How can I detect duplicate observations in Stata?

Authored by

stats writer

July 1, 2024

Stata is a statistical software commonly used for data analysis and management. One of the common tasks in data analysis is to identify and remove duplicate observations from a dataset. Duplicate observations occur when there are multiple entries of the same data in a dataset, which can distort the results of statistical analysis. To detect duplicate observations in Stata, one can use the "duplicates" command. This command allows the user to identify duplicate observations based on specific variables or across all variables in the dataset. It also provides options to either mark, list, or drop the duplicate observations. Additionally, the "isid" command can be used to check for unique identifiers in the dataset, which can help in identifying and handling duplicates. Overall, the use of these commands in Stata can help in efficiently detecting and managing duplicate observations, ensuring the accuracy and reliability of statistical analysis results.

How can I detect duplicate observations? | Stata FAQ

This Stata FAQ shows how to check if a dataset has duplicate observations. There are two methods available for this task. The first example will use commands available in base Stata. The second example will use a user-written program. This user-written command is nice because it creates a variable that captures all the information needed to replicate any deleted observations.

Example 1

This example uses the High School and Beyond dataset, which has no

duplicate observations. Therefore, we add five duplicate observations to
the data, and then use the duplicates command to detect which
observations are repeated. Also, to evaluate the sensitivity of the
command, we change a value of one of the
duplicate observations. The rationale for changing a value is to mimic what may
happen in practice; we often search for "duplicate" cases that are not
identically entered into the dataset. In the dataset, the variable id is the
unique case identifier.

To add the duplicate observations, we sort the data by id, then
duplicate the first five observations (id = 1 to 5). This leads to 195 unique and 5 duplicated observations in the
dataset. For subject id =1, all of her values are duplicated except
for her math score; one duplicate score is set to 84.

```
use https://stats.idre.ucla.edu/stat/stata/notes/hsb2, clear
(highschool and beyond (200 cases))

keep id female ses math read write
sort id
expand 2 if _n
(5 observations created)

sort id
replace math = 84 if _n==1
(1 real change made)

list in 1/15, noobs

+-----------------------------------------+
| id female ses read write math |
|-----------------------------------------|
| 1 female low 34 44 84 |
| 1 female low 34 44 40 |
| 2 female middle 39 41 33 |
| 2 female middle 39 41 33 |
| 3 male low 63 65 48 |
|-----------------------------------------|
| 3 male low 63 65 48 |
```

| 4 female low 44 50 41 |
| 4 female low 44 50 41 |
| 5 male low 47 40 43 |
5 male low 47 40 43
6 female low 47 41 46
7 male middle 57 54 59
8 female low 39 44 52
9 male middle 48 49 52
10 female middle 47 54 49
+----------+

We start by running
the duplicates report command to see the number of duplicate rows in
the dataset. This is followed by duplicate reports id,
which gives the number of replicate rows by the variables specified;
in this instance we have just id. We could have used the duplicates examples
command instead of the duplicates report
command. The duplicates examples command lists one example of each duplicated

set.

duplicates report

Duplicates in terms of all variables

----

copies | observations surplus

----------+-------------------------

1 | 197 0

2 | 8 4

----

duplicates report id

Duplicates in terms of id

----

copies | observations surplus

----------+-------------------------

1 | 195 0

2 | 10 5

Clearly, the output from duplicates report and duplicates

report id differ. The duplicates report output shows the number of replicate rows over all variables. Note that in the duplicate whose value we changed (id=1), the two rows are not technically
the same, and this command correctly did not pick them up. The
second command duplicates report id shows that we have 195
unique id values, and five ids (surplus) that appear two times each
(copies), which leads to a total of 10 questionable observations based on
id.

Next we list duplicate observations with the duplicates list
command.

```
duplicates list, nolabel sepby(id)

+-----------------------------------------------------+
| group: obs: id female ses read write math |
|-----------------------------------------------------|
| 1 3 2 1 2 39 41 33 |
| 1 4 2 1 2 39 41 33 |
```

```
|-----------------------------------------------------|
| 2 5 3 0 1 63 65 48 |
| 2 6 3 0 1 63 65 48 |
|-----------------------------------------------------|
| 3 7 4 1 1 44 50 41 |
| 3 8 4 1 1 44 50 41 |
|-----------------------------------------------------|
| 4 9 5 0 1 47 40 43 |
| 4 10 5 0 1 47 40 43 |
+-----------------------------------------------------+
```

This duplicates list corresponds to listing those observations with duplicate rows; however, as found with duplicates report, it does not identify the five duplicated ids. Therefore, we try duplicates list id.

```
duplicates list id, nolabel sepby(id)

Duplicates in terms of id

+--------------------+
| group: obs: id |
|--------------------|
```

| 1 1 1 |
1 2 1
2 3 2
2 4 2
3 5 3
3 6 3
4 7 4
4 8 4
5 9 5
5 10 5
+---+

Now we see which five subjects are duplicated; however, the duplicate list only lists the variable specified.

We may want to list

the other variables to see which variables are causing the difference between

the duplicates list and duplicates list id outputs. To have an output

like that given from duplicates list, we use the duplicates tag command to create a new variable dup_id that assigns a 1 if the id is duplicated, and 0 if it appears once. Then we list those cases where dup_id is equal to 1.

```
duplicates tag id, gen(dup_id)

Duplicates in terms of id

list if dup_id == 1, nolabel sepby(id)

+-----------------------------------------------+
| id female ses read write math dup_id |
|-----------------------------------------------|
| 1 1 1 34 44 84 1 |
| 1 1 1 34 44 40 1 |
|-----------------------------------------------|
| 2 1 2 39 41 33 1 |
| 2 1 2 39 41 33 1 |
|-----------------------------------------------|
| 3 0 1 63 65 48 1 |
| 3 0 1 63 65 48 1 |
|-----------------------------------------------|
| 4 1 1 44 50 41 1 |
```

```
| 4 1 1 44 50 41 1 |
|-------------------------------------------------|
| 5 0 1 47 40 43 1 |
| 5 0 1 47 40 43 1 |
+-------------------------------------------------+
```

It is apparent that id = 1 has different values on math scores over the duplicate observations. From this, it would be advisable to check which score, if either, is the correct one.

Suppose in this instance both scores were incorrect, and the real score was 44. We correct the scores, and after the correction, the results from duplicates report and duplicates report id should coincide.

```
replace math = 44 if id==1
(2 real changes made)
```

Now, we can use the duplicates drop command to drop the duplicate observations. The command drops all observations

except the first occurrence
of each group with duplicate observations. After we run duplicates drop,
we check that there are no other duplicate observations.

```
duplicates drop
Duplicates in terms of all variables
(5 observations deleted)

duplicates report

Duplicates in terms of all variables

--------------------------------------
copies | observations surplus
----------+---------------------------
1 | 200 0
--------------------------------------
```

It appears that we have gotten rid of the duplicate observations.

Example 2

In Stata, several programs are available to detect the

**duplicates and can also optionally drop the duplicates. One of the
programs is called dups. The program dups is not a built-in
program in Stata, but can be installed over the internet using search dups (see
How can I use the search command to search for programs and get additional
help? for more information about using search). Once dups
is installed we can use it right away. This example uses the following
subset of the larger dataset used above with added duplicates.**

First we enter the data:

```
input id female race read
18 0 1 50
22 0 1 42
26 1 2 60
147 1 4 47
171 0 4 60
191 1 4 47
```

18 0 1 50
18 0 1 50
end

Then we look at them:

list, clean

id female race read
1. 18 0 1 50
2. 22 0 1 42
3. 26 1 2 60
4. 147 1 4 47
5. 171 0 4 60
6. 191 1 4 47
7. 18 0 1 50
8. 18 0 1 50

In our example, we have
one group of observations with duplicates consisting of observation number 1, 7 and 8. That is
what we see below. Now we will use the command dups. Without any arguments, dups
returns information on the number of groups of

observations that have duplicates
and the number of duplicates in each group.

dups

group by: id female race read

groups formed: 1 containing 3 observations
unique observations: 5

groups of duplicate observations:

```
+-----------------+
| _group _count |
|-----------------|
| 1 3 |
+-----------------+
```

We can add a variable list after dups. For example, in the following
example, we add the variable race after dups. Now dups counts
how many duplicate observations in variable race only. We can see
from the list of the data set that there are three groups of observations of

race (1, 2 and 4) and two of them have duplicates. That is shown by dups below.

```
dups race

group by: race

groups formed: 2 containing 7 observations
unique observations: 1

groups of duplicate observations:
+-----------------+
| _group _count |
|-----------------|
| 1 4 |
| 2 3 |
+-----------------+
```

By adding the option unique, we also request information on groups that have a single unique observation. For example,

```
dups id, unique
group by: id
```

groups formed: 1 containing 3 observations
unique observations: 5

groups of duplicate observations:

```
+----------------+
| _group _count |
|----------------|
| 1 3 |
+----------------+
```

unique observations:

```
+----------------+
| _group _count |
|----------------|
| 1 1 |
| 2 1 |
| 3 1 |
| 4 1 |
| 5 1 |
+----------------+
```

With the option key(varlist), we can request to list the observations. For example,
in the following example, we see the values of id in each

group.

dups id, unique key(id)
group by: id

groups formed: 1 containing 3 observations
unique observations: 5

groups of duplicate observations:

```
+----------------------+
| _group _count id |
|----------------------|
| 1 3 18 |
+----------------------+
```

unique observations:

```
+----------------------+
| _group _count id |
|----------------------|
| 1 1 22 |
| 2 1 26 |
| 3 1 147 |
| 4 1 171 |
| 5 1 191 |
+----------------------+
```

An option called terse can be added to get summary information on
duplicates. For example,

dups id, unique key(id) terse

group by: id

groups formed: 1

total observations: 8
in duplicates 3
in unique 5

Now what if we want to drop the duplicates? We can do so by adding an option
called drop. We do want to warn you that it is always dangerous to delete
observations since you may lose your data. So always do it with caution. What is nice
about dups is that it creates a new variable which has enough
information to recover the deleted observations if we change our
mind on what we just did. The default name of the

variable is _expand
(you can change the name by using the option expand after dups). Using the variable _expand we can get the deleted observations back by
using a command called expand. See the example below.

dups, drop key(id)

group by: id female race read

groups formed: 1 containing 3 observations
unique observations: 5

groups of duplicate observations:

```
+----------------------+
| _group _count id |
|----------------------|
| 1 3 18 |
+----------------------+
```

(2 observations deleted)

observations remaining: 6

list

```
+-------------------------------------+
| id female race read _expand |
|-------------------------------------|
1. | 18 0 1 50 3 |
2. | 22 0 1 42 1 |
3. | 26 1 2 60 1 |
4. | 147 1 4 47 1 |
5. | 171 0 4 60 1 |
|-------------------------------------|
6. | 191 1 4 47 1 |
+-------------------------------------+
```

If for some reason you wanted to return to a dataset that has duplicates, you can use the expand command, using the _expand variable created by dups to specify the number of duplicates to be made. Note that this will only recover your original dataset if you detected duplicates based on all variables in your dataset. If you used only a subset of variables, then you will only be able to accurately recreate the values of those cases (because you have no data on

the variables that weren't used
to determine duplicates).

expand _expand
(2 observations created)

list
id female race read _expand
1. 22 0 1 42 1
2. 26 1 2 60 1
3. 147 1 4 47 1
4. 171 0 4 60 1
5. 191 1 4 47 1
6. 18 0 1 50 3
7. 18 0 1 50 3
8. 18 0 1 50 3

Now we have seen how to detect and drop duplicate observations using
the user-written dups command.