

POL 681: Stata for Regression Analysis

1 Getting Started

When using Stata for regression analysis, or anything else for that matter, the first thing you have to do is make sure Stata has data upon which to execute the analysis you specify. Typically, you will be using so-called Stata datasets; that is, data sets that are converted to code such that Stata can easily read them. Analysis is much faster if you use a data set in Stata format than a data set in plain text (ascii) format. There is a program called **Stat Transfer** that will allow you to convert data sets from one format (commonly, this means Excel (.xls) format, SPSS format, or plain text format) to another (for our purposes, Stata format). Note that the default extension for Stata datasets is .dta.

Once your data are converted to Stata format, reading in the data is simple. This tutorial is based on a data set called 2000flor. Save the file to a relevant directory on your computer (or to a disk, CD-R, or zip disk). Type the following Stata command:

```
use a:\2000flor
```

Obviously, if the file is not on the “a” drive, you’ll need to specify another locale. Note that you do not need to specify the “.dta” extension (although you could if you wanted to). The `use` command is the Stata command that will be subsequently used by you to access your data sets.

After reading your data into Stata, the variables in the data set will appear on the left side of the screen. For the Florida data, you should see 12 variables.

2 Basic Statistical Analysis in Stata

2.1 Univariate Statistics

To illustrate some of the basic commands in Stata, note the following commands. To obtain summary statistics of the variables, type:

```
summarize varlist
```

where *varlist* corresponds to the variables for which you want summary statistics. So typing,

```
summarize Bush Buchanan Gore
```

for this data set will get you the mean, standard deviation, and maximum and minimum values for the variables. Most Stata commands have several options that you can specify. If you ever need assistance with these options, note that Stata has outstanding on-line help files. So if you type,

```
help summarize
```

you will get the on-line help screen for the `summarize` command. Often, you may want more descriptive information about your variables than just the mean and standard deviation. The `detail` option gives you the full set of univariate statistics for each variable:

```
summarize Bush Buchanan Gore, detail
```

2.2 Simple Correlation

To obtain the correlation between or among variables in a data set, type

```
correlate varlist
```

(if *varlist* is omitted, you will get the correlation matrix for each variable in the data set. Again, Stata's help files are useful. Typing `help correlate` will allow you to see the many options available. To obtain the *p*-value for the test of significance, you need to use the `pwcorr` command:

```
pwcorr varlist, sig
```

2.3 Graphing Two Variables

To visually inspect your data, you can take advantage of Stata's graphing capabilities. I will give you a separate handout regarding Stata graphing options. For now, let's simply graph some of the variables in your data set using a few of Stata's options. If you type

```
graph Buchanan Bush
```

you will get an *xy* graph of the Buchanan vote (the *y* axis) by the Bush vote (*x* axis) for each county in Florida. Stata formats the axes using defaults computed from the actual data. You can make the graph look nicer by typing

```
graph Buchanan Bush, ylab xlab
```

which gives you nicer looking *x* and *y* axes. If you type

```
graph Buchanan Bush, ylab xlab b1("Buchanan Vote by Bush Vote in Florida Counties")
```

you will place a title at the bottom (hence "b1") of the graph. Please note that there are numerous other graphing options, but for now, we'll stick with the easy commands. As a side note, it is possible to give your variables labels such that when they are graphed, the label names will appear (this can enhance the visual presentation of the data). So, for example, type

```
label var Buchanan "Buchanan Votes"
```

and

```
label var Bush "Bush Votes"
```

and you will give labels to the "Buchanan" and "Bush" variables. Now if you retype your graph command,

```
graph Buchanan Bush, ylab xlab b1("Buchanan Vote by Bush Vote in Florida Counties")
```

the labels you typed will appear on the axes. Finally, it is useful to note how you can change the symbols that are plotted on graph. For example, suppose you wanted to know which data points corresponded to which county. To do this, you could type

```
graph Buchanan Bush, ylab xlab b1("Buchanan Vote by Bush Vote in Florida Counties") s([County])
```

which would plot the names of the counties on the graph. Note that with lots of data points, you'll usually want to rely on Stata defaults. However, in looking for possible outlying data points and potential influential observations (or coding errors), this is a good option to use. Take note where Palm Beach County lies in your graph. What is noteworthy about this data point?

Note that to save your graph, you need to first create it, then click on the **file** button in the upper left corner and click on the **save graph** option. If you are working in the data lab, I recommend saving the graph to a diskette. Later in the semester, we will learn how to export the graphic into a wordprocessing file.

3 Regression Analysis

Now the moment we've all been waiting for: regression analysis. Estimation of a simple regression model is very simple. By typing

```
regress Buchanan Bush
```

you are invoking Stata's **regress** module to produce a least squares regression model of the form $y_i = a + b_1x_i$, where y , the dependent variable corresponds to the Buchanan vote and x , the independent variable, corresponds to the Bush vote. Aside from the coefficient estimates, you get the standard information generated from the model: standard errors, variance components, goodness-of-fit indicators. Note that the "Source" table at the top of the output corresponds to the variance components. Hence the number in the cell denoted by "Model" (on the row) and "SS" on the column is the sum of squares due to the regression (i.e. the variance "explained."). Notationally, we referred to this in class as $\Sigma(\hat{y}_i - \bar{y})^2$ and called it the SSR, or model sum of squares. The number immediately below the SSR is the sum of the squares due to error, or the SSE (Stata uses the term "Residual" to denote this variance component). Notationally, this number is $\Sigma(y_i - \hat{y}_i)^2$. Note also that the SSR divided by the total sum of squares gives you the R^2 .

3.1 Obtaining Predicted Values and Residuals

Usually, you will be interested in assessing predicted values for all or some of your observations; moreover, analysis of residuals will prove to be helpful in assessing the adequacy of your model. In Stata, obtaining predicted values and residuals is easy. Note, first, that to reprint on the screen the results of the model you *most recently executed*, type the root command again. That is, to reprint the results from your regression model, type

```
regress
```

and Stata will redisplay your results. Suppose you were interested in obtaining the predicted values, \hat{y} , for each observation from the model just estimated. If you typed

```
predict PredBuch
```

Stata would compute \hat{y}_i and save the results in a newly created variable called "PredBuch" which is a variable denoting the predicted value of the Buchanan vote. Note that to give this variable a more informative name, you can label it by typing

```
label var PredBuch 'Predicted Buchanan Vote'
```

which is a far more informative name to use in your output. In order to see what your predicted values look like compared to the observed values, you can list the data by typing

```
list County Buchanan PredBuch
```

thus allowing you to quickly compare values. Of course the difference between the observed data and the predicted values correspond to the *residuals*. You can use Stata's `generate` command to create a variable storing the residuals. By typing

```
generate res=Buchanan-PredBuch
```

you will obtain the residuals, which are stored in a new variable called "res". The `generate` command is an all-purpose command you can use to generate new variables from existing data; however, there is a more efficient way to get the residuals, and that is through using options in the `predict` command. By typing

```
predict resid, residuals
```

Stata will automatically create the residuals for you and store the results in the variable "resid." Now, if you type

```
list County Buchanan PredBuch res resid
```

you will see the observed, the predicted, and the residual values for each data set. To verify to yourself that the sum of the residuals is 0, type

```
summarize resid
```

and you will note that the mean of the residuals is to any practical level of precision, 0. Now type

```
gen SqResid=resid^2
```

and you've just created a variable called "SqResid", that is, the square of the residuals. If you were to sum this variable, divide through by $n - 2$, and take the square root, you would have the *standard error of the regression* (see Fox, p. 90). Of course it would be silly to do this for a large data set as Stata does it for you. Look again at the variance component denoted by the SSE. It's value for this regression should be 8141533.66. This is the sum of the squared residuals. Dividing this number through by $n - 2 = 65$ (question: why $n - 2$?), you get the *variance* of the residuals, or what is called the *Mean Square Error* (i.e. it is the average value of the squared residuals), which is equal to 125254.364. Taking the square root of the MSE (i.e. $\sqrt{125254.364}$) gives you the standard error of the residuals, which is 353.91. This is the standard error of the regression and is a statistic that can be used to describe goodness-of-fit. Note that this quantity is typically referred to as the *Root Mean Square Error* or *Root MSE* in many software packages. Look on your Stata output and find the Root MSE. What does it equal? (It should be the same number as above!)

3.2 Graphing Quantities of Interest

It is useful to graph relationships in a linear regression model. For the bivariate case, graphs are very easy to generate and interpret (they are also easy in the multiple regression setting, and we'll learn how to do this later). Suppose you wanted to graph the predicted values by x . Using the results from your model, type

```
graph PredBuch Bush, ylab xlab c(1) b1("Buchanan Vote by Bush Vote")
```

which is similar to the command typed earlier, with addition of the `c1`, which tells Stata to connect the predicted values with a straight line, hence the "c" denotes "connect" and the "1" in the parentheses denotes "line." This graph obviously will produce a straight line, where the slope of the line corresponds to the slope from the regression model and the y -intercept corresponds to the constant term estimated in the model.

To visualize the predicted regression line in relation to the actual data points, it is useful to include the observed values of the dependent variable in the graph. Hence, by typing

```
graph PredBuch Buchanan Bush, ylab xlab c(1) b1("Buchanan Vote by Bush Vote")
```

you will get the observed data and the predicted regression line in the same graph. Note that by including the `symbol` option as used earlier, you can include the county name on the graph:

```
graph PredBuch Buchanan Bush, ylab xlab c(1) b1("Buchanan Vote by Bush Vote") s(o[County])
```

Again, note anything unusual about Palm Beach County? If you want to see just how unusual Palm Beach County is relative to the other data points, you can graph the data in terms of the residuals. First, let's re-graph the previous graph but change one thing. If you type

```
graph PredBuch Buchanan Bush, ylab xlab c(1) b1("Buchanan Vote by Bush Vote") s([County])
```

you will see where the residuals come from in terms of the actual data (it may be helpful to omit the "s" command):

```
graph PredBuch Buchanan Bush, ylab xlab c(1) b1("Buchanan Vote by Bush Vote")
```

(this enables you to more clearly see the data points). Clearly, something is unusual about Palm Beach County. Now, if we graph the residuals by the predicted Buchanan vote, we obtain what is sometimes called a "residual-versus-fitted plot." In Stata, there is a built-in module that will allow us to do this. By typing

```
rvfplot, yline(0)
```

you get this plot. The `ylines(0)` option places a horizontal reference line at the 0 point (q: why 0?). Again, we see something is amiss with Palm Beach County. If we type

```
rvfplot, yline(0) s([County])
```

the county names are placed on the graph.

4 Conclusion

This is a simple introduction to using Stata for regression. Later, we will learn how to construct 95 percent confidence intervals around the estimates and do additional procedures that will facilitate interpretation and presentation of our statistical results. Also, we'll learn to take advantage of Stata's suite of regression diagnostics.