Mgmt 469

Getting Your Data Ready for Analysis with Statistical Software

Here are some pointers for getting your data ready for analysis using statistical software.

Putting together a data file

All advanced statistical software packages, including Stata, SAS, SPSS, and even Excel, require you to organize the data in *flat files*. A flat file contains data broken out in rows and columns, such as a single Excel spreadsheet. Here is an example of a flat file:

New York	18000000	10000000
Illinois	10000000	5000000
California	40000000	1000000

In this example, the variable in the first column is a *string* variable that identifies different states. A string variable -- sometimes called an *alphanumeric* variable – contains at least some letters and/or symbols. More to the point, a string variable can not be mathematically manipulated. However, you can sort string variables (e.g., alphabetically) and you can use them to create dummy variables. The syntax for working with string variables in Stata can be a bit tricky, so see me if you need any help!

The second column is a *numeric* variable that gives the population of each state. Numeric variables can be manipulated mathematically; for example, you could divide these values by 1000. Note that there are no commas separating the 000s. Most statistical software use comma to separate variables for large scale data entry and do not react well to commas. (Excel is a notable exception.) The third column is the number of residents in each state who

know how to ice skate. (Don't ask where I got this information from!) This is also a numeric variable.

Most statistical software can figure out on its own whether a column contains a string or numeric variable. If there is just one letter or symbol in one value of one variable, the software will assume that this is a string variable. You will not be able to perform mathematical operations on it. This often occurs when you put some kind of symbol into your spreadsheet, perhaps a "???," when a value is missing. You should leave this blank if you want the other values to be read in as numeric.

If the software makes a mistake, you can always override the computer's default assignment. Your software will also know that any new variable that is created by operating on numeric variables is itself a numeric variable. On rare occasion, you may want to create a new string variable during your work session. I can show you how to do this if the need ever arises.

If you are unsure whether Stata thinks a variable is string or numeric, type **de varname** (for "describe varname"). Or you can type **de** if you want to see a description of all the variables.

Here is a Stata description of the variables in a data set that I used in last year's class: set

```
. de

Contains data from D:\Mgmt469\coursedata\sportsteams.dta
  obs:           323
 vars:            35                          9 Feb 2006 10:33
 size:        83,011  (99.9% of memory free)
-------------------------------------------------------------------------------
              storage  display     value
variable name   type   format      label      variable label
-------------------------------------------------------------------------------
fipsmsa         float  %9.0g                  federal 'FIPS' code for
                                                metropolitan statistical are
MSA_name        str35  %35s                   Metropolitan area name
div_code        float  %9.0g                  (mean) div_code
nhl             byte   %8.0g                  number of National Hockey
                                                League franchises
```

Note that Stata reports the names of each variable as well as the *storage type*. The storage type for numeric variables can be "float" or "byte." The difference has to do with how Stata organizes its memory space and need not concern you. The storage type for string variables is strxx, where xx represents the maximum number of characters permitted in the string. (For example, MSA_name, or Metropolitan area name, contains up to 35 characters.)

Recall that flat files are broken into rows and columns. Each column contains a different variable. Each row contains a different observation of these variables. The *unit of observation* of your data set is the thing that distinguishes one row from another. In the skating example that opens this note, the unit of observation is the state. (Each row in the data is a distinct state.) If we had data from five years for each state, there would be five times as many rows of data, and the unit of observation would be the state/year. (Each row would be a distinct state/year pair.) *It is absolutely vital that you are able to identify the unit of observation of any data set you work with.* This will allow you to choose the correct empirical model, explain your findings, and merge your data with other data. When you read empirical research by others, one of the first questions you should ask yourself is "what is the unit of observation?"

Going from spreadsheet data to statistical software data

If you are using a spreadsheet to organize your data, then you already have a flat file. Be sure that you can identify the unit of observation. Also be sure that you have one observation per row. Once you have done this, it is very easy to generate a Stata data set. Here is how:

1) Give names to all of your variables. The names should appear in the first row of your spreadsheet. Use names that others will understand. *Be sure there are no spaces in your variable names*.

2) Don't worry if some of your data cells are blank. Stata will treat these as missing values.

3) Eyeball your data. If you can, be sure to eliminate all letters, commas, and other symbols from columns containing numeric data.

4) Save your spreadsheet as a *tab delimited text file*. Give it a name like myfile.txt.

5) Open Stata. Within Stata, type **insheet using c:path\myfile.txt** where c:path\myfile.txt is the full logical path and file name of the text you just created.

(Alternatively, you can use the DOS command **cd c:path** to change to the folder that contains the text file and then type **insheet using myfile.txt**)

6) You now have an active Stata data set. If you named the variables in the first row of your Excel spreadsheet, Stata will use these names. If you failed to provide names, Stata will name the variables v1, v2, etc. You can use the **rename** command to changes these names. For example: **rename v1 state**

Remember, when reading in the spreadsheet, Stata will try to guess which variables are string variables and which are numeric. If a column in your spreadsheet contains all numbers, Stata will treat the variable as numeric. But if any entry in the column, even just one, contains one or more non-numeric characters, Stata treats the variable as a string.

Reading in plain text

You can also generate a statistical data set from a text file. Your data should be in either *free format* or *fixed format*. In free format, you will have one row of data per observation and spaces or tabs separating each variable. In fixed format, you will also have one row of data per observation but each variable must appear in a specific column or columns. You will not need to separate the values with spaces or tabs. If I have the choice, I prefer to work with free format data, although it is not overly difficult to work with fixed format data.

Inputting free format data is easy if the data is prepared properly:

1) Start with a flat file that you can save in text format.

2) Be sure that there are spaces or tabs between each and every value in the data, and that there are no spaces within string variables. (E.g., New York would need to be rewritten NewYork).

3) Be sure that there is a value for every observation and every variable. If you are missing a value, do not leave the field blank. Instead, use a readily identifiable *placeholder*. Stata will recognize "." (i.e., a period) as a missing value. You may prefer to assign the value such as -999 as a placeholder for missing observations and then use Stata to replace this with the missing value designation: **replace var = . if var == -999**. The advantage of doing this is that when you eyeball the data, you will know that if any variable equals -999, in reality the data is missing.

4) Get rid of any unwanted commas. For example, rewrite 97,235 as 97235.

5) Save the data in text format.

6) Use the **infile** command to input in the data. Here is an example, including a demonstration of the correct syntax:

Suppose we had fixed up the ice skating data as follows:

NewYork 18000 10000
Illinois 10000 5000
California 40000 1000

Note that I have converted both population and number of skaters into thousands, and there are no commas. Note further that NewYork is one word. We save the file as c:\path\puck.dat.

We can now type:

infile str15 state population skaters using c:\path\puck.dat

Here how the infile command works:

1) **infile** alerts Stata to input data from a text file. Following **infile** is a list of variable names of our choosing – in this case **state population skaters**.

2) We include the expression **str15** to inform Stata that **state** is a string variable that may be up to 15 characters long.

3) **using c:\path\puck.dat** tells Stata where to find the text file (the text file can have any name – I prefer to use .dat or .txt as my suffix.)

Fixed format entry is a bit more difficult, but I can show you how to do it if you are interested. The main thing is to keep track of the columns for each variable, as you will need to specify either column numbers when you input the data.