Stata Summer Series

Stata 201 – Data Cleaning

Our introductory Stata sessions #100 and #101 laid out the fundamentals of interacting with Stata; setting up DO and LOG files to easily save, replicate, modify, and share your analysis and its results; and how to detect and deal with missing and special values as you check your work.

Stata 201 will step things up a bit, so do not worry if things are slightly confusing or a bit more difficult to follow along on. We will be around to answer questions and debug during and after the session. Up to now we have been using pristine, pre-cleaned data, the goal of this session is to lay out the basics of importing, cleaning, and manipulating messy data that you may encounter in the real world from administrative, survey, or your own data collection.

What you will get out of this session:

- Learn how to utilize messy, real-world data
- Find and deal with duplicate records
- Convert string to numeric and date data or vice versa
- Convert information stored in text strings into indicators

Basic command structure

```
command   objects    conditions          , options
use       file.dta                       , clear
generate  age = 15   if AGE2 == 15
tabulate  state      if country == "US"  , missing
```

Helpful resources

- Stata manual: access by typing "help command" in the stata console
- Statalist: https://www.statalist.org/forums/forum/general-stata-discussion/general
  - Often will come up if you google a question that isn't covered by the documentation
- UCLA IDRE: https://stats.idre.ucla.edu/stata/
  - Provides helpful tips on how to use Stata as well as the statistics behind the programming
- UNC CPC: http://www.cpc.unc.edu/research/tools/data_analysis/statatutorial
  - Guide to working with and analyzing data in Stata

Remember: Getting errors is a normal part of programming! The best way to debug is to read through every line carefully.

Next classes:

- Stata 202 – **Data Manipulation** (Wednesday, July 31, 2:00 pm-3:00 pm in Rm. 3030)
- Stata 301 – **Automating Tasks and Exporting Output** (Wednesday, August 7, 2:00 pm-3:00 pm in Rm. 3030)

```
clear
capture log close
capture ssc install savesome                /*Some commands have to be installed, but
                                              only once, capture prevents the error from
                                              popping up*/

set more off, permanently

*Set a working directory where all your project files will be located (replace my username
with your own):
*cd "D:\Users\CLou\Desktop\"
cd "C:\Users\urbanmeet\Desktop\"


log using "StataClass_$S_DATE.log", replace         /*$S_DATE saves the date in
                                                      the name of your log file, so
                                                      you save a log daily. This is
                                                      helpful if you're changing
                                                      things and need to go back*/


/*Easiest way to import a file is to use the pull down menu, it will give you the stata
code for the command, otherwise you can use StatTransfer*/

/*Importing an excel file*/
import excel "Stata Class File_long.xlsx",  /// Imports the excel file
            sheet("Sheet1") /// Tells stata which sheet to import
            firstrow        /// Tells stata first row is variable names
            case(lower)     /// Makes variable names lowercase (Stata is case sensitive)
            allstring       /// Forces all to be string, some numeric won't import well
            clear

describe
list in 1/10                /* Lists the first 10 observations*/

*DUPLICATES
/*Checking for duplicates, this should includ one observation per person per arrest date*/
duplicates report id            /*more than one arrest for most of these individuals*/
duplicates report id arrdate    /*reports number of duplicates on id arrdate*/
duplicates tag id arrdate, gen(t_dup) /*generates a variable called t_dup to identify
                                        duplicates*/
list if t_dup!=0
duplicates drop                 /*drops any true duplicates*/
duplicates report id arrdate    /*reports number of duplicates on id arrdate*/


/*Converting from string to numeric and vice versa*/
list id arrdate in 1/10             /*displays first 10 observations of id and arrest date*/
describe id arrdate                 /*describes the characteristics of id and arrestdate*/

destring id, gen(t_id)              /*Turns string into numeric, need all characters
                                      to be numeric. Having temporary variables start
                                      with t_ helps you keep track of which variables
                                      to drop*/
destring arrdate, gen(t_arrdate)    /*This doesn't work because arrdate has "/"s
                                      in it.*/
list id arrdate in 1/10             /*displays first 10 observations of id and arrest date*/
describe id arrdate                 /*describes the characteristics of id and arrestdate*/

tostring t_id, replace              /*Turns numeric back to string*/
drop t_id

/*Cleaning Dates*/
list arrdate in 1/10
gen arr_d=date(arrdate, "MDY")          /*Need to input format of the date of birth,
                                          "MDY" (Month Day Year) or
                                          "DMY" (Day Month Year)*/
    label var arr_d "Arrest Date"       /*Adds a label to the variable*/
    list arr_d arrdate in 1/10
    format arr_d %td                    /*dates in stata are numeric values that are
```

```
                                        the number of days (or other denomination)
                                        since 1/1/1960, they need to be formatted*/
    list arr_d arrdate in 1/10
    assert arr_d!=. if arrdate!=""      /*Assert checks the statement, if it's not true
                                        then it will stop your program. This is helpful
                                        if you're running the same program on different
                                        data it checks your assumptions*/
    drop arrdate

/*Once it's in stata date format it's easy to transform*/
    gen arr_m=month(arr_d)              /*generates month of arrest variable*/
    gen arr_y=year(arr_d)               /*generates year of arrest variable*/
    gen arr_my=mofd(arr_d)              /*generates variable equal to the month/year of
                                        arrest, note numerically this is the
                                        number of months since 1/1/1960, need to
                                        format*/
    list arr_m arr_y arr_my arr_d in 1/10
    format arr_my %tm                   /*formats it as year month*/
    list arr_m arr_y arr_my arr_d in 1/10
    gen arr_jan25=(arr_d==td(25jan2014)) /*Shortcut for creating dummies is to put
                                        the if statement after the equal sign in
                                        parenthesis, downside is only for zero/one
                                        variables, problem for missings*/
                                        /*Note td() calls the label otherwise
                                        you would have to calculate days from
                                        1/1/1960*/
    tab arr_d if arr_jan25==1           /*Check it is defined correctly*/
    drop arr_jan25 arr_my arr_m
/*Clean arrest reason*/
tab violdesc, m
    replace violdesc=lower(violdesc) /*makes all lower case, opposite is upper()*/
    replace violdesc=strtrim(violdesc) /*drops excess spaces at the beginning and end of a
string*/
    tab violdesc, m

/*Say we wanted to create a variable for arrest reason*/
gen arrestreason=0
    replace arrestreason=1 if strpos(violdesc, "theft")!=0 /*strpos tells you the
                                                           position of the string
                                                           in the variable, 0 means
                                                           it's not in the variable*/
    replace arrestreason=2 if (strpos(violdesc, "liquor")!=0 | strpos(violdesc, "alcohol")!=
0)
    tab violdesc arrestreason, m
    label define arrestreason 0 "Other" 1 "Theft" 2 "Alcohol"
    label values arrestreason arrestreason
    label var arrestreason "Reason For Arrest"
    tab violdesc arrestreason, m

/*Other useful string commands: */
    tab violdesc
    gen t_violdesc = subinstr(violdesc, "-", " ",.) /*pull out certain characters*/
    tab t_violdesc, m
    split violdesc, p("-") gen(t_)                  /*Divides the string into a bunch
                                                    of strings based on where the parse
                                                    is, useful for full names*/
    help string functions                           // Stata HELP file listing various
string functions...

    list violdesc t_1 t_2 if t_2!="" in 1/100
    drop t_*            /* a * after a variable name calls all variables that start
                        with that name, use a t_ prefix  for all your temporary
                        variable and then you can drop with a drop t_*
                        when your done*/
/*renaming variables*/
rename arrestreason arr_r
    describe
rename arr_*    arrest_*    /*You can use * to rename a bunch of variable */
    describe
rename *_d  d_*             /*or to add/remove a prefix*/
```

```
137         describe
138     rename d_*  *_d
139     rename arrest_* arr_*
140         describe
141
142     *Right now, the data is setup in LONG format, meaning there are repeated
        observations/rows/records
143         *of the same entity/unit of analysis, usually across time. Sometimes it is easier
144         *or better to work with data in WIDE format, where there is one observation/row/record
145         *for each entity or unit of analysis, as you do not have to worry about double counting.
146         *The code below will help with converting from LONG to WIDE format; he code is similar
        for WIDE to LONG.
147
148     /*Say we want to get rid of duplicates so that a person has their arrest reasons
149     listed horizontally*/
150     duplicates tag id arr_d, gen(t_dup) /*generates a variable called t_dup to identify
151                                         duplicates*/
152
153     list if t_dup!=0
154
155     *This code will create indicators for if each individual (ID) was ever arrested for each of
        the 3 reasons we specified on each date:
156     gen t_arr=(arr_r==1)
157     list id arr_d arr_r t_arr in 1/10
158     by id arr_d, sort: egen arr_theft=max(t_arr)
159     list id arr_d arr_r t_arr arr_theft in 1/10
160     drop t_arr
161     gen t_arr=(arr_r==2)
162     by id arr_d, sort: egen arr_alcohol=max(t_arr)
163     drop t_arr
164     gen t_arr=(arr_r==0)
165     by id arr_d, sort: egen arr_other=max(t_arr)
166     drop t_arr
167     label define yesno 0 "No" 1 "Yes"
168     label values arr_theft arr_alcohol arr_other yesno
169
170     list if t_dup!=0
171     *Now we can drop the specific arrest descriptions as well as the categorical description
172         *since we have indicators for our categories of arrest for each date:
173     drop arr_r violdesc t_*
174     duplicates drop
175     duplicates report id arr_d
176
177
178     save "long.dta", replace /*Save the long form of this data set*/
179
180
181
182
183     ***EXERCISES****
184     *(It will be helpful to run the code above 1st,
185     * as the questions below use some of the data files that are created.)
186
187     /*Open the long.dta data set. How many duplicates are there on id?*/
188
189     *Answer:
190     use "long.dta", clear
191         duplicates report id
192
193
194     /*import the csv file called main_names.csv  (Hint: you can either search help import
195     or use the interface to determine the code for importing a csv)*/
196
197     import delimited "main_names.csv", clear
198
199     /*How many true duplicates are there? How many duplicates on id? Drop any true duplicates*/
200
201     duplicates report
202     duplicates report id
203     duplicates drop
```

```
204
205   /*Generate a variable which identifies the duplicates, sort by id and then
206   list the first few duplicates (hint: set more off)*/
207
208   duplicates tag id, gen(t_dup)
209   list if t_dup==1 in 1/100  /*Note that in 1/100 command just shows the first 100
210                               observations not the first 100 that meet the criteria*/
211   set more on
212   list if t_dup==1           /*Set more on then click the red (x) at the top to break */
213   set more off
214   /*The duplicates show different versions of the same name, use the string commands
215   learned above so that the names are all in trimmed and in lower case, then remove
216   any remaining true duplicates*/
217
218   replace full_name=trim(full_name)
219   replace full_name=lower(full_name)
220   duplicates drop
221
222   /*How many duplicates on id do you have now?*/
223
224   duplicates report id
225
226   /*Now split the name into f_name for their first name and l_name for their
227   last name. Drop any middle initials.*/
228
229   split full_name, p(" ")
230
231   rename full_name1 f_name
232   rename full_name2 l_name
233   replace l_name=full_name3 if full_name3!="" /*There is only a third name if they
234                                               had a middle name*/
235   replace l_name=full_name3 if length(l_name)<3  /*Another way: All the middle
236                                                  names are middle initials
237                                                  less than 3 characters*/
238   replace l_name=full_name3 if strpos(l_name, ".")!=0 /*Another way: All the middle
239                                                  names are middle initials
240                                                  with a period*/
241   drop full_name3
242
243   /*Tab  last name to view the values.  Do you notice any non-alphabetical characters?
244   Remove them from the first and/or last names*/
245
246   replace l_name=subinstr(l_name, "/", "", .)
247
248   /*Extra credit, do a google search to see how you could remove non-alphabetic characters
249   in the name varaibles without knowing which non-aphabetic characters where in the
      variable*/
250
251    capture noisily ssc install egenmore
252    egen new_l_name=sieve(l_name), keep(a)
253
254
255
256   capture log close
257   exit
258
```