Matching Lab

After today's lab you will be able to analyze the extent to which covariates are balanced between treatment and control groups and use Stata to produce matched data and estimate causal effects. To achieve these objectives, we will (1) use General Social Survey data (from earlier in the course) to estimate causal effects of public versus private employment status, and (2) use data from the Political Socialization Panel Study to replicate analyses by Kam and Palmer and Henderson and Chatfield on the causal effects of university education.

GSS Data

1. Download the `GSS2002.dta` data file from Blackboard.

2. Download the `Matching.do` file (if you don't already have it) and add the following analyses. We can save ourselves some time by using the some of covariates we already know how to recode from working with Houston's data in Session 6.

3. Create a new variable, `pubemp`, to represent public employment, using `recode wrkgovt 2=0, gen(pubemp)` (this is already in the do-file)

Balance Testing

4. The first step in preparing for matching is assessing covariate overlap (common covariate support). This can be done using tables or graphs, and both unidimensionally and multidimensionally.

   - Assess common covariate support on several covariates among those attending university and those not attending university (variable `college`) using `centile` and/or `summarize` with the `by` syntax. For example: `by pubemp, sort:  centile age, c(0 100)` or `by pubemp, sort:  summ age, d`.
   - Repeat this analysis using the `graph box` (or `graph hbox`) command to produce visual summaries of covariate distributions.
   - Use a scatterplot to compare common covariate support on two dimensions. Use the `jitter` option to clarify categorical variables. For example:
     ```
     twoway (scatter age educ if pubemp == 1, jitter(3))
     (scatter age educ if pubemp == 0, jitter(3)),
     legend(label(1 Public) label(2 Private))
     ```

5. If the treatment and control groups have common covariate, we typically then focus on balance across groups in terms of the covariate's mean in each group. Use `summarize`, again with the `by pubemp, sort:` prefix to compare the means of covariates in each group.

6. We can also compare balance across groups by looking at density plots to see not only whether the distributions are mean-balanced but whether they are similar. This is not critical for matching, but the more similar the distributions, the less likely there is to be bias in the causal effect estimate attributable to this variable. Use the `kdensity` plotting command to produce comparative density plots of (continuous) covariates for the treatment and control groups. For example:
   ```
   twoway (kdensity educ if pubemp == 1) (kdensity educ if pubemp == 0),
   legend(label(1 Public) label(2 Private))
   ```

7. After you've assessed balance on a number of covariates, you should have a sense of which are already balanced (and thus unrelated to treatment assignment) and those that are not. We need to match on everything that imbalanced. We can also try to match on other variables, at the cost of making matching more difficult with potentially little gain in bias reduction. At this point you should have a list of variables you want to match on.

Matching

Stata 13 includes several matching commands as part of its built-in `teffects` package. We will also use an add-on package called `CEM` (for exact matching). To install it, use: `ssc install CEM, replace`

Matching is a fundamentally an iterative process. Once the subset of observations within the range of common covariate support are identified, the actual matching of treatment to control units needs to take place by identifying one treatment unit, matching it it to one or more control units and then continuing with the next treatment unit. (Depending on whether matching is done *with replacement*, the matched control units might remain available for further matching.) As a result, it is hard to show how matching works "behind-the-scenes". You could do matching by-hand, by sort the data by covariate values and identify observations with matching covariate values (or matching propensity score values). But this is a tedious exercise. You're welcome to try it at home on your own. We'll instead let Stata do most of the work for us.

Exact Matching

8. The first matching technique we can look at is exact matching (which is implemented via the CEM add-on in command `cem`). This command will print some information about the ability to find exact matches within the data and then create some new variables to identify matches that we can use in subsequent analysis.

9. Try running cem with just a single covariate to see the output, for example: `cem educ` . Stata will report that it has formed several strata within the data of nearly-exact matches. We can see how our original variable is reflected in this set of strata by doing a simple cross-tabulation of the new variable against our original variable: `tab educ cem_strata` .

10. If we specify a `treatment()` option to `cem`, it will further identify which treatment and control cases can be matched. Try this with: `cem educ, tr(pubemp)` .

11. Now try to specify a more complete list of covariates to match on. Depending on how many variables you use and the number of levels you should run up against the "problem of dimensionality" and no few or no matchable cases will be found. You could try, for example: `cem gender racebin educ married age hhchildren, tr(pubemp)`

12. Looking at `tab cem_strata`, you should find that each observation has been assigned to its own stratum and is, therefore, unmatchable.

13. We can address the problem of dimensionality by coarsening our covariates. You can either do this manually (by recoding) variables, or you can specify "cutpoints" telling `cem` where to cut your variables. For example: `cem gender racebin educ (16) married age (25 50 75) hhchildren (1), tr(pubemp)` . The result is that some cases are now matched, while many remain unmatchable.

14. `cem` prints the numbers of matched cases automatically, but you can also see these reflected in the `cem_weights` variable. Cases that cannot be matched are given a weight of zero, while those can be matched are given a positive weight (the precise value of which depends on how many cases they are matched against).

15. Recall that with exact matching we do not need to do post-matching balance testing because treatment and control groups are matched by definition; the only cases we included in the matched subset are those with an exactly matching counterpart in the other level of `pubemp`.

16. Now, let's use this matched dataset to estimate a causal effect. We'll look at a few different measures, included in the do-file.

17. Estimate the "naive causal effect" using either `pwmeans`, `ttest`, or `reg`. All three methods of estimating the difference-in-means between public and private employees on the `jobstress` variable should give the same results. For example: `reg jobstress pubemp`

18. You can try the same using a binary outcome measure, like `bloodbin` and a logit model: `logit bloodbin pubemp`

19. Now estimate these models on only the matched data. To do this we need to specify a "weights" option to the regression models. For example: `reg jobstress pubemp [iweight=cem_weights]` or `logit bloodbin pubemp [iweight=cem_weights]` . How do these results compare to the original results? How many cases are the results estimated on?

Propensity Score Matching

20. Our experience with exact matching and coarsened exact matching shows that we have to discard a lot of data due to the "problem of dimensionality." We can overcome this problem to some extent by estimating a propensity score model and matching exclusively on that.

21. Estimate a propensity score model using `logit` to predict `pubemp` as a function of the imbalanced pretreatment covariates you identified above.

22. Estimate propensity scores from the model using `predict` on both the linear (log-odds) and probability scales. For example: `predict pscore, pr` and `predict pscorexb, xb`. Plot the two versions of the propensity score to understand how the predicted probability is a simple non-linear transformation of the linear (log-odds) prediction: `twoway scatter pscore pscorexb`

23. Assess (pre-matching) balance in the propensity scores using a comparative density plot:
```
twoway (kdensity pscore if pubemp == 1)
(kdensity pscore if pubemp == 0),
legend(label(1 Public) label(2 Private))
```
Note: These distributions should overlap (to have common support) but need not be identical since we haven't done any matching yet. If the distributions do not overlap much at all, repeat the steps in this section with different combinations of covariates until the propensity score distributions have greater overlap.

24. We can now match on the propensity score using `cem` but this time using just `pscore` or `pscorexb` as a single covariate: `cem pscore, tr(pubemp)`. We should be able to produce greater numbers of matches than when matching directly on the covariates.

25. Given that we're no longer using exact matching, we should assess balance before proceeding with estimating any effects on outcomes. We can do this the same was as pre-matching but we now need to account for how the observations are weighted by the matching procedure. The observations that are unmatchable are given weights of 0 in the `cem_weights` variable and are therefore ignored when we assess balance. Start by assessing balance on the propensity scores using `by pubemp, sort:  summ pscore [iweight=cem_weights]` .

26. Repeat the balance assessment for the original covariates, using the same syntax. Remember, this is what Kam and Palmer forgot to do!

27. Now, using the same regression approaches as above, reestimate the effect of `pubemp` on outcomes. Do the results change? How many cases are the results estimated on?

Propensity Score Subclassification

28. We can further attempt to preserve cases by using propensity score subclassification. This process involves cutting our propensity score into a few discrete levels and then matching on those levels. Generate propensity score subclasses using the `egen` command: `egen pclass = cut(pscore), group(5)`. Note: the `group(5)` option indicates you are requesting five subclasses. You can use `tab pclass` to confirm that this variable has been created correctly and `twoway scatter pclass pscore` to see where Stata created the cutpoints and `tab pclass pubemp` to see the distribution of treated and control units across subclasses. You can also try more subclasses or manually creating the cutpoints by looking at the quantiles of the `pscore` variable.

29. Again, we need to assess balance in the original covariates. By subclassifying, we're allowing units that are somewhat dissimilar on the propensity score to matched together. We need to know if this introduces imbalance in our original covariates.

30. Now use `cem` to match on `pclass`. Then reestimate the effects on the outcomes using the updated weights. Do the results change? How many cases are the results estimated on?

One-step matching and estimation

31. Stata's built-in `teffects` command implements matching and effect estimation in one step. This is convenient for quickly estimating effects from a propensity score model (or one of several other matching techniques) but it masks the procedures we just went through in order to produce matches. As a result, I prefer to use `cem`, but you can also use `teffects`. Here we'll show a few of the ways to use `teffects`. Note: They may not produce the same results do to differences in how the programs identify matches and estimate the resulting effects.

32. The `teffects` command takes the form of:
    `teffects psmatch (outcome) (treatment covariate1 covariate2 ...)`
    In other words, we combine the effect estimation with the matching procedure. Try it with the `jobstress` variable: `teffects psmatch (jobstress) (pubemp gender racebin educ married age hhchildren)` . Do the results change? How many cases are the results estimated on?

33. By default, `teffects` make some relatively strong assumptions about how you want the matching procedure to proceed. In short, a nearest-neighbor algorithm is implemented that allows any treatment unit to be matched to any control unit. This means that all cases are automatically matched regardless of their similarity on the propensity score. We can adjust this tolerance using the `, caliper()` option, which specifies the maximum distance two units can be from each other (on the propensity score) in order to be considered matches. If you reestimate the effect with option `, caliper(1)` , the results will be identical. See what happens if you try setting a smaller caliper (e.g., 0.1 or 0.01). If a small caliper prevents Stata from matching a given unit, the procedure will fail.

34. We can install another add-on package called `psmatch2` that handles these failures differently. Install it with: `ssc install psmatch2, replace` .

35. `psmatch2`, unfortunately, again uses a different syntax for matching:
    `psmatch2 (treatment covariate1 covariate2 ...), outcome(outcome) ate`
    Note: We need to specify the `ate` option to compare these results to those from `teffects`.

36. Try reestimating our model using `psmatch2`: `psmatch2 (pubemp gender racebin educ married age hhchildren), outcome(jobstress) ate`

37. Now, try specifying a small caliper. Now, `psmatch2` will simply discard unmatchable observations (those that fall off of "common support") and estimate the effect.

38. Use the techniques we used earlier to assess pre-matching balance in the covariates to assess post-matching balance in the covariates. For example: `by pubemp, sort: summ educ [fweight=_weight]`

Matching and Regression

39. Compare the following estimates of the effect of `pubemp` on `jobstress`:

    - `reg jobstress pubemp [iweight=cem_weights]`
    - `teffects psmatch (jobstress) (pubemp gender racebin educ married age hhchildren)`
    - `psmatch2 (pubemp gender racebin educ married age hhchildren), outcome(jobstress) ate`
    - `reg jobstress pubemp gender racebin educ married age hhchildren`

    How do the results compare? Which should we trust?

Replicate Education's Effect on Political Participation

If you have time or want to work on this more at home, consider replicating the Kam/Palmer and Henderson/Chatfield results. Note: You will not be able to do "genetic matching" because the technique is not implemented in Stata.

40. Download the `WhoMatches.dta` data file from Blackboard.

41. Open the file and start a new `.do` file to contain the following analyses.

42. The outcome participation measures are listed as "y1982". All have missing values coded as `99`. Recode these to missing using: `recode y1982vote76 y1982vote80 y1982meeting y1982other y1982button y1982money y1982communicate y1982demonstrate y1982community (99=.)`.

43. Create an outcome variable `out` that is a sum of these variables.

44. Use the balance testing techniques above to assess pre-matching balance of covariates and propensity scores, if appropriate.

45. Use the matching procedures we've tried here to generate estimates of the effect of university education (the `college` variable) on the outcome participation index, and/or on individual types of participation.

46. Use the balance testing techniques to assess post-matching balance of covariates and propensity scores, if appropriate.