HW 2: Average Treatment Effects and Heterogeneous Treatment Effects

See the course syllabus for more instructions about working in teams. Students should turn in individual write-ups but may collaborate on code.

Getting Data and R Packages

For the first part of this assignment, please continue with the "altered" experimental dataset you used in the last assignment. For the second part, go back to the original "un-altered" experimental dataset. (Or you can switch datasets for the second part—but you should work with an experimental dataset.)

To install packages:

- You need to install the package "devtools"
- On Windows, you also need RTools installed. That is available here: https://cran.r-project.org/bin/windows/Rtools/
    - R will prompt you to install if you have not. I found that I had to copy the files that were installed from c:\rbuildtools\3.4\ into a different directory, c:\Rtools (with no version number, e.g. no 3.4 subdirectory, below it)
- There is a bug in R 3.4 that prevents installing packages. To get the patch, find it here: https://cran.r-project.org/bin/windows/base/rpatched.html

For causalTree and causalForest:

- To install:
    - install_github("susanathey/causalTree")
- To see some examples of how to use causalTree, see this:
    - https://github.com/susanathey/causalTree/blob/forestCode/test/test_causalTree.R
- In addition, this file tests out causalForest, propensityForest, and shows how to construct a propensityTree:
    - https://github.com/susanathey/causalTree/blob/forestCode/test/test_causalForest.R

For gradient.forest:

- To install:
    - install_github("swager/gradient.forest")
    - OR, on Windows, it is easiest to install from a zip file that will be posted on Canvas. You need rtools installed as well (see above). If you have trouble, let me know.
- This has overall tests (the testthat function/library used here requires a directory and subdirectory) https://github.com/swager/gradient-forest/tree/master/r-package/gradient.forest/tests
- To see some more examples of how to use instrumental forest, see this:
    - https://github.com/swager/gradient-forest/blob/master/experiments/instrumental_examples/IV_simu_2_with_ci.R

Specific Assignment

For your assignment:

Part I: Heterogeneous Treatment Effects in Observational Studies

- Using the data set from your first homework, test out propensity forest (see the homework 2 tutorial file for an example) as a way to estimate heterogeneous treatment effects. This builds a tree with W~X, and then estimates treatment effects within each leaf. It can be thought of as propensity matching.
- Average up the results from propensity forest to get an ATE estimate; you can think of the propensity tree as a way to do propensity score matching. Briefly compare to the last homework.
- Use the gradient forest package (https://github.com/swager/gradient-forest/blob/master/r-package/gradient.forest/tests/testthat/test_causal_forest.R has test code; see also the problem set 2 tutorial on canvas) as an alternative.
    - Try it first without residualizing—just run with Y, W, X as in your basic data.
    - Second try it after residualizing first. Use a standard random forest or lasso prediction package to estimate the conditional means of the treatment, the control outcomes, and the treated outcomes. Use gradient forest's causal forest routine on the residuals.
    - Compare the ATE to your other estimates.

Part II: Heterogeneous Treatment Effects in Randomized Experiments

- Return to the un-altered randomized experiment. Use random sampling to divide the dataset into three equal size datasets, call them A, B, and C (in the posted problem set 2 tutorial, these are train.1, train.2, and test).
- Building on the initial "training" R script for this class, use LASSO to estimate heterogeneous treatment effects (interactions between w and X's) in Sample A.
    - Choose lambda via cross-validation.
    - Also compare your results to post-selection OLS: take the variables with non-zero coefficients and run an OLS regression of y on the selected coefficients.
    - If you don't get any interactions between x and w selected, try resampling the data. If you still don't find any, you can try reducing the penalty on one or two of the promising interactions (as described in the previous homework assignment). If you still don't get any, try eliminating the penalty on those.
    - Next, take the non-zero selected variables, and repeat the regression in Sample B, Sample C, and the union of Samples B and C. How do the coefficients and confidence intervals compare for your results on Sample A, and the results on Samples B and C?
- Using https://github.com/susanathey/causalTree, use the command honest.causalTree to build and prune an honest Causal Tree, following this syntax to build the tree:
    - A subdirectory "test" in causalTree has sample code that helps with the items below; see also the problem set 2 tutorial on canvas.

    - Create a factor variable for the leaves in sample B and sample C, and run linear regressions that estimate the treatment effect magnitudes and standard errors (see sample code).
- Using the causalForest function from the causalTree package, estimate a causal forest.
    - Visualize your results by creating a fake dataset with covariates set to the median for all except one or two variables, and with a grid on the variables of interest. The code is provided in the tutorial for problem set 2.
- Repeat the exercise using causal.forest from gradient.forest package, using residualization of the Y's as shown in the problem set 2 tutorial.
- Compare your results across methods. Which methods give more heterogeneity? How do the results differ in terms of the average estimates? You can also compare predictions on subgroups.
- One way to compare the methods is to look at the MSE on a test set using Y* (the transformed outcome, discussed in class in the lecture on "Recursive partitioning") as a proxy for the true treatment effect. Define Y* in the test set, and compare the MSE across methods.

Your write-up should include code with output (preferably generated by "knitting" as per the R instructions provided in the tutorial, although sometimes that has bugs), as well as an electronic document (submitted individually) that discusses the results. Try to make your document self-contained by pasting in figures and referring to specific numbers/standard errors in the text where relevant. If you worked with group members on your code, indicate the group members on the assignment, but your write-ups should be done individually, and each member should submit the code/knit file.