" Census region and the "Mountain" Census division?

```
tab region
tab region, nolabel
tab division
tab division, nolabel
count  if region == 4 & division == 8
```

Ans: 61 cities in both the "West" Census region and the "Mountain" Census division

Discrete/Categorical variables : Two-way tabulation and Pearson's chi-2 test

Two way tabulation

Produce counts of number of observations in each cell of a two-way table.

Examples

```
clear
sysuse nlsw88.dta
tab2 race married
```

- First variable goes in rows, second variable goes in columns.
- This is important when you have a variable with many categories:

  ```
  tab2 age married
  tab2 married age
  ```

- Many different options with tab:

  ```
  help tab2
  ```

- We've seen the missing option with one-way tabulations:

  ```
  tab2 union married
  tab2 union married , m
  ```

- Look at total number of observations for these two tables.
- Compare with Obs number from summarize:

  ```
  sum married union
  ```

Important

- We are moving into commands that take data from multiple variables
- If an observation has missing data for any of the variables, that observation is dropped from the calculation.
- With some commands, like `tab2`, we can avoid that behavior. But that will not be possible for other commands.

More options for tab2

- column : Gives percentage breakdown of row category within each column.

```
tab2 race union, column
tab2 race if union==0
tab2 race if union==1
```

- row : Gives percentage breakdown of column category within each row.

```
tab2 race union, row
tab2 union if race==1
tab2 union if race==2
tab2 union if race==3
```

- cell : Gives percentage of observations in each cell.

```
tab2 race union, cell
```

- expected : Gives the expected number of observations in each cell based on marginal distributions of each variable

```
tab2 race union, row column
```

- expected number of observations in the white, nonunion cell:

```
display 0.7204*0.7545*1878
tab2 race union, expected
```

Report Chi2 test statistic:

- Test for independence of two categorical variables:
- This will be covered in class, but you should know how to calculate and find test statistic.

```
tab2 race union, chi2
```

- test statistic: Pearson chi2(2) = 13.0814
- P-value: Pr = 0.001
- Components of chi2 test statistic can be reported using option: `cchi2`
- Students should read through these options after learning about the Chi-2 test in class.

Labels

- Three types of labels, data set, variable, and values

```
help label
```

Variable Labels

- Show up in variable window
- Show command syntax in help file
- Example from previous recitation: "‘ clear sysuse nlsw88.dta

gen weekwage = wage*hours label variable weekwage "Ave. Weekly Pay"

```
- Changes can also be made in the Variables Manager `Data > Variables Manager`
- Remember to put the resulting "label" command into your do-file.
```

generate agesq = age^2

```
- Use the varibles manager to label, then add label to do-file.
```

label variable agesq "Age Squared"

```
- A very useful function when you start working with large data sets
- `lookfor`: searches variable names and labels
```

lookfor age

```
### Data Set Labels

- Data set label is similar to variable label, but applies to entire data set.
- Show syntax in help file
- Data labels show up when a data set is opened and in the describe command.
- Useful when you have to save a modified version of your data.
```

- Label and save modified data label data "Modified data set from recitation 6" save "nlsw88 - recitation 6.dta"
- Reopen data to demonstrate data label clear use "nlsw88 - recitation 6.dta"
- Data label can also be seen with describe describe, short "'

Value Labels

- Value labels are more complicated than data or variable labels
- Value labels are defined and exist independently of variables
- Show value labels using describe and labelbook
- Individual labels can be listed:

  ```
  label list occlbl
  ```

Labeling values is a two-step process

1. define label
2. apply label to variable

Example from problem set last week:

```
* Create an indicator called tenure20 for people with 20 or more years tenure.
gen tenure20=0
replace tenure20=1 if tenure>=20
replace tenure20=. if tenure==.

* Label variable
label variable tenure20 "Tenure of 20 or more years"

* Create value label
label define tenure20lbl 0 "Less than 20 years" 1 "20 or more years"
```

```
* Apply value label
label values tenure20 tenure20lbl

tab tenure20
```

- Value label management can be done with the "Manage Value Labels" dialogue box:
- `Data > Data utilities > Label utilities > Manage value labels`

- Applying value labels to variables can be done in the Variable Manager
- As always, commands should be recorded in do-file
- Another example from last weeks problem set:
- Create an indicator variable called once_married, for people who were once married, but are not currently married

```
gen once_married=0
replace once_married=1 if married==0 & never_married==0
replace once_married=. if married==. | never_married==.

* Label variable and values
label variable once_married "Once married, but not currently married"
label define once_marriedlbl 1 "Once married" 0 "Never or currently married"
label values once_married once_marriedlbl
```

Encode/decode

- Changing between strings and labels
- A categorical variable exists as a string and needs to be changed to a number.
- Or the other way around.

```
help encode
```

- See examples in help file and manual.