Cleaning Data With Stata

Andy Grogan-Kaylor

19 Feb 2022 11:37:49

Background

It sometimes seems like 80% of our work as data analysts is cleaning the data, while only 20% is the actual analysis. Here are some Stata commands that are useful in cleaning data.

First we simulate some data to work with, and to clean.

Simulate Some Data

This section is provided for illustration only, as it may be helpful to see *how* the data was simulated, and the decisions that went into simulating the data. You may also *safely ignore* this section if you like.

Show / Hide Data Simulation Code

```
. clear all

. set obs 100 // 100 observations
Number of observations (_N) was 0, now 100.

. generate id = _n // random id

. generate age = rnormal(50,10) // random generated age

. replace age = 200 in 1 // someone is 200 years old!
(1 real change made)

. generate happy = runiformint(1,5) // randomly generated happiness

. replace happy = 999 in 10 // simulate a missing value
(1 real change made)

. generate somethingelse = rnormal(0, 1) // something else!
```

Look At Some Of The Data

```
. list in 1/10 // list first 10 observations
```

	id	age	happy	somethi_e
1.	1	200	5	-.911319
2.	2	73.38687	4	-.5628491

```
  3.  │  3   52.8767     2  -.9555836
  4.  │  4  54.27218     2  -.0608678
  5.  │  5  51.55188     4   .0179151
      ├─────────────────────────────────
  6.  │  6   32.8428     5  -1.565369
  7.  │  7  30.44114     4  -.2971638
  8.  │  8  45.72283     3   .7488565
  9.  │  9  45.93948     4   1.127129
 10.  │ 10  33.88575   999    .814114
```

Clean The Data!

Look at The Data and Think About The Data (`describe, summarize, tabulate, codebook, browse`)

When we look at variables we are looking for values that don't make sense, or that are outside the plausible range. As we are working with the data, it may sometimes be helpful to `browse` the data.

```
. describe // describe the data

Contains data
 Observations:          100
    Variables:            4
```

Variable name	Storage type	Display format	Value label	Variable label
id	float	%9.0g		
age	float	%9.0g		
happy	float	%9.0g		
somethingelse	float	%9.0g		

```
Sorted by:
     Note: Dataset has changed since last saved.

. summarize // descriptive statistics
```

Variable	Obs	Mean	Std. dev.	Min	Max
id	100	50.5	29.01149	1	100
age	100	52.60221	17.39019	30.44114	200
happy	100	13.03	99.60259	1	999
somethinge_e	100	-.0508028	.9586304	-2.228811	2.444018

```
. tabulate happy // tabulation of this particular categorical variable
```

happy	Freq.	Percent	Cum.
1	17	17.00	17.00
2	21	21.00	38.00
3	19	19.00	57.00
4	22	22.00	79.00
5	20	20.00	99.00
999	1	1.00	100.00
Total	100	100.00	

```
. codebook happy // VERY detailed view of this particular categorical variable

─────────────────────────────────────────────────────────────────────────────
happy
─────────────────────────────────────────────────────────────────────────────

                  Type: Numeric (float)

                 Range: [1,999]                    Units: 1
         Unique values: 6                      Missing .: 0/100
```

2

```
Tabulation: Freq.  Value
           17   1
           21   2
           19   3
           22   4
           20   5
            1   999
```

Notice that. . .

- There are variables in which we may not have interest.
- None of the variables are labelled informatively.
- Variables do not seem to have informative value labels.
- Someone appears to 200 years old.
- There appear to be missing values in the variable `happy` that need to be recoded.

  Remember that the command `lookfor` is often very helpful in *looking for* a particular variable. e.g. `lookfor happy`.

Only `keep` The Variables Of Interest

We may only be interested in keeping some variables to keep our analytic data set more manageable.

For this particular analysis we may wish to drop the variable called `somethingelse`.

```
. keep id age happy // keep only relevant variables
```

We could also have said `drop somethingelse`.

Add *Variable* Labels (`label variable "..."`)

```
. label variable id "ID" // label variable

. label variable age "Age in Years" // label variable

. label variable happy "Happiness Scale" // label variable
```

Create *Value* Labels (`label define ...`)

```
. label define happy 1 "Rarely" 2 "Sometimes" 3 "Often" 4 "Always" // create value label
```

Attach *Value* Labels To *Variables* (`label values ...`)

Variables and *value labels* can have the same names but are different things. We add the variable label `happy` to the variable named `happy`.

```
. label values happy happy // attach VALUE LABEL happy to VARIABLE happy
```

Recode Outliers, Values That Are Errors, Or Values That Should Be Coded As Missing (`recode`)

```
. recode happy (999 = .) // recode values as missing
(1 changes made to happy)

. recode age (100/max = 100) // age is topcoded at 100 (may or may not be plausible)
(1 changes made to age)
```

3

We `describe` and `summarize` The Data And See The Changes That Have Been Made

```
. describe

Contains data
 Observations:            100
    Variables:              3
─────────────────────────────────────────────────────────────────────────────
Variable      Storage   Display    Value
    name         type    format    label      Variable label
─────────────────────────────────────────────────────────────────────────────
id             float    %9.0g                 ID
age            float    %9.0g                 Age in Years
happy          float    %9.0g      happy      Happiness Scale
─────────────────────────────────────────────────────────────────────────────
Sorted by:
     Note: Dataset has changed since last saved.

. summarize
    Variable │        Obs        Mean    Std. dev.        Min        Max
─────────────┼───────────────────────────────────────────────────────────
          id │        100        50.5    29.01149          1        100
         age │        100    51.60221    10.22963   30.44114        100
       happy │         99    3.070707    1.394249          1          5
```