GENERIC MACHINE LEARNING INFERENCE ON HETEROGENEOUS TREATMENT EFFECTS: MODIFIED CODE TO IMPLEMENT THE METHOD OF CHERNOZHUKOV, DEMIRER, DUFLO, AND FERNÁNDEZ-VAL (2019) WITH EXTENSIONS

Introduction

The interests of researchers and policymakers often extend beyond a simple average treatment effect (ATE) when evaluating interventions in randomised experiments. Exploring heterogeneous treatment effects, or average treatment effects by subgroups and covariates, can provide useful answers to a variety of important questions. These include identifying which subgroups will benefit the most from an intervention. However, there are a large number of potential ways to create subgroups and choosing these ex-post can lead to overfitting and issues with multiple hypothesis testing. While this can be partially overcome through pre-analysis plans that specify pre-registered subgroups, such a restriction underutilises the experimental data. In order to discover relevant heterogeneity in treatment effects by covariates ex-post, we employ the method proposed in Chernozhukov et al (2019), henceforth referred to as CDDF. The original code written by CDDF is available on Mert Demirer's github page here. This post discusses an extensive list of modifications made to their code as described below. The modified version of the code is attached and any updated versions will be available on the github repository here in the heterogeneous treatment effects folder.

Overview of the Methodology

For this analysis, let us define Y(1) and Y(0) as potential outcomes for the treatment and control groups, respectively, and define Z as a set of covariates.

The main causal functions are the baseline conditional average (BCA):

$$b(Z) = E[Y(0) \mid Z] \quad (1)$$

and the conditional average treatment effect (CATE):

$$s(Z) = E[Y(1) \mid Z] - E[Y(0) \mid Z]. \quad (2)$$

The first stage of this approach is to use machine learning methods to build a proxy predictor for the CATE, S(Z), using a subset of the data for estimation. The second stage uses this estimated function S(Z) to obtain predictions for the other subset of the data which can then be used to develop inference on three objects of interest: i) the best linear predictor (BLP) of s(Z) based on the predictor S(Z), ii) group average treatment effects (GATES) and iii) classification analysis (CLAN). The BLP regression provides an estimate of the ATE as well as a heterogeneity predictor loading parameter which can be used to test the null hypothesis of no heterogeneity in the treatment effects. The GATES rely on sorting the observations into quantiles based on the predicted treatment effect from S(Z). Finally, the CLAN provides average characteristics from the most and least affected groups. We repeat this process through repeated split sampling for estimation and inference by taking median values of target parameters. The attached code will provide coefficient estimates, p-values and confidence intervals from the top two machine learning methods for the BLP, GATES and CLAN as well as plots of the GATES.

Guide to Code

In total, there are three R codes: "gmlhte_ml.R", "gmlhte_an.R" and "ML_Functions.R". The first two are for implementing this approach while the last file is required for the machine learning algorithms. In order to run the R codes, you need data on outcomes, a treatment dummy and a list of covariates; you may also include fixed effects, additional control variables and a cluster variable for standard errors. Once this data has been prepared, begin with the "gmlhte_ml.R" file. The code has been well commented to guide the reader with 15 input blocks which require attention. These include loading the data, identifying necessary parameters and options, assigning variables and specifying the tuning parameters for each machine learning method. The output of the first code is an RData file to be used in the second code file: "gmlhte_an.R". There are only 4 input blocks that require attention which include the number of quantiles, the significance level and other options. The output of the second code are i) a BEST table reporting the best ML methods, ii) a BLP table reporting the ATE and the heterogeneity predictor loading parameter, iii) a GATES table reporting the top and bottom quantile GATES and the difference, iv) two plots for each outcome variable reporting the GATES where the first plot is for all ML methods and the second plot is for the top two and v) an RData file.

Modifications

The original code in R is written by CDDF and has been modified in two key ways. Firstly, the code is split into two files: i) a machine learning part and ii) an analysis part. The code is split to facilitate the changing of parameters. The machine learning part can be very time consuming depending on the size of the data, the number of covariates, the number of outcomes, the number of machine learning methods and other parameters while the analysis part, which amounts to running many linear regressions, is much quicker. Therefore, if researchers want to change any analysis parameters or variables (such as reducing the list of CLANS), they can avoid having to rerun the machine learning part.

Secondly, several additions have been made to the code. This includes:

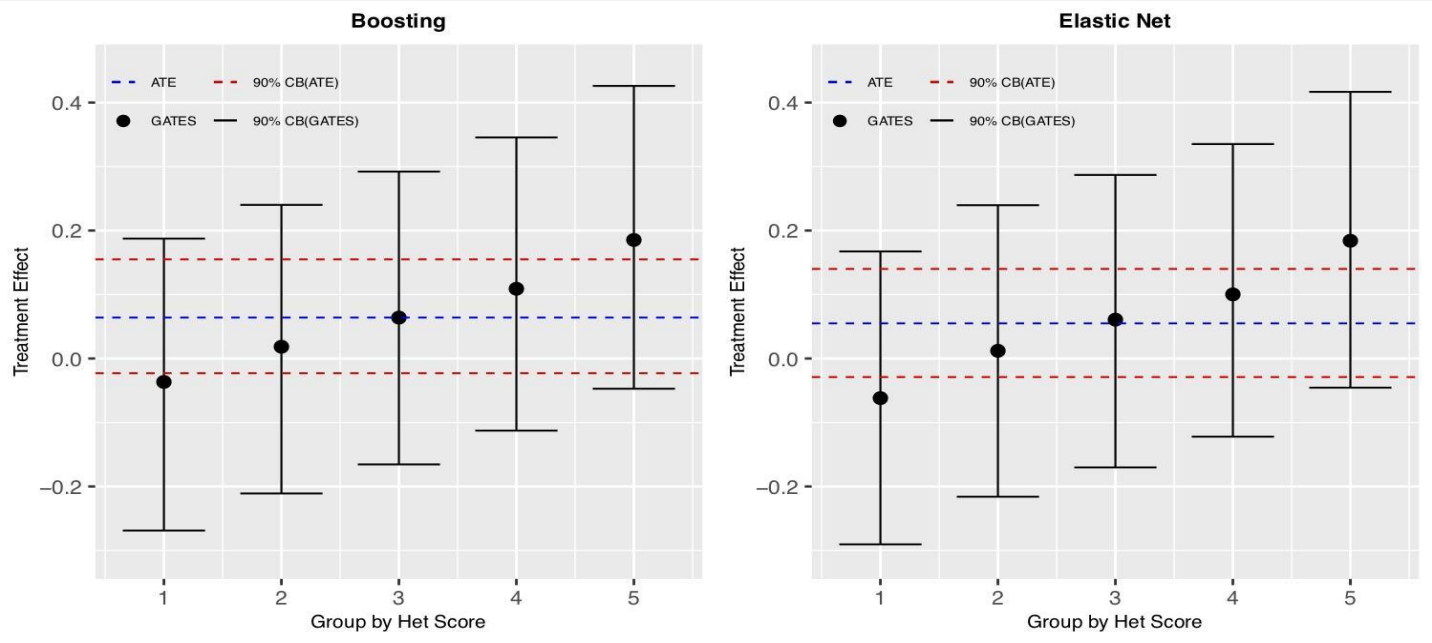
- custom number of quantiles for the GATES,
- option to rearrange (or not) the GATES for monotonicity,
- second fixed effect factor variable, if needed,
- additional control variables for estimating the BLP and GATES,
- option for endogenous stratification (sorting by BCA instead of CATE),
- option to adjust split proportions to $1/n$ and $(n\text{-}1)/n$ for any positive integer n, and vice versa,
- option to adjust the significance level which will also be reflected in the plots (note that the price of splitting uncertainty is reflected in the discounting of the confidence level from $1 - \alpha$ to $1 - 2\alpha$),
- stratified sampling (equal number of control and treatment observations per slit),
- k-fold cross-validation extension.

A more exploratory modification that is worth discussing is the introduction of a k-fold cross-validation option for each repetition. The CDDF approach is analogous to the holdout method in that the sample is split into a subsample for the first stage (estimation) and another subsample which is used for the second stage (inference). In order to use the entire sample for each repetition, while avoiding overfitting, I modify the code to obtain out of sample predictions for the entire sample through a k-fold cross-validation approach such that we do not exclude any observations from the inference stage. As an example, suppose that we split the data in two. The first subsample is used for the first stage and predictions for the second subsample are obtained. The second subsample is then used for the first stage and predictions for the first subsample are obtained. Therefore, out-of-sample predictions for the entire sample are available for the second stage. Please note that this modification is purely explorative at the time of this post.

Interpreting Output

A further description of the first four outputs from the analysis file and their interpretations is provided here. Given that this approach works for any generic machine learning method, it is beneficial to focus on the methods which yield the best outcomes. Two metrics are suggested by CDDF and are presented for each method-outcome pair in the BEST table. The methods which score highest on these metrics are selected and used for the following three outputs. The BLP table includes the ATEs for each method-outcome pair as well as the heterogeneity predictor loading parameter. This last coefficient is equal to 0 if the machine learning is not predictive of the CATE or there is no heterogeneity. If the coefficient is equal to 1 the machine learning is perfectly predictive. Therefore, rejecting the null hypothesis that this coefficient is 0 means that there is both heterogeneity and the machine learning predictions are relevant. The GATES table provides the estimated ATE for the lowest and highest group (sorted by predicted baseline response or treatment effect) and the difference in these coefficients. Finally, the plots depict the full set of GATES for each quantile of the data in addition to the ATE for reference (see Figure 1).

Figure 1 - **GATES Plots**



Relevance to Behavioural Economics

Understanding heterogeneous treatment effects is an important part of evaluating randomised control trials (RCTs) and many papers commonly report treatment effects by sub-groups (Chernozhukov et. al (2019) report that this is true for 40% of RCT papers published in top economic journals since 2006). Given the importance of RCTs for testing behavioural economic theories, combined with the wide range of potential outcomes of interest, a systematic approach to heterogeneous treatment effects such as the one outlined in this post is an essential part of the analysis. This method has been used in the context of credit and saving contracts by Afzal et al. (2019) as well as microcredit availability and immunisation incentives by Chernozhukov et al. (2019).

Paul Brimble, Research Assistant in Development Economics, Blavatnik School of Government
5 November 2020

References

Chernozhukov, Victor, Mert Demirer, Esther Duflo, and Iván Fernández-Val. "Generic Machine Learning Inference on Heterogenous Treatment Effects in Randomized Experiments." *arXiv preprint arXiv:1712.04802v4* (2019).

Afzal, Uzma, Giovanna D'Adda, Marcel Fafchamps, Simon R. Quinn, and Farah Said. *Implicit and Explicit Commitment in Credit and Saving Contracts: A Field Experiment.* No. w25802. National Bureau of Economic Research, 2019.