

Dataset Manipulation

Will Doyle

2020-09-16

Introduction

Learning to manipulate datasets is a key skill for statistical analysis. Today we'll work on four skills: subsetting data using preserve and restore commands, appending data, doing a simple one-to-one merge, and collapsing data.

```
. capture log close                // closes any logs, should they be open
. log using "dataset_manipulation.log", replace // open new log
-----
      name: <unnamed>
      log:  /Users/doylewr/lpo_prac/lessons/sl-04-dataset_manipulation/dataset_manipulation.log
      log type: text
      opened on: 16 Sep 2020, 10:28:46

. clear all                        // clear memory

. import delimited "https://stats.idre.ucla.edu/wp-content/uploads/2016/02/hsb2-2.csv", clear
(11 vars, 200 obs)

. import excel "tabn304.10.xls", cellrange(A5:L64) clear
(12 vars, 60 obs)

. global urldata "https://stats.idre.ucla.edu/stat/stata/seminars/svy_stata_intro/apipop"

. use $urldata, clear
```

Subsetting with preserve and restore

A feature and sometimes curse of Stata is that it can only hold one dataset in active memory at a time. As a feature, it helps you keep your work organized and, through numerous warning messages, tries to make sure you don't lose your work by accidentally forgetting to save or mindlessly overwriting your data. The feature feels more like a curse when you have multiple datasets that you would like to work with simultaneously or, as we will do below, split a single dataset into smaller parts.

To repeatedly subset a large dataset, there are two primary choices: 1. Reload the full dataset into memory after each subset and save 2. Use the preserve and restore commands

In the code below, notice how the preserve and restore commands bookend the keep command, which keeps only those observations that fulfill the if statement (in this case, the type of school). The steps are : 1. preserve dataset in memory 2. subset to keep only school type that we want 3. save new subset dataset 4. restore old dataset

```
. preserve

. keep if stype == 1
(1,773 observations deleted)

. tab stype

      stype |      Freq.      Percent      Cum.
-----+-----
          E |         4,421         100.00         100.00
-----+-----
        Total |         4,421         100.00

. save elem, replace
file elem.dta saved

. restore

. preserve

. keep if stype == 2
(5,439 observations deleted)

. save hs, replace
file hs.dta saved

. restore

. keep if stype == 3
(5,176 observations deleted)

. save middle, replace
file middle.dta saved
```

Appending Data

Appending data is done when we want to add additional observations to an existing dataset, using a dataset that has exactly the same variable names but different observations. Suppose you have data on high schools, middle schools, and elementary schools on a variety of performance indicators and you'd like to merge them together. The syntax uses, appropriately enough, the append command, which takes the format append <new dataset> (the command assumes the first dataset is the one in memory; remember that the middle school subset data are still in memory):

```
. append using elem
(label yr_rnd already defined)
(label awards already defined)
(label both already defined)
(label comp_imp already defined)
(label sch_wide already defined)
(label stype already defined)

. append using hs
(label stype already defined)
(label sch_wide already defined)
(label comp_imp already defined)
(label both already defined)
(label awards already defined)
(label yr_rnd already defined)

. use elem, clear

. append using middle
(label yr_rnd already defined)
(label awards already defined)
```

```
(label both already defined)
(label comp_imp already defined)
(label sch_wide already defined)
(label stype already defined)

. use elem, clear

. merge 1:1 snum using middle, nogen
(label stype already defined)
(label sch_wide already defined)
(label comp_imp already defined)
(label both already defined)
(label awards already defined)
(label yr_rnd already defined)

Result                                     # of obs.
-----
not matched                               5,439
    from master                           4,421
    from using                             1,018
matched                                   0
```

The `append` command will not copy over labels from the using dataset, so you'll need to make sure they're right in the master dataset. The most common error with an `append` command is to not have exactly matching variable names.

Merging Data

You can also use Stata's `merge` command to do an `append` operation in special cases. This happens when the merging variable doesn't have repeated *observations* in the two datasets, which in turn have exactly the same variable structure. Think of a Venn diagram where the circles contain exactly the same types of information, but don't overlap; in combining them, we've really just grown them into one bigger circle. One of the virtues of using `merge` when `append` will suffice is that you have access to more information about where the data came from once you're done.

```
. use elem, clear

. merge 1:1 snum using hs, gen(_merge_a)
(label yr_rnd already defined)
(label awards already defined)
(label both already defined)
(label comp_imp already defined)
(label sch_wide already defined)
(label stype already defined)

Result                                     # of obs.
-----
not matched                               5,176
    from master                           4,421 (_merge_a==1)
    from using                             755 (_merge_a==2)
matched                                   0 (_merge_a==3)

. merge 1:1 snum using middle, gen(_merge_b)
(label stype already defined)
(label sch_wide already defined)
(label comp_imp already defined)
(label both already defined)
(label awards already defined)
(label yr_rnd already defined)

Result                                     # of obs.
-----
not matched                               6,194
    from master                           5,176 (_merge_b==1)
    from using                             1,018 (_merge_b==2)
matched                                   0 (_merge_b==3)
```

Once you've completed the merge, you can take a look at the `_merge_*` variables that were generated to see where the data came from.

```
. tab _merge_a
      _merge_a |      Freq.      Percent      Cum.
-----+-----
      master only (1) |      4,421      85.41      85.41
      using only (2) |       755      14.59     100.00
      Total |      5,176     100.00

. tab _merge_b
      _merge_b |      Freq.      Percent      Cum.
-----+-----
      master only (1) |      5,176      83.56      83.56
      using only (2) |      1,018      16.44     100.00
      Total |      6,194     100.00

. use $urldata, clear

. preserve

. keep snum api00 api99 ell meals          // variable set 1

. save api_1, replace
file api_1.dta saved

. restore

. keep snum full emer                      // variable set 2

. save api_2, replace
file api_2.dta saved
```

One-to-one merges

A one-to-one merge is when you have exactly the same *observations* but new variables to add to the dataset. Say you have *observations* with variables split across datasets, e.g., School 1 has variables A, B, and C in dataset 1 and variables X, Y, and Z in dataset two. As long as School 1 has a unique identifier--a name, an id number, etc--you can `merge` these two datasets together so that you have access to all of the school's variables for your analysis.

First, we need to subset our data again, only this time by splitting along columns (*variables*) rather than rows (*observations*):

```
. merge 1:1 snum using api_1
```

```

Result                                     # of obs.
-----
not matched                               0
matched                                6,194  (_merge==3)
-----

. tab _merge

      _merge |      Freq.      Percent      Cum.
-----+-----
      matched (3) |      6,194      100.00      100.00
-----+-----
             Total |      6,194      100.00
-----

. use $urldata, clear

```

Collapsing data

Collapsing data refers to summarizing data across a type and creating a new dataset as a result. Say we want to create a county-level dataset from our school data, using the average figures for the schools across a set of characteristics. The command would look like this:

```

. unique cnum
Number of unique values of cnum is  57
Number of records is  6194

. preserve

. collapse (mean) pcttest mobility, by (cnum)

. restore

. collapse (sum) district_enroll=enroll, by(dnum)

. save district_enroll, replace
file district_enroll.dta saved

. use $urldata, clear

. merge m:1 dnum using district_enroll

Result                                     # of obs.
-----
not matched                               0
matched                                6,194  (_merge==3)
-----

. count
6,194

. log close                                // close log
      name: <unnamed>
      log:  /Users/doylewr/lpo_prac/lessons/s1-04-dataset_manipulation/dataset_manipulation.log
      log type: text
      closed on:  16 Sep 2020, 10:28:53
-----

. exit                                    // exit script

```