Supervised Machine Learning I: Prediction

Paul Goldsmith-Pinkham

April 29, 2021

Machine Learning + Causality

- Today, focusing on how to tie machine learning methods into estimation of causal effects
- Most of our ideas revolve around how to think about estimating CATEs – conditional average treatment effects
    - Why is this interesting? Why is knowing CATEs preferable to ATEs?
- Recall that with exhaustively defined CATEs, we can estimate our ATE
    - But, crucially, we could *target* appropriately
    - Well-estimated CATEs help identify better decisions based on decision rules
    - Also good for economic theory!
- But, can be hard to do in a disciplined way

Why can ML be powerful in this space?

- A serious concern in empirical work is specification hunting – looking for significant effects in subgroups, and then telling a story about it

- One solution is pre-analysis plans – tying our hands before the fact about what we will look at

- However, sometimes we would like to let the “data speak”
    - What if we could automate the process for estimating signficant CATEs?

- Machine learning could allow us to estimate these approaches in a standardized way, while using out-of-sample testing to ensure that we are not data mining

The literatures with Machine learning and CATEs

- Today, will talk about two papers/lits:
  - Causal Trees (From Athey and Imbens (2016)
    - More generally in the space of causal partitioning
    - Causal “Forests” by Wager and Athey (2019) as well
  - GATES and CLAN from Chernozhukov et al. (2020)
    - GATES = Sorted Group Average Treatment Effects
    - CLAN = Classification Analysis

- These approaches are similarly focusing on CATEs, but solving a crucial statistical problem in two distinct ways

Machine learning and CATEs

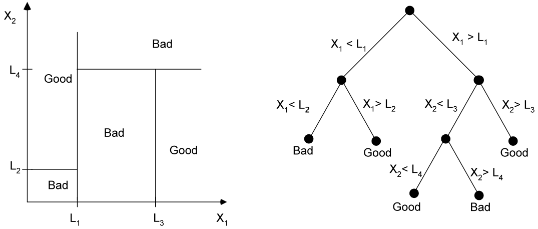
- What is the statistical problem? There are two (related) issues:
  1. Inference: even if we predict or show the effect of a treatment is higher in one subgroup than another, can we say whether this is just due to random variation, or a meaningful difference?
  2. Testing causal inference out-of-sample: Evaluating how “accurate” you are requires knowing your target outcome. E.g. $Y_i - \hat{Y}_i$. But, $\tau_i = Y_i(1) - Y_i(0)$ is fundamentally unknown.

- These issues are in large part solved by additional sample splitting

- Importantly: these approaches do *not* solve the issue of exogeneous variation
  - In most settings, this should be viewed as a setting where we have a randomly varying treatment (e.g. an RCT) and we want to study CATEs
  - However, if we have a good IV, we could study the reduced form quite sensibly!

Causal trees (Athey and Imbens (2016)

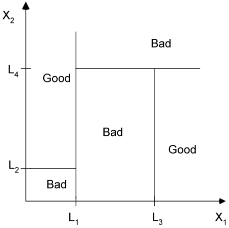
- Necessary notation: Binary treatment, D_i, and covariates (potentially high dimensional) X_i. Outcome Y_i.
- In our discussion, we'll assume completely random assignment of D_i, but it is possible to account for conditioning variables as well using a p-score method
- The key approach will be following the tree-based approach from last class, but with some essential modifications
    - Recall that trees worked by splitting up observations at a given node based on a given characteristic

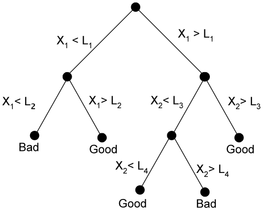


Causal trees (Athey and Imbens (2016))

- Key insight of this paper: when you choose what to split on, you are picking something that is “unusual” relative to the underlying data generating process
    - This induces bias!

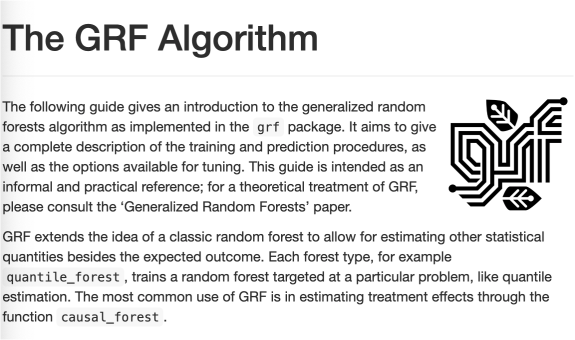
- Hence, the CT approach splits the sample again – first using part of the sample to pick the tree leaves, then testing the effects within the leaves using the left-out sample
    - This gives you three samples: two training, and one true test





Causal trees (Athey and Imbens (2016)

- How to implement? In `R`, there's a very nice package that includes the Random Forest (see Wager and Athey (2019)) approach as well: `https://grf-labs.github.io/grf/`



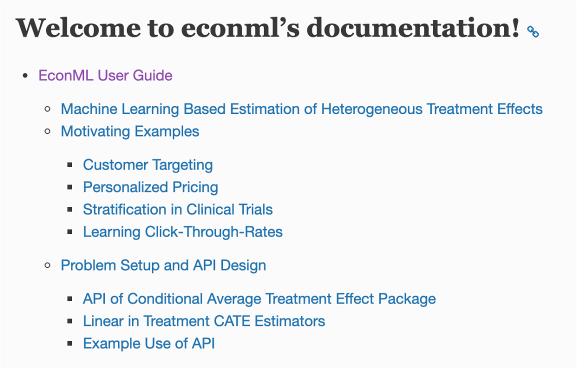
The GRF Algorithm

The following guide gives an introduction to the generalized random forests algorithm as implemented in the `grf` package. It aims to give a complete description of the training and prediction procedures, as well as the options available for tuning. This guide is intended as an informal and practical reference; for a theoretical treatment of GRF, please consult the 'Generalized Random Forests' paper.

GRF extends the idea of a classic random forest to allow for estimating other statistical quantities besides the expected outcome. Each forest type, for example `quantile_forest`, trains a random forest targeted at a particular problem, like quantile estimation. The most common use of GRF is in estimating treatment effects through the function `causal_forest`.

Causal trees (Athey and Imbens (2016)

- How to implement? In `R`, there's a very nice package that includes the Random Forest (see Wager and Athey (2019)) approach as well: `https://grf-labs.github.io/grf/`
- For `Python`, the `econml` package can do this as well (as well as many other approaches): `https://econml.azurewebsites.net/spec/spec.html`



Welcome to econml's documentation!

- EconML User Guide
  - Machine Learning Based Estimation of Heterogeneous Treatment Effects
  - Motivating Examples
    - Customer Targeting
    - Personalized Pricing
    - Stratification in Clinical Trials
    - Learning Click-Through-Rates
  - Problem Setup and API Design
    - API of Conditional Average Treatment Effect Package
    - Linear in Treatment CATE Estimators
    - Example Use of API

Causal trees (Athey and Imbens (2016)

- How to implement? In R, there's a very nice package that includes the Random Forest (see Wager and Athey (2019)) approach as well: https://grf-labs.github.io/grf/

- For Python, the econml package can do this as well (as well as many other approaches): https://econml.azurewebsites.net/spec/spec.html

- Nothing in Stata, sorry

GATES and CLAN

- The causal tree approach is a beautiful approach in solving the bias and infernece issues
- However, the general inference solution does not account for the uncertainty in the binning of the covariates
    - Recall how the method works – by using a split sample to choose the bins, the CATEs within those bins work just as well as any standard regression approach
    - But this fails to account for the fact that these bins may change in different samples
- Chernozhukov et al. (2020) highlight this issue, and propose a much more general approach
    - This approach has more limitations, but at the benefit of being even more general

GATES and CLAN

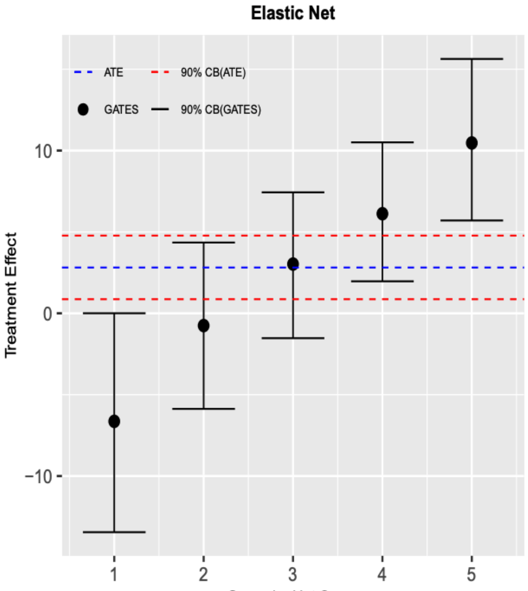
- This approach has a lot of technical details
    - I will not be able to do it justice today

- However, the key concept, as highlighted in the paper, is that instead of trying to identify the CATEs directly, identify the key features of the CATEs instead
    - More precisely, identify how much heterogeneity there is in the underlying estimates
    - Then, figure out the characteristics of those groups with heterogeneous effects

- The key approach starts with the following concept:
    - Randomly split the sample into a main and auxiliary sample
    - In the auxiliary sample, estimate the control mean, $B(X)$ and the treatment effect $\tau(X)$ using any ML method.
    - With these predictions, we will now proceed

GATES and CLAN

- The problem is that $\tau(X)$ is very high-dimensional
- The GATES approach says – what if we grouped the effects into bins G, increasing in effect size.
  - We can talk about the property of these GROUPED average treamtent effects, which average of the high dimensional properties
  - In turns out we can say a lot about that, statistically
- Moreover, we can test for whether these are all the same
  - Harkens back to binscatter and testing for monotonicity!



GATES and CLAN

- The issue is that we still haven't solved for what these groups are
  - Knowledge of heterogeneity doesn't get us very far

- The CLAN approach asks how important characteristics vary by these binned groups

- We can use this to identify bins worth targetting

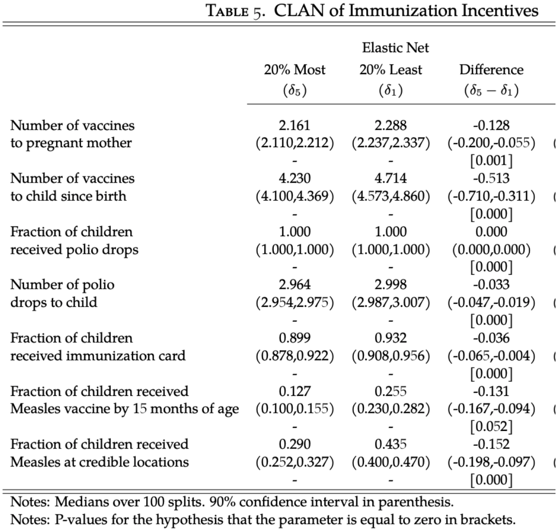


TABLE 5. CLAN of Immunization Incentives

	Elastic Net		
	20% Most (δ_5)	20% Least (δ_1)	Difference $(\delta_5 - \delta_1)$
Number of vaccines to pregnant mother	2.161 (2.110,2.212) -	2.288 (2.237,2.337) -	-0.128 (-0.200,-0.055) [0.001]
Number of vaccines to child since birth	4.230 (4.100,4.369) -	4.714 (4.573,4.860) -	-0.513 (-0.710,-0.311) [0.000]
Fraction of children received polio drops	1.000 (1.000,1.000) -	1.000 (1.000,1.000) -	0.000 (0.000,0.000) [0.000]
Number of polio drops to child	2.964 (2.954,2.975) -	2.998 (2.987,3.007) -	-0.033 (-0.047,-0.019) [0.000]
Fraction of children received immunization card	0.899 (0.878,0.922) -	0.932 (0.908,0.956) -	-0.036 (-0.065,-0.004) [0.000]
Fraction of children received Measles vaccine by 15 months of age	0.127 (0.100,0.155) -	0.255 (0.230,0.282) -	-0.131 (-0.167,-0.094) [0.052]
Fraction of children received Measles at credible locations	0.290 (0.252,0.327) -	0.435 (0.400,0.470) -	-0.152 (-0.198,-0.097) [0.000]

Notes: Medians over 100 splits. 90% confidence interval in parenthesis.
Notes: P-values for the hypothesis that the parameter is equal to zero in brackets.

Implementation in practice

- Chernozhukov et al. (2020) outline the algorithm in detail in the paper
- Best I have found is Max Eber has code here: `https://github.com/maximilianeber/ml-treat`
- Otherwise... good luck! I think this is very doable, if you've done it once, but there's a serious learning curve