Chapter 16 Coding Homework The Effect

Zahid Asghar

Updated 2022-10-25

Follow the below instructions and turn in both your code and results:

1. Load the mathpnl.csv data file provided (in R or Python store it as mp), which comes from Leslie Papke and consists of data at the school district level, and was featured in the Wooldridge (2010) textbook.

* We are only going to be working with a few variables. Limit the data to these variables:
  + distid: the district identifier (our “individual” for fixed effects)
  + year: the year the data is from
  + math4: the percentage of 4th grade students who are “satisfactory” or better in math
  + expp: expenditure per pupil
  + lunch: the percentage of students eligible for free lunch

1. Panel data is often described as “N by T”. That is, the number of different individuals N and the number of time periods T. Write code that outputs what N and T are in this data.

*Language-specific instructions*:

* This will entail counting the number of unique values of your individual and time identifiers. In R try unique() to get the unique values and length() to count how many there are, or similarly pd.unique() and len() in Python. In Stata try codebook to get the number of unique values.

1. A *balanced* panel is one in which each individual shows up in every single time period. You can check whether a data set is a balanced panel by seeing whether the number of unique time periods each individual ID shows up in is the same as the number of unique time periods, or whether the number of unique individual IDs in each time period is the same as the total number of unique individual IDs. Think to yourself a second about why these procedures would check that this is a balanced panel. Then, check whether this data set is a balanced panel.

(hint: there are many ways to do this, but the easiest is to limit the data to just individual ID and year, drop any duplicates (keeping only unique() values in R, drop duplicates in Stata, or .drop\_duplicates() in Python), and tabulating how many times each year appears (table() in R, tabulate in Stata, .value\_counts() in Python))

1. Run an OLS regression, with no fixed effects, of math4 on expp and lunch. Store the results as m1.

*Language-specific instructions*:

* Yes, store them in Stata too. Following your regression use estimates store.

1. Modify the model in step 4 to include fixed effects for distid “by hand”. That is, subtract out the within-distid mean of math4, expp, and lunch, creating new variables math4\_demean, expp\_demean, and lunch\_demean, and re-estimate the model using those variables, storing the result as m2.

(tip: be careful that your code doesn’t overwrite the original variables, or at least if it does, reload the data afterwards)

1. Next we’re going to estimate fixed effects by including distid as a set of dummies. This can be extremely slow, so for demonstration purposes use only the first 500 observations of your data (don’t get rid of the other observations, though, you’ll want them for the rest of this assignment). Run the model from step 4 but with dummies for different values of distid, saving the result as m3. Then, do a joint F test on the dummies (see Chapter 13), and report if you can reject that the dummies are jointly zero at the 99% level.

Tip: distid is stored as a numeric variable, but you want it to be treated as a categorical variable. If your regression result only has one coefficient for distid you’ve done it wrong.

*Language-specific instructions*:

* For Python, visit [this page](https://www.statsmodels.org/stable/generated/statsmodels.regression.linear_model.OLSResults.f_test.html) of the **statsmodels** documentation. The first example shows how to do a joint F-test comparing a long list of parameters to 0. You’ll need to change the 1: part, though. So pay attention to which parameter index numbers have the coefficients you want to test.

1. Now we will use a specially-designed function to estimate a model with fixed effects. (Using the whole data set once again), use feols() from the **fixest** package in R, reghdfe from the **reghdfe** package in Stata, or PanelOLS from **linearmodels** in Python to estimate the model from step 4 but with fixed effects for distid. Save the result as m4. Include standard errors clustered at the distid level.
2. Now add fixed effects for year to your model from step 7 to create a two-way fixed effects model. Keep the standard errors clustered at the distid level. Save the results as m5.
3. Using modelsummary() from **modelsummary** in R, esttab from **estout** in Stata, or Stargazer from **stargazer.stargazer** in Python, make a regression table including m1 through m5 in the same table so you can compare them all. Read the documentation of your command to figure out how to include the expp, lunch, expp\_demean, and lunch\_demean predictors in the table without clogging the thing up with a bunch of dummy coefficients from m3.

Write down two interesting things you notice from the table. Multiple possible answers here.

1. Finally, we’ll close it out by using correlated random effects instead of fixed effects (see 16.3.3). You already have expp\_demean and lunch\_demean from earlier. Now, modify the code from that slightly to add on expp\_mean and lunch\_mean (the mean within distid instead of the value *minus* that mean). Then, regress math4 on expp\_demean, lunch\_demean, expp\_mean, and lunch\_mean, with random effects for distid using lmer() from **lme4** in R, xtreg, re in Stata, or RandomEffects from **linearmodels** in Python. Show a summary of the regression results.

*Language-specific instructions*:

* In R, lmer() has a hard time with numeric variables as categorical indicators. Create a new, factor version of distid directly in the data before running the model, and use that instead.