Development Data Boot Camp
Intro to Stata: More tricks in exploring data

Ge Sun

University of Notre Dame

May 17, 2023

Outline

Outline

Dummy variables

- A Dummy variable usually appears in the regression analysis to represent subgroups of the sample.
- It can only take two values: 0 and 1

Dummy variables

- A Dummy variable usually appears in the regression analysis to represent subgroups of the sample.
- It can only take two values: 0 and 1
- How would this two-dimensional dummy variable can represent the subgroup?

test scores	**girls**	**boys**
89	1	0
73	1	0
56	0	1
92	1	0
87	0	1
64	1	0

  * How many girls are in this sample?

Dummy variables

- A Dummy variable usually appears in the regression analysis to represent subgroups of the sample.
- It can only take two values: 0 and 1
- How would this two-dimensional dummy variable can represent the subgroup?

test scores	**girls**	**boys**
89	1	0
73	1	0
56	0	1
92	1	0
87	0	1
64	1	0

  * How many girls are in this sample?
  * About debugging: Make sure that dummies variables successfully regulate exclusive subgroups.

Dummy variables

- A Dummy variable usually appears in the regression analysis to represent subgroups of the sample.
- It can only take two values: 0 and 1
- How would this two-dimensional dummy variable can represent the subgroup?

test scores	**girls**	**boys**
89	1	0
73	1	0
56	0	1
92	1	0
87	0	1
64	1	0

  * How many girls are in this sample?
  * About debugging: Make sure that dummies variables successfully regulate exclusive subgroups.
  * What does *mean(girls)* mean?

Dummy variables

- A Dummy variable usually appears in the regression analysis to represent subgroups of the sample.
- It can only take two values: 0 and 1
- How would this two-dimensional dummy variable can represent the subgroup?

test scores	**girls**	**boys**
89	1	0
73	1	0
56	0	1
92	1	0
87	0	1
64	1	0

  * How many girls are in this sample?
  * About debugging: Make sure that dummies variables successfully regulate exclusive subgroups.
  * What does *mean(girls)* mean?
  * What would I get if

    gen var = test_scores * girls

Categorical variables

- Variables that take a list of (integer) numbers to indicate each observation belonging to a particular group.
- Example:

educational level	illiterate	primary school	secondary school
edu	1	2	3

- Can I add or subtract categorical variables?

Categorical variables

- Variables that take a list of (integer) numbers to indicate each observation belonging to a particular group.
- Example:

educational level	illiterate	primary school	secondary school
edu	1	2	3

- Can I add or subtract categorical variables?
- Categorical variables are like the ordinal number (compared with cardinal number). You can compare them but you can not calculate them. It is like that you can not say that people finishing secondary school are twice as smart as people finishing primary school.

The new dataset with more continuous variables

- Quarterly Labour Force Survey (QLFS) in 2008, fourth quarter.
  - **Abstract** It is a household-based sample survey conducted by Statistics South Africa (Stats SA). It collects data on the labour market activities of individuals aged 15 years or older who live in South Africa.
  - **Kind of data**: Sample survey data
  - **Unit of analysis** Individual
- **Question**: Can you give some examples of categorical variables in the dataset? Are there any dummy variables?

The data source link

Outline

Let's generate all group dummies at once!

- A magic way to generate dummy variables:

```
tab categorical_var, gen(dummy)
```

Let's generate all group dummies at once!

- A magic way to generate dummy variables:

```
tab categorical_var, gen(dummy)
```

- Regular expressions "*"
  - regular expressions use a notation system that allows for matching complex patterns of text with minimal effort
  - "*" means: match zero or more

Example

```
tab Sector_Sep2003, gen(work_type)
des work_type*
```

Manipulating categorical variables

- Command *recode* helps you recode *categorical* variables.
  - Example:

    ```
    recode race(4=2), gen(race_new)
    ```

    which is equivalent to

    ```
    gen race_new = race
    replace race = 2 if race_new == 4
    ```

Command: *recode*

- It is useful when you have a bunch of survey questions, like our CIF, where you use 1-5 to express ”Strongly disagree” to ”Strongly agree”. But the numbers in different questions have different meanings.
- It is also useful to handle missing values.
    - the survey has missing values for many reasons:
        - refuse to answer this question
        - did not take the survey
        - the answer they give does not make sense

Have a better understanding about categorical variables!

```
codebook Sector_Sep2003
```

Have a better understanding about categorical variables!

```
codebook Sector_Sep2003
```

- Let's make a value label for our new race_new variable
- We will make the label called "black" when race_new is 1 and called "white" when the race_new is 2.

Have a better understanding about categorical variables!

codebook Sector_Sep2003

- Let's make a value label for our new race_new variable
- We will make the label called "black" when race_new is 1 and called "white" when the race_new is 2.
- This is a two-step process:
  1. We first define the label:

     ```
     label define race_newl 1 "black" 2 "white"
     ```
  2. Then we assign the label to the variable:

     ```
     label values race_new race_newl
     ```

Have a better understanding about categorical variables!

codebook Sector_Sep2003

- Let's make a value label for our new race_new variable
- We will make the label called "black" when race_new is 1 and called "white" when the race_new is 2.
- This is a two-step process:
  1. We first define the label:

     label define race_newl 1 "black" 2 "white"
  2. Then we assign the label to the variable:

     label values race_new race_newl
- If you want to rename the label value, you need to first drop the old one:

  label drop race_newl

Have a better understanding about categorical variables!

codebook Sector_Sep2003

- Let's make a value label for our new race_new variable
- We will make the label called "black" when race_new is 1 and called "white" when the race_new is 2.
- This is a two-step process:
  1. We first define the label:

     label define race_newl 1 "black" 2 "white"
  2. Then we assign the label to the variable:

     label values race_new race_newl
- If you want to rename the label value, you need to first drop the old one:

  label drop race_newl
- Another trick:

  label list

Command: *list* and *sort*

- *list*: list values of the variables

```
list in 1/5
list in -5/L
list edyears earnings_week in 1/10
list edyears earnings_week in 1/10, sep(4)
```

- *sort*: arrange observations into ascending order

```
sort earnings_week
sort earnings_week edyears
```

  * Question: What if you want to arrange observations into descending order?

Command: *egen*

- The *egen* command is an extension to *gen* that enables creation of variables that would be difficult to create using *gen*
- For example, suppose we want to create a variable that for each observation equals sample average earnings.

```
egen ave_earnings = mean (earnings)
```

Command: *by* and *bysort*

- *by*: We can use the *by* function to create variables within groups, but in order to use by you must sort beforehand. Thus, we recommend to use *bysort* instead.
- *bysort* + *egen*: the powerful and important code we have when cleaning and managing dataset
  - bysort Indus_Sep2003: egen industry_mean = mean(earnings_week)

Command: *by* and *bysort*

- *by*: We can use the *by* function to create variables within groups, but in order to use by you must sort beforehand. Thus, we recommend to use *bysort* instead.
- *bysort* + *egen*: the powerful and important code we have when cleaning and managing dataset
  - bysort Indus_Sep2003: egen industry_mean = mean(earnings_week)
  - bys Prov_Sep2003: egen sum_employers = sum(Q44NrEmp_Sep2003)

Command: *by* and *bysort*

- *by*: We can use the *by* function to create variables within groups, but in order to use by you must sort beforehand. Thus, we recommend to use *bysort* instead.
- *bysort* + *egen*: the powerful and important code we have when cleaning and managing dataset
  - bysort Indus_Sep2003: egen industry_mean = mean(earnings_week)
  - bys Prov_Sep2003: egen sum_employers = sum(Q44NrEmp_Sep2003)
  - bys Age_Sep2003: egen count_age = count(UqNr)

Command: *by* and *bysort*

- *by*: We can use the *by* function to create variables within groups, but in order to use by you must sort beforehand. Thus, we recommend to use *bysort* instead.
- *bysort* + *egen*: the powerful and important code we have when cleaning and managing dataset
  * bysort Indus_Sep2003: egen industry_mean = mean(earnings_week)
  * bys Prov_Sep2003: egen sum_employers = sum(Q44NrEmp_Sep2003)
  * bys Age_Sep2003: egen count_age = count(UqNr)
  * bys Prov_Sep2003: egen x = sum(female)