Translation from narrative text to standard codes variables with Stata

Federico Belotti[†] Domenico Depalo[‡]

[†]Università di Roma Tor Vergata, [‡]Bank of Italy

Florence, November 19th, 2009

Outline

1. Motivation

2. The screening command

3. Examples

4. Conclusion

Outline

1. Motivation
2. The screening command
3. Examples
4. Conclusion

Outline

1. Motivation
2. The screening command
3. Examples
4. Conclusion

Outline

1. Motivation

2. The screening command

3. Examples

4. Conclusion

- Raw data collected as narrative text are frequently encountered in applied research
- Examples are:
  1. Electronic Patient Record (EPR) could be used for decision making and clinical research if and only if patient data, which are currently documented as narrative text, are captured in coded form
  2. Different sources of data can use different spelling to identify the same unit of interest
  3. Because of verbatim responses to open-ended questions, survey data items must be converted into nominal categories on a fixed coding frame to be useful for applied research
- These data require (through) treatment before being used
- We provide a command that enormously simplifies such treatment
- `screening` has 2 main features:
  1. Identification
  2. Recoding

  by matching keywords within variables, possibly **from various sources, possibly various keywords**

Why do we need one more command?

- A translation with available Stata commands is possible, **but** it might be:
  1. tedious
  2. risky
  3. time demanding
- while by using `screening`:
  1. flexible & general
  2. safe
  3. fast

```
screening source_var [if] [in], wordscreen([rule] "string") cases (string)
    [ alternative([rule] "string" [[rule] "string" ...])
    sourcecomplementary( string)
    letters(#)
    checksource
    tabcheck
    report (string)
    detail (string)
    newcode( string [, replace])
    recode( "string" [ "string" ..., fromnew])
    save ]
```

Tips

1. The low flexibility of string variables is a reason of concern, because:
   - Capitalizations matters
   - It is important to choose an appropriate matching rule:
     - The number of letters should be specified as the minimum number of letters that uniquely identifies the case of interest
   - Tabulate all encountered cases, as it is the fastest and safest way to detect incorrect matching
2. Advanced users can take maximum advantage of the potentiality of `screening` by mixing keywords with Stata regular expression operators
3. Although **NOT** required, it could be useful to split the original variable

We present 3 examples to illustrate the usefulness and the flexibility of `screening`:

1. Electronic Patient Record
2. Merging from different sources
3. Extracting a piece of string

Example #1: Electronic Patient Record

- We begin with the most complicated case: EPR data from **original** prescriptions of doctors
- Bear in mind that **original** is a polite synonymous for **messy**!!!!
- Suppose the interest is in extraction and generation of standard code determined according to a National Health System (NHS) coding scheme
- Proceed as follows:
  1. First of all convert to UPPERCASE
  2. Break the *source_var* into pieces
  3. run `screening` to find the correct **“recoding syntax”** step by step

Related Code - Example 1

Fix data and variables

```
. replace diagn_test_descr=upper(diagn_test_descr)
(87332 real changes made)
. forvalues i=1/3 {
generate diagn_test_word‘i’=word(diagn_test_descr,‘i’)
}
```

```
. list diagn_test_word1 diagn_test_word2 diagn_test_word3 in 1/15, noobs separator(15)
  +------------------------------------------+
  | diagn_test~1   diagn_test~2   diagn_tes~3 |
  |------------------------------------------|
  | TRIGLICERIDI                              |
  |     EMOCROMO        FORMULA               |
  |  COLESTEROLO         TOTALE               |
  |      ALTEZZA                              |
  |           PT          TEMPO   PROTROMBINA |
  |      VISITA   CARDIOLOGICA     CONTROLLO |
  |          HCV             AB       EPATITE |
  |   COMPONENTE    MONOCLONALE               |
  |    ATTIVITA’         FISICA               |
  |          PSA       ANTIGENE    PROTASTICO |
  |           RX       CAVIGLIA            SN |
  | FAMILIARITA’              K         UTERO |
  | TRIGLICERIDI                              |
  |       URINE          ESAME      COMPLETO |
  |       URINE           PESO     SPECIFICO |
  +------------------------------------------+
```

Related Code - Example 1

1^{st} step → simple syntax

```
. screening diagn_test_word1, wordscreen(COLESTEROLO) cases(cases) report(rep)
. tabulate rep

    rep |      Freq.     Percent        Cum.
--------+-----------------------------------
      0 |     84,133       96.31       96.31
      1 |      3,222        3.69      100.00
--------+-----------------------------------
  Total |     87,355      100.00

. tabulate cases

        cases |      Freq.     Percent        Cum.
--------------+-----------------------------------
  COLESTEROLO |      3,222      100.00      100.00
--------------+-----------------------------------
        Total |      3,222      100.00
```

Related Code - Example 1

2^{nd} step → `letters()` option

Option `letters()` contains the minimum number of letters that uniquely identifies the case of interest

```
. screening diagn_test_word1, wordscreen(COLESTEROLO) cases(cases) letters(5)
.
. tabulate cases

              cases |      Freq.     Percent        Cum.
--------------------+-----------------------------------
              COLES |      1,343       29.26       29.26
            COLEST. |          5        0.11       29.37
         COLEST.TOT |          1        0.02       29.39
        COLEST.TOT. |          4        0.09       29.48
  COLESTER.TOT.HDL, |          1        0.02       29.50
     COLESTEROLEMIA |         14        0.31       29.80
        COLESTEROLO |      3,222       70.20      100.00
--------------------+-----------------------------------
              Total |      4,590      100.00
```

Related Code - Example 1

3^{rd} step → Options `alternative()` and `sourcecomplementary()`

```
. screening diagn_test_word1, wordscreen(beg "COLESTEROLO") cases(cases) letters(5)  ///
                                       sourcecomplementary(diagn_test_word2)   ///
                                       alternative(beg "TOT" "HDL" "LDL")
. tabulate cases
                                 cases |      Freq.     Percent        Cum.
---------------------------------------+-----------------------------------
                           6,9PROTEINE |          1        0.01        0.01
                           8,1PROTEINE |          1        0.01        0.02
                                   BIL |          1        0.01        0.03
                            BILIRUBINA |        643        5.81        5.84
                          BILIRUBINEMIA |          3        0.03        5.87
                             BILRUBINA |          2        0.02        5.88
                                CALCIO |         14        0.13        6.01
                               CLSTHDL |          1        0.01        6.02
                               CLSTLDL |          2        0.02        6.04
                                CO=191 |          1        0.01        6.05
                                   COL |      2,038       18.42       24.47
                     COL=245LDL=193TR=91 |          1        0.01       24.48
                                 COLES |      1,343       12.14       36.61
                               COLEST. |          5        0.05       36.66
                            COLEST.TOT |          1        0.01       36.67
```

Continue on next slide

Related Code - Example 1

3^{rd} step → Options `alternative()` and `sourcecomplementary()`

```
                   COLEST.TOT. |          4        0.04       36.70
             COLESTER.TOT.HDL, |          1        0.01       36.71
                COLESTEROLEMIA |         14        0.13       36.84
                   COLESTEROLO |      3,222       29.12       65.96
                   COLONSCOPIA |          2        0.02       65.98
                 DENSITOMETRIA |          8        0.07       66.05
                           HDL |      2,723       24.61       90.66
                           IGE |         33        0.30       90.96
                           LDL |        626        5.66       96.62
                          LDL, |          1        0.01       96.63
                      LDLGAMMA |          1        0.01       96.64
                        LIPIDI |          1        0.01       96.65
                      MAGNESIO |         97        0.88       97.52
                           PET |          1        0.01       97.53
                      PROTEINE |        253        2.29       99.82
                   PROTEINEMIA |          3        0.03       99.85
                           PSA |          9        0.08       99.93
                      RAPPORTO |          3        0.03       99.95
                        SCINTI |          1        0.01       99.96
                           TAC |          2        0.02       99.98
                   TESTOSTERONE |          1        0.01       99.99
                       TOTALIP |          1        0.01      100.00
---------------------------------------+-----------------------------------
                         Total |     11,064      100.00
```

Related Code - Example 1

Last step → **final syntax** → Options `newcode()`, `recode()` and `detail()`

```
. screening diagn_test_word1 if   regexm(diagn_test_word1, "^[6-8]")==0   ///
        & regexm(diagn_test_word1, "^CAL")==0 & regexm(diagn_test_word1, "^COLON")==0   //
        & regexm(diagn_test_word1, "^D")==0  & regexm(diagn_test_word1, "^FREE")==0     //
        & regexm(diagn_test_word1, "^I")==0  & regexm(diagn_test_word1, "^LDLGAMMA")==0 //
        & regexm(diagn_test_word1, "^LI")==0 & regexm(diagn_test_word1, "^[M-Z]")==0   ///
        & regexm(diagn_test_word1, "^B")==0                  ///
         , wordscreen(beg "COLESTEROLO") cases(cases) letters(5)        ///
        sourcecomplementary(diagn_test_word2) detail(det) report(rep)        ///
        alternative(beg "TOT" "HDL" "LDL") newcode(new_code)
        recode("not classified" "90.14.3" "90.14.1" "90.14.2")

. tabulate new_code

       new_code |      Freq.     Percent        Cum.
----------------+-----------------------------------
        90.14.1 |      2,868       28.73       28.73
        90.14.2 |      2,013       20.16       48.89
        90.14.3 |      5,049       50.58       99.47
 not classified |         53        0.53      100.00
----------------+-----------------------------------
          Total |      9,983      100.00
```

Example # 2: Merging from different sources

- We want to merge 2 (or more) datasets from different sources, where for example truncation of words occurs at different length
- Our example → two Italian datasets: one provided by the National Statistical Office (ISTAT in Italy), the other provided by the Italian Ministry of the Interior
- The two datasets contain, for each Italian municipality, the complete name and an alphanumerical code, the latter being different across sources
- **IF** the name of each municipality uniquely identifies observations, it is easy to merge the two datasets. In fact, **NOOOO**
- We proceed as follows:

  1. Merge all simple cases
  2. Use `screening` to identify and replace the names that are referred to the same municipalities but are spelled differently across the sources

Related Code - Example 2

1st step → merge datasets

```
. use istat,clear
. quietly replace comune=upper(comune)
. sort comune

. merge comune using ministero
variable comune does not uniquely identify observations in the master data
variable comune does not uniquely identify observations in ministero.dta

. tabulate _merge

     _merge |      Freq.     Percent        Cum.
------------+-----------------------------------
          1 |        288        3.43        3.43
          2 |        290        3.46        6.89
          3 |      7,812       93.11      100.00
------------+-----------------------------------
      Total |      8,390      100.00
```

Related Code - Example 2

2^{nd} step → Run `screening` on cases that are spelled differently across the sources

```
forvalues i=1/2 {
preserve
keep if _merge==‘i’

screening comune,wordscreen(ALBISSOLA) alternative("MARINA" "SUPERIORE" "BARCELLONA" ///
       beg "BARZAN" "BRIGNANO" beg "CAVAGLI") cases(cases_ALBISSOLA)      ///
       newcode(comune, replace) recode("ALBISOLA" "ALBISOLA MARINA"      ///
       "ALBISOLA SUPERIORE" "BARCELLONA POZZO DI GOTTO" "BARZANO’"  ///
       "BRIGNANO FRASCATA" "CAVAGLIA’")

if ‘i’==1 drop codice_ente
if ‘i’==2 drop codice
keep  comune codice
sort comune
save new_‘i’,replace
restore
}
keep if _merge==3
save perfect_match,replace
```

Related Code - Example 2

Last step → re-merge datasets

```
. use new_1,clear
. merge comune using new_2
variable comune does not uniquely identify observations in the master data
variable comune does not uniquely identify observations in new_2.dta

. tabulate _merge
     _merge |      Freq.     Percent        Cum.
------------+-----------------------------------
          1 |        282       49.30       49.30
          2 |        284       49.65       98.95
          3 |          6        1.05      100.00
------------+-----------------------------------
      Total |        572      100.00

. append using perfect_match
. tabulate _merge
     _merge |      Freq.     Percent        Cum.
------------+-----------------------------------
          1 |        282        3.36        3.36
          2 |        284        3.39        6.75
          3 |      7,818       93.25      100.00
------------+-----------------------------------
      Total |      8,384      100.00
```

Example # 3: Extracting a piece of string

- We notices that `screening` could be extended in a useful direction, using Stata regular expressions operators
- We extended the command to encompass some built-in Stata functions
- We give an alternative solution to an (unofficial) example provided by *http://www.ats.ucla.edu/stat/stata/faq/regex.htm*

We observe

```
. list, noobs sep(10)

  +------------------------------------------------------+
  |                                              address |
  |------------------------------------------------------|
  | 4905 Lakeway Drive, College Station, Texas 77845 USA |
  |            673 Jasmine Street, Los Angeles, CA 90024 |
  |               2376 First street, San Diego, CA 90126 |
  |                66666 West Central St, Tempe AZ 80068 |
  |              12345 Main St. Cambridge, MA 01238-1234 |
  |            12345 Main St  Sommerville  MA 01239-2345 |
  |             12345 Main St  Watertwon  MA 01239   USA |
  +------------------------------------------------------+
```

Related Code - Example 2

- We want a fast and safe way to obtain the ZIPCODE, i.e. {01238,01239,77845,80068,90024,90126}.
- This is what `screening` produces:

```
. screening address, wordscreen("([0-9][0-9][0-9][0-9][0-9])[\-]*[0-9]*[ a-zA-Z]*$")  ///
                                  cases(c) new(zipcode) recode("regexs(1)")

WARNING: In ([0-9][0-9][0-9][0-9][0-9])[\-]*[0-9]*[ a-zA-Z]*$ you are screening using a
regular expression operator. If you do NOT expect this, try with \; e.g. \* rather than *

. tabulate zipcode

        new |      Freq.     Percent        Cum.
------------+-----------------------------------
      01238 |          1       14.29       14.29
      01239 |          2       28.57       42.86
      77845 |          1       14.29       57.14
      80068 |          1       14.29       71.43
      90024 |          1       14.29       85.71
      90126 |          1       14.29      100.00
------------+-----------------------------------
      Total |          7      100.00
```

Future directions

- We are currently trying to make the command somewhat more flexible
- The command still needs heavy manual interventions by the user that could be implemented within the routine with little effort
- Improve robustness check

THANK YOU FOR YOUR ATTENTION