

Jacob, Daniel

Working Paper

CATE meets ML: Conditional average treatment effect and machine learning

IRTG 1792 Discussion Paper, No. 2021-005

Provided in Cooperation with:

Humboldt University Berlin, International Research Training Group 1792 "High Dimensional Nonstationary Time Series"

Suggested Citation: Jacob, Daniel (2021) : CATE meets ML: Conditional average treatment effect and machine learning, IRTG 1792 Discussion Paper, No. 2021-005, Humboldt-Universität zu Berlin, International Research Training Group 1792 "High Dimensional Nonstationary Time Series", Berlin

This Version is available at:

<https://hdl.handle.net/10419/233509>

Standard-Nutzungsbedingungen:

Die Dokumente auf EconStor dürfen zu eigenen wissenschaftlichen Zwecken und zum Privatgebrauch gespeichert und kopiert werden.

Sie dürfen die Dokumente nicht für öffentliche oder kommerzielle Zwecke vervielfältigen, öffentlich ausstellen, öffentlich zugänglich machen, vertreiben oder anderweitig nutzen.

Sofern die Verfasser die Dokumente unter Open-Content-Lizenzen (insbesondere CC-Lizenzen) zur Verfügung gestellt haben sollten, gelten abweichend von diesen Nutzungsbedingungen die in der dort genannten Lizenz gewährten Nutzungsrechte.

Terms of use:

Documents in EconStor may be saved and copied for your personal and scholarly purposes.

You are not to copy documents for public or commercial purposes, to exhibit the documents publicly, to make them publicly available on the internet, or to distribute or otherwise use the documents in public.

If the documents have been made available under an Open Content Licence (especially Creative Commons Licences), you may exercise further usage rights as specified in the indicated licence.

IRTG 1792 Discussion Paper 2021-005



CATE Meets ML: Conditional Average Treatment Effect and Machine Learning

Daniel Jacob *



* Humboldt-Universität zu Berlin, Germany

This research was supported by the Deutsche Forschungsgesellschaft through the International Research Training Group 1792 "High Dimensional Nonstationary Time Series".

<http://irtg1792.hu-berlin.de>
ISSN 2568-5619

International Research Training Group 1792

CATE Meets ML

Conditional Average Treatment Effect and Machine Learning

Daniel Jacob

Received: March 31, 2021/ Accepted: date

Abstract For treatment effects - one of the core issues in modern econometric analysis - prediction and estimation are flip-sides of the same coin. As it turns out, machine learning methods are the tool for generalized prediction models. Combined with econometric theory allows us to estimate not only the average but a personalized treatment effect - the conditional average treatment effect (CATE). In this tutorial, we give an overview of novel methods, explain them in detail, and apply them via Quantlets in real data applications. We study the effect that microcredit availability has on the amount of money borrowed and if the 401(k) pension plan eligibility has an impact on net financial assets, as two empirical examples. The presented toolbox of methods contains meta-learners, like the Doubly-Robust, the R-, T- and X-learner, and methods that are specially designed to estimate the CATE like the causal BART and the generalized random forest. In both, the microcredit and the 401(k) example, we find a positive treatment effect for all observations but diverse evidence of treatment effect heterogeneity. An additional simulation study, where the true treatment effect is known, allows us to compare the different methods and to observe patterns and similarities.

Keywords Causal Inference · CATE · Machine Learning · Tutorial

Financial support of the European Union's Horizon 2020 research and innovation program "FIN-TECH: A Financial supervision and Technology compliance training programme" under the grant agreement No 825215 (Topic: ICT-35-2018, Type of action: CSA), the European Cooperation in Science & Technology COST Action grant CA19130 - Fintech and Artificial Intelligence in Finance - Towards a transparent financial industry and the Deutsche Forschungsgemeinschaft's IRTG 1792 grant is gratefully acknowledged.

Daniel Jacob
School of Business and Economics
Humboldt-Universität zu Berlin
Blockchain Research Center
Unter den Linden 6, 10099 Berlin, Germany
E-mail: daniel.jacob@hu-berlin.de

1 Introduction

Estimation and prediction of treatment effects are important tasks for every economist and financial econometrician since treatment effects are often the basis for policy and business decisions. As an illustration, let us look at an idea of microcredits, dating back to Muhammad Yunus, a Nobel Price winner, who discovered in 1976 that very small loans could make a disproportional difference to a poor person. Microcredits work as shown in Figure 1. They can increase investments since such credit is easy to get and to pay back. Business activity is hence more flexible and could be improved. Increasing gains from a business could increase the household income and further allow for more savings which can be invested in, for example, education.



Fig. 1: The theory of microcredits.

This specific example was recently applied by [Crépon et al. \(2015\)](#) who studied the setting where certain villages in Morocco get access to microcredit (the treatment group) while others don't (the control group). As economists, one is interested in the effect that microcredit availability has on the amount of loans which could be an indicator of how demanded such microcredits are. Since we observe certain characteristics for each household we can condition on such observed variables to see if there is heterogeneity in the effect from the microcredit. Figure 2 shows an example of what we want to do. The goal is to find subgroups based on characteristics where we believe that the treatment effect is different. As an example, we can partition the households by age and look at young vs. older household members. In both subgroups, we need to make sure that we observe people that are treated and others that did not receive treatment. We can estimate the average treatment effect (ATE) for the young household members by e.g. taking the difference of their mean outcome given treatment status. We repeat this for the subgroup of older households. Recent methods to estimate the ATE using nonparametric methods on the whole sample include target maximum likelihood estimation (TMLE) ([van der Laan, 2010](#)) and double machine learning ([Chernozhukov et al., 2018a](#)). If the data has many covariates (let us say it has high-dimensionality) and if we don't know which specific subgroup we should focus on, as is the case here, we can use methods that are presented in this tutorial. These methods estimate a treatment effect for each observation based on their covariates, the conditional (on covariates) average treatment effect (CATE). In a further step, we can then

look at the heterogeneity and try to link characteristics that are drivers for different treatment effects.

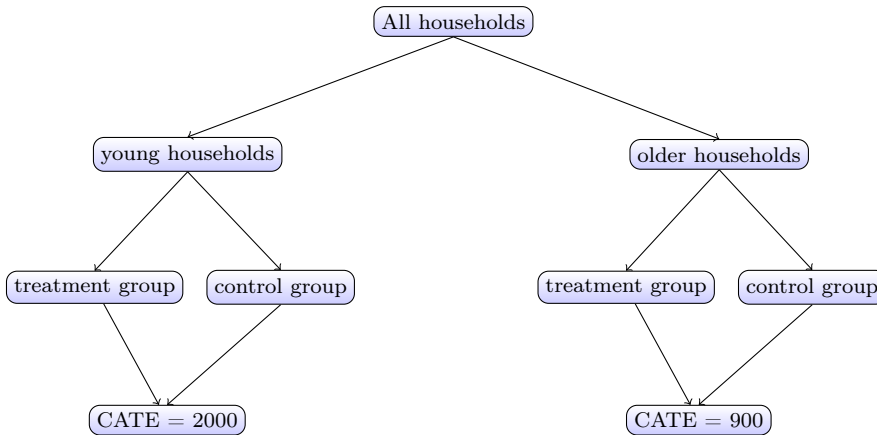


Fig. 2: CATE example for microcredits

High-dimensionality of a dataset does not necessarily mean that one has more covariates than observations by default. However, if we are unsure about the structural form we could include interaction and quadratic terms, and soon the number of dimensions increases. For example, if we have 1000 Observations and 30 covariates then by only including quadratic interactions the amount of covariates increases to 495. Including up to cubic terms leads to a dimension of 5455. If we further assume that only a few covariates are of interest (the so-called approximate sparsity) the task transfers into a selection problem where standard parametric models are limited and we might want to use machine learning (ML) methods. The reason why this is the case is either that we have more covariates than observations or that the functional forms are complex and we don't know which interaction terms to include in a linear model. The simplest approach to get the CATE is to estimate two conditional mean functions, one for the treated observations and one for the non-treated (the control group). For each observation, we can predict the outcome under treatment and control by plugging each observation in both functions. Taking the difference between the two outcomes results in the CATE. Mapping the support of X on Y is a classic regression task for which machine learning methods are well suited to find generalizable predictive patterns. Since we are only interested in getting a good prediction of the conditional mean, we do not need to know the underlying structural form of this function which enables vanilla ML methods to be sufficient. We call such functions, where the parameters are not of immediate interest a nuisance function. While the above example to estimate the CATE is quite simple and intuitive, we will see

that there are more efficient or automated methods to estimate heterogeneous treatment effects.

This tutorial is structured as follows. First, we provide an overview of the potential outcome framework and state the necessary assumptions to interpret our parameter of interest as a causal parameter. We then explain different methods that we consider, methods that are very flexible in the choice of the ML algorithm, and methods that are specifically designed to estimate the CATE and mostly rely on tree-based algorithms. As in classical ML, we make use of sample splitting to limit overfitting and allowing for less restrictive assumptions on the nuisance functions. We cover explanations on why and how to do sample splitting and cross-validation. Next, we investigate two empirical datasets, the microcredit example, and the 401(k) pension plan survey. Last, we include a simulation study where we generate the true treatment effect. This allows us to directly compare all different methods in terms of accuracy. Whenever possible we provide and link to Quantlets [🔗](#) that are ready-to-use code snippets to implement the discussed methods (the Quantlets are all written in **R**). The files are not only a replication code for the empirical analysis and the simulation study but contain functions to implement novel methods that aim to estimate the CATE directly. During this tutorial, we will use the terms model, method, and algorithm interchangeably.

Figure 3 gives an idea of how a causal structure may look like. In the first graph, only the treatment has an impact on the outcome while the second graph also includes covariates that might make the treatment effect dependent on some characteristics. The same is true for the third graph but now the covariates also influence the treatment probability. We say that such a setting is from an observational study since the researcher has no control of the treatment assignment. The first two settings can be seen as a randomized controlled trial (RCT) but only in the second and the third one can we hope to observe treatment heterogeneity and hence estimate the CATE.

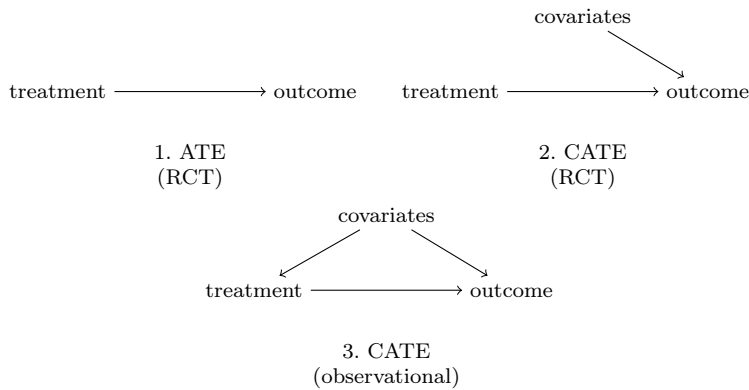


Fig. 3: Simple causal diagrams - from ATE to CATE.

2 Methods

Let us start with an introduction of the potential outcome framework for which we use the following notations: Each observation has two potential outcomes, Y^1 and Y^0 of which we only observe one, namely the former if someone was treated or the latter if not. We denote this by the binary treatment indicator $D \in \{0, 1\}$ and denote observed covariates $X \in \mathbb{R}^p$. To interpret the estimated parameter as a causal relationship, the following assumptions are needed; see, for example, [Rubin \(1980\)](#):

1. Conditional independence (or conditional ignorability/exogeneity or conditional unconfoundedness):

$$(Y_i^1, Y_i^0) \perp\!\!\!\perp D_i | X_i.$$

2. Stable Unit Treatment Value Assumption (SUTVA) (or counterfactual consistency):

$$Y_i = Y_i^0 + D_i(Y_i^1 - Y_i^0).$$

3. Overlap Assumption (or common support or positivity):

$$\begin{aligned} \forall x \in \text{supp}(X_i), \quad 0 < P(D_i = 1 | X_i = x) < 1, \\ P(D_i = 1 | X_i = x) \stackrel{\text{def}}{=} e(x). \end{aligned} \tag{1}$$

4. Exogeneity of covariates:

$$X_i^1 = X_i^0.$$

Assumption 1 together with Assumption 4 is very natural since they state that the treatment assignment is independent of the two potential outcomes and that the covariates are not affected by the treatment. Assumption 2 ensures that there is no interference, no spillover effects, and no hidden variation between treated and non-treated observations. Assumption 3 states that no subpopulation defined by $X_i = x$ is entirely located in the treatment or control group, hence the treatment probability needs to be bounded away from zero and one. Equation (1) is called the propensity score.

Now we define the conditional expectation of the outcome for the treatment or control group as

$$\mu_d(x) = \mathbb{E}[Y_i | X_i = x, D_i = d] \quad \text{with} \quad D \in \{0, 1\}.$$

If we don't use any subscript, we refer to this function as the general conditional expectation.

Our parameter of interest is the CATE ($\tau(x)$), which is formally defined as:

$$\tau(x) = \mathbb{E}[Y_i^1 - Y_i^0 \mid X_i = x] = \mu_1(x) - \mu_0(x). \quad (2)$$

Equation 3 shows how the two conditional mean functions can represent the two potential outcomes and hence, by taking the difference, lead to the CATE.








$$\begin{aligned} \tau(x) &= \mu_1(x) - \mu_0(x) \\ &= \mathbb{E}[Y_i \mid D_i = 1, X_i = x] - \mathbb{E}[Y_i \mid D_i = 0, X_i = x] \\ &= \mathbb{E}[Y_i^1 \mid D_i = 1, X_i = x] - \mathbb{E}[Y_i^0 \mid D_i = 0, X_i = x] \\ &= \mathbb{E}[Y_i^1 \mid X_i = x] - \mathbb{E}[Y_i^0 \mid X_i = x] \\ &= \mathbb{E}[Y_i^1 - Y_i^0 \mid X_i = x] \end{aligned} \quad (3)$$

This estimator is of special interest in many areas like medicine or policy actions since it tells us if there are differences in the treatment effect in the population and how big these differences are. It could be, for example, that the average treatment effect of a policy is +2, containing half of the people with a treatment effect of +6 and the other half of -2. Instead of treating everyone, we should only treat people that have a positive effect from the policy (if positive means better). If this is not possible, let us say due to laws or ethical reasons, the policy should not be implemented at all. The CATE will tell us exactly the distribution of the effects and, at best, allows us to identify subgroups. To estimate the CATE we are not primarily interested in the coefficient from regressing X on Y , nor are we interested in the coefficients from the propensity score model. What we want instead is to have a good approximation of the function and hence good estimates from e.g. $\mu^1(x)$ and $\mu^0(x)$. This is why ML methods are so suited for the job.

When reviewing recently proposed methods for the estimation of the CATE, we can categorize them into two groups. The first group contains methods that are build on off the shelf machine learning methods (such as the lasso, random forest (RF), Bayesian Adaptive Regression Trees (BART), boosting methods or neural networks) Since the base learners are not designed to estimate the CATE directly the literature calls them meta-learners, or generic ML algorithms. The second group of methods alters existing machine learning methods in a way that they can be used to estimate the CATE directly (examples are causal boosting by Powers et al. (2018), causal forest by Athey et al. (2019) or Bayesian regression tree models for causal inference by Hahn et al. (2020)). See Künzel et al. (2019) for a comparison between meta-learners like the S-, T-, and X-learner as well as the causal forest in a simulation study. Knaus et al. (2020) compare the meta-learners like the inverse probability weighting (IPW) estimator, doubly-robust (DR), modified covariate method (MCM), R-learner, and different versions of the causal forest in an empirical Monte Carlo study

while Nie and Wager (2017) compare their R-learner with the S-, T-, X- and U-learner as well as causal boosting. Regarding the base learners (the ML methods), Künzel et al. (2019) use a random forest (RF) and Bayesian additive regression trees (BART). Knaus et al. (2020) use RF and the lasso while Nie and Wager (2017) use boosting and the lasso for the estimation of the nuisance functions. In high dimension, the use of machine learning methods, such as boosting or random forests to estimate the propensity score, works quite well as McCaffrey et al. (2004) and Wyss et al. (2014) show. The estimation of probabilities given a large set of covariates is nothing less than a prediction problem in where ML methods are superior. Table 1 lists all the methods by category, including links to the Quantlets. The references refer to recent papers that use these methods and provide theoretical properties as well as implementation examples.

Table 1: Methods to estimate CATE

Category	Method	Reference	Quantlet
Meta-Learner	DR-learner	Kennedy (2020)	 <i>DR</i>
	R-learner	Nie and Wager (2017)	 <i>R</i>
	T-learner	Hansotia and Rukstales (2002)	 <i>T</i>
	X-learner	Künzel et al. (2019)	 <i>X</i>
Modified ML Methods	Causal Forest	Athey et al. (2019)	 <i>GRF</i>
	Causal Boosting	Powers et al. (2018)	 <i>CB</i>
	Causal BART	Hahn et al. (2020)	 <i>CBART</i>

2.1 Meta-Learners

In the following, we briefly describe the considered meta-learners. Except for the T-learner, all other methods generate a pseudo-outcome in the first step which can be seen as an approximation of the conditional average treatment effect. The last step regresses this function on the covariates to get the final estimate and to make predictions on new observations. The reason is that data after a treatment assignment includes the outcome, covariates, and the treatment assignment variable. If we want to classify new observations we only observe the covariates. Hence we need a model that maps the covariates on the estimated treatment effect. The DR-, R- and X-learner also require to estimate the propensity score as an additional nuisance function to further control for any confounding bias. Currently, R-packages are available for the R-, S-, T-, U-, and X-learner (`install_github("xn timer/learner")`) and the M-, S-, T-, and X-learner (`install_github("soerenkuenzel/causalToolbox")`).

Causal analysis via the potential outcome framework and causal graph theory for Python can be found in [Sharma et al. \(2019\)](#). For heterogeneous treatment effect analysis via machine learning in Python see [EconML \(2019\)](#).

Single (S-learner) and two-model learner (T-learner):

Let us first start with a very simple and intuitive method, the T-learner. It is a two-step approach where the conditional mean functions $\mu_1(x) = \mathbb{E}[Y^1|X_i = x]$ and $\mu_0(x) = \mathbb{E}[Y^0|X_i = x]$ are estimated separately with any generic machine learning algorithm. The difference between the two functions results in the CATE as shown in Table 2 and as seen in equation 3. One problem with the T-learner is that it aims to minimize the mean squared error for each separate function rather than to minimize the mean squared error of the treatment effect. By splitting the sample in two groups there is only information on one group. This might be problematic if the two functions shrink different covariates which are actually important in both groups. This is especially the case in a RCT. See, for example, [Künzel et al. \(2019\)](#); [Kennedy \(2020\)](#) for settings when the T-learner is not the optimal choice. An alternative is to model only one function and taking the treatment assignment into this function. This approach is called the S-learner. See for example, [Hill \(2011\)](#) and [Foster et al. \(2011\)](#) for early examples of proposing the S-learner.

Algorithm 1: S-learner

- Input:** $Z_i = \{Y_i, D_i, X_i\}_{i \in N}$
- 1 Split sample Z into subsets $S_t = z_1, \dots, z_{\lfloor n/2 \rfloor}$ and $S_v \cup S_t$
 - 2 **regress** $Y_i = \hat{\mu}(X_i, D_i) + \hat{U}_i$, with $i \in S_t$
 - 3 **estimate** $\hat{Y}_i^0 = \hat{\mu}(X_i, D = 0)$, with $i \in S_v$
 - 4 **estimate** $\hat{Y}_i^1 = \hat{\mu}(X_i, D = 1)$, with $i \in S_v$
 - 5 **create** $\hat{\tau}(X_i) = \hat{\mu}(X_i, D = 1) - \hat{\mu}(X_i, D = 0)$
-

Algorithm 2: T-learner

- Input:** $Z_i = \{Y_i, D_i, X_i\}_{i \in N}$
- 1 Split sample Z into subsets $S_t = z_1, \dots, z_{\lfloor n/2 \rfloor}$ and $S_v \cup S_t$
 - 2 **regress** $Y_i^0 = \hat{\mu}_0(X_i^0) + \hat{U}_i^0$, with $i \in S_t | D = 0$
 - 3 **regress** $Y_i^1 = \hat{\mu}_1(X_i^1) + \hat{U}_i^1$, with $i \in S_t | D = 1$
 - 4 **estimate** $\hat{Y}_i^0 = \hat{\mu}_0(X_i)$, with $i \in S_v$
 - 5 **estimate** $\hat{Y}_i^1 = \hat{\mu}_1(X_i)$, with $i \in S_v$
 - 6 **create** $\hat{\tau}(X_i) = \hat{\mu}_1(X_i) - \hat{\mu}_0(X_i)$
-

Doubly-Robust learner (DR-learner):

A more efficient method than the T-learner can be the DR-learner. It builds on the T-learner and adds a version of the inverse probability weighting (IPW) scheme on the residuals of both regression functions $\{Y^d - \hat{\mu}_d(x)\}$. We can think of it as combining two different models and hence avoid drawbacks like

the minimization goal from the T-learner and a potentially high variance from an IPW model when some propensity scores are small. The doubly-robust learner takes its name from a double robustness property which states that the estimator remains consistent if either the propensity score model or the conditional outcome model is correctly specified. This is at least true for the average treatment effect (Lunceford and Davidian, 2004). Recently, this estimator has gained popularity to estimate the CATE, especially in high-dimensional settings. See, for example, the work by Fan et al. (2019) and Zimmert and Lechner (2019). Most recently, Kennedy (2020) find that for estimating the CATE, the finite-sample error-bound from the DR-learner at most deviates from an oracle error rate by the product of the mean squared error of the propensity score and the conditional mean estimator.

Algorithm 3: DR-learner

Input: $Z_i = \{Y_i, D_i, X_i\}_{i \in N}$

- 1 Split sample Z into subsets $S_t = z_1, \dots, z_{\lfloor n/2 \rfloor}$ and $S_v \cup S_t$
- 2 **for** b in $\{1, \dots, B\}$ **do**
- 3 Split sample S_t into K random subsets
- 4 **for** k in $\{1, \dots, K\}$ **do**
- 5 **assign** Sample $S_a = S_t \cup S_k$ and $S_m = S_k$
- 6 **regress** $D_i = \hat{e}(X_i) + \hat{V}_i$, with $i \in S_a$
- 7 **regress** $Y_i^0 = \hat{\mu}_0(X_i^0) + \hat{U}_i^0$, with $i \in S_a | D = 0$
- 8 **regress** $Y_i^1 = \hat{\mu}_1(X_i^1) + \hat{U}_i^1$, with $i \in S_a | D = 1$
- 9 **estimate** $\hat{D}_i = \hat{e}_0(X_i)$, with $i \in S_m$
- 10 **estimate** $\hat{Y}_i^0 = \hat{\mu}_0(X_i)$, with $i \in S_m$
- 11 **estimate** $\hat{Y}_i^1 = \hat{\mu}_1(X_i)$, with $i \in S_m$
- 12 **create** $\hat{\psi}_{DR} = \hat{\mu}_1(x) - \hat{\mu}_0(x) + \frac{D\{Y - \hat{\mu}_1(x)\}}{\hat{e}(x)} - \frac{(1-D)\{Y - \hat{\mu}_0(x)\}}{(1 - \hat{e}(x))}$
- 13 **regress** $\hat{\psi}_i = \hat{t}_{DR}(X_i) + W_i$, with $i \in S_m$
- 14 **estimate** $\hat{\tau}_k(X_i) = \hat{t}_{DR}(X_i)$, with $i \in S_v$
- 15 **end**
- 16 **cross-fit** $\hat{\tau}_b(X_i) = \frac{1}{K} \sum_{k=1}^K \hat{\tau}_k(X_i)$
- 17 **end**
- 18 **create** $\tilde{\tau}(X_i) = \text{median}\{\hat{\tau}_b(X_i)\}_{b=1}^B$

R-learner:

The orthogonal-learner makes use of the idea of orthogonalization to cancel out any selection bias that may arise in observational studies from observed covariates. Here, the residuals from the regression of Y on X are regressed on the residuals from the regression of D on X and weighted by the squared residuals, $\{D - \hat{e}(x)\}^2$. This is similar to the double machine learning approach from Chernozhukov et al. (2018a) where their estimator of interest is the ATE. Nie and Wager (2017) develop a general class of two-step algorithms for the estimation of the CATE. The R-learner, as from residualized and a homage to Robinson 1988, makes explicit use of machine learning methods. Achieving

Neyman orthogonality using a residuals-on-residuals (or debiasing) approach has a long history in econometrics (see the Frisch–Waugh–Lovell theorem from the 1930s for linear regression) and mainly builds on the work by [Robinson \(1988\)](#) who replaces the linear parts by non-parametric kernel regression. The CATE from the R-learner is obtained by the following minimization task:

$$\hat{\tau}(\cdot) = \operatorname{argmin}_{\tau} \left\{ \frac{1}{n} \sum_{i=1}^n [\{Y_i - \hat{\mu}^{(-i)}(X_i)\} - \{W_i - \hat{e}^{(-i)}(X_i)\} \tau(X_i)]^2 + \Lambda_n \{\tau(\cdot)\} \right\}. \quad (4)$$

The superscript $(-i)$ indicates the sample splitting. The conditional mean functions are trained without the i -th observations and evaluated only for i . We will explain certain sample splitting procedures later. The term $\Lambda_n \{\tau(\cdot)\}$ can be interpreted as a regularizer on the complexity of the $\tau(\cdot)$ function. In practice, this regularization term could be explicitly given as in penalized regression or implicitly introduced, e.g., as provided by a carefully designed deep neural network.

Algorithm 4: R-learner

Input: $Z_i = \{Y_i, D_i, X_i\}_{i \in N}$

- 1 Split sample Z into subsets $S_t = z_1, \dots, z_{\lfloor n/2 \rfloor}$ and $S_v \cup S_t$
- 2 **for** b in $\{1, \dots, B\}$ **do**
- 3 Split sample S_t into K random subsets
- 4 **for** k in $\{1, \dots, K\}$ **do**
- 5 **assign** Sample $S_a = S_t \cup S_k$ and $S_m = S_k$
- 6 **regress** $D_i = \hat{e}(X_i) + \hat{V}_i$, with $i \in S_a$
- 7 **regress** $Y_i = \hat{\mu}(X_i) + \hat{U}_i$, with $i \in S_a$
- 8 **estimate** $\hat{D}_i = \hat{e}_0(X_i)$, with $i \in S_m$
- 9 **estimate** $\hat{Y}_i = \hat{\mu}(X_i)$, with $i \in S_m$
- 10 **create** $\hat{\psi}_R = \frac{(Y_i - \hat{\mu}(X_i))}{(D_i - \hat{e}(X_i))}$
- 11 **regress** $\hat{\psi}_i = \hat{t}_R(X_i) + W_i$ and weights $(D_i - \hat{e}(X_i))$, with $i \in S_m$
- 12 **estimate** $\hat{\tau}_k(X_i) = \hat{t}_R(X_i)$, with $i \in S_v$
- 13 **end**
- 14 **cross-fit** $\hat{\tau}_b(X_i) = \frac{1}{K} \sum_{k=1}^K \hat{\tau}_k(X_i)$
- 15 **end**
- 16 **create** $\tilde{\tau}(X_i) = \operatorname{median}\{\hat{\tau}_b(X_i)\}_{b=1}^B$

X-learner:

[Künzel et al. \(2019\)](#) propose the X-learner which estimates a treatment effect separately for the control and the treatment group. This might be especially helpful in situations where the proportion of the two groups is highly imbalanced. The X-learner has several steps. The first step is identical to the T-learner, namely estimating the two conditional mean functions. In the

second step, we take the difference of the groups separately and from the observed outcome rather than from the two estimated functions (as we would do when applying the T-learner). This results in two imputed treatment effects ($\hat{\psi}_X^1 \stackrel{\text{def}}{=} Y^1 - \hat{\mu}_0(x^1)$ and $\hat{\psi}_X^0 \stackrel{\text{def}}{=} \hat{\mu}_1(x^0) - Y^0$) which are now used in a third step to regress them individually on the covariates to obtain $\hat{\tau}_0(x)$ (the CATE for the control group) and $\hat{\tau}_1(x)$ (the CATE for the treatment group). The final estimator combines the two estimators plus some weights, $g(x)$:

$$\hat{\tau}(x) = g(x)\hat{\tau}_0(x) + \{1 - g(x)\}\hat{\tau}_1(x).$$

The weights can, for example, be set to $1 - \hat{e}(x)$ for the treatment group and $\hat{e}(x)$ for the control group estimate, respectively.

Algorithm 5: X-learner

Input: $Z_i = \{Y_i, D_i, X_i\}_{i \in N}$

- 1 Split sample Z into subsets $S_t = z_1, \dots, z_{\lfloor n/2 \rfloor}$ and $S_v \cup S_t$
- 2 **for** b *in* $\{1, \dots, B\}$ **do**
- 3 Split sample S_t into K random subsets
- 4 **for** k *in* $\{1, \dots, K\}$ **do**
- 5 **assign** Sample $S_a = S_t \cup S_k$ and $S_m = S_k$
- 6 **regress** $D_i = \hat{e}(X_i) + \hat{V}_i$, with $i \in S_a$
- 7 **regress** $Y_i^0 = \hat{\mu}_0(X_i^0) + \hat{U}_i^0$, with $i \in S_a | D = 0$
- 8 **regress** $Y_i^1 = \hat{\mu}_1(X_i^1) + \hat{U}_i^1$, with $i \in S_a | D = 1$
- 9 **estimate** $\hat{D}_i = \hat{e}_0(X_i)$, with $i \in S_v$
- 10 **estimate** $\hat{Y}_i^0 = \hat{\mu}_0(X_i)$, with $i \in S_m$
- 11 **estimate** $\hat{Y}_i^1 = \hat{\mu}_1(X_i)$, with $i \in S_m$
- 12 **create** $\hat{\psi}_X^1 \stackrel{\text{def}}{=} Y^1 - \hat{\mu}_0(x^1)$
- 13 **create** $\hat{\psi}_X^0 \stackrel{\text{def}}{=} \hat{\mu}_1(x^0) - Y^0$
- 14 **regress** $\hat{\psi}_X^1 = \hat{t}^1(X_i) + W_i^1$, with $i \in S_m$
- 15 **regress** $\hat{\psi}_X^0 = \hat{t}^0(X_i) + W_i^0$, with $i \in S_m$
- 16 **estimate** $\hat{\tau}_k^1(X_i) = \hat{t}^1(X_i)$, with $i \in S_v$
- 17 **estimate** $\hat{\tau}_k^0(X_i) = \hat{t}^0(X_i)$, with $i \in S_v$
- 18 **average** $\hat{\tau}_k(X_i) = \hat{e}(X_i)\hat{\tau}_k^0 + (1 - \hat{e}(X_i))\hat{\tau}_k^1$
- 19 **end**
- 20 **cross-fit** $\hat{\tau}_b(X_i) = \frac{1}{K} \sum_{k=1}^K \hat{\tau}_k(X_i)$
- 21 **end**
- 22 **create** $\tilde{\tau}(X_i) = \text{median}\{\hat{\tau}_b(X_i)\}_{b=1}^B$

Summary of meta-learners:

We summarise the considered meta-learners in Table 2 where $\hat{\psi}$ states the pseudo-outcome or estimator for each of the learners. The last column counts the number of nuisance functions needed to estimate the pseudo-outcome or estimator. In brackets, we state the total number of models needed to get the final CATE estimate. Note that the X-learner is regressed only for the treated

observations and again only for the observations in the control group. This is why we need two more additional models for the final estimate.

Table 2: Summary of meta-learners

Method	Estimator/Pseudo-outcome	Weights (w_i)	# of Models
S-learner	$\hat{\psi}_S = \hat{\mu}(x, d = 1) - \hat{\mu}(x, d = 0)$	1	1 (2)
T-learner	$\hat{\psi}_T = \hat{\mu}_1(x) - \hat{\mu}_0(x)$	1	2 (3)
DR-learner	$\hat{\psi}_{DR} = \hat{\psi}_T + \frac{D \{Y - \hat{\mu}_1(x)\}}{(1 - D) \{Y - \hat{\mu}_0(x)\}}$	1	3 (4)
R-learner	$\hat{\psi}_R = \frac{(1 - \hat{e}(x)) \{Y - \hat{\mu}(x)\}}{\{D - \hat{e}(x)\}}$	$\{D - \hat{e}(x)\}^2$	2 (3)
X-learner	$\hat{\psi}_X^1 \stackrel{\text{def}}{=} Y^1 - \hat{\mu}_0(x^1)$ $\hat{\psi}_X^0 \stackrel{\text{def}}{=} \hat{\mu}_1(x^0) - Y^0$	1	3 (5)

Notes: Considered meta-learners that estimate the CATE. # of Models counts the number of nuisance functions to estimate the pseudo-outcome. Numbers in brackets count the total number of models to train to get the final CATE estimate.

The estimators from Table 2 can be represented as a weighted minimization problem which solves the following:

$$\min_{\tau} \left\{ N^{-1} \sum_{i=1}^N w_i \{ \hat{\psi}_i - \tau(x) \}^2 \right\}.$$

The choice of ML algorithms for meta-learners:

The accuracy of the CATE estimation depends on the accuracy of the nuisance functions and hence on the choice of the ML method. To minimize the dependence of the ML methods on our estimates, we do not assign specific machine learning methods for the estimation but consider a range of different popular methods. To choose which ML method to use for each nuisance function as well as for any additional functions, we use a stacking method. In such a setting, not only one ML method may be chosen but an ensemble of methods that are stacked together with different weights. We use the SuperLearner package as proposed by Polley et al. (2011). It also enables us to choose different models for each nuisance function and setting. The package offers a general class of prediction methods to be considered by the ensemble. From the 42 different algorithms, we select the lasso (`glmnet`) and random forest (`ranger`) for our analysis. This list can of course be extended to e.g. Gradient boosted trees (`xgboost`) and or Neural Network (`nnet`). Note that the R-learner needs to include weights in the algorithm, so we need to make sure that the ML methods we use have this possibility included.

We use 10-fold cross-validation to estimate the performance of all machine learning models. Cross-validation is a resampling procedure used to evaluate ML models on a finite data sample. Depending on the ML model, the data can be fit perfectly and hence produce a high variance (overfitting). This is, however, on the training sample and the model can behave poorly on unseen data. Hence, we have to validate our models. We could use a part of the data for validation. Since there is never enough data, removing a part of it poses a potential for underfitting (we might lose trends in the data or important patterns). What we require instead, is a method that provides enough data for training the model and also leaves enough data for validation. K-fold cross-validation does exactly that. This approach involves randomly splitting the set of observations into k groups, or folds, of approximately equal size. The model is fit on folds 2 to k while the first fold is used as a validation set. It is also important that any preparation of the data before fitting the model occur on the training sample that is used for cross-validation within the loop rather than on the broader data sample. This also applies to any tuning of hyperparameters e.g. the number of trees, the minimum observations within a node, learning rates, or shrinkage parameters. There is no formal rule for the choice of k but usually, it is set to 5 or 10. These values have been shown empirically to yield test error rate estimates that suffer neither from excessively high bias nor from very high variance. The reason is the following: The larger k , the smaller the difference in size between the (original) training set and the resampling subset ($k - 1$ folds). As this difference decreases, the bias of the technique becomes smaller. This means that the bias is smaller for $k = 10$ than for $k = 5$. A special case of cross-validation is the so-called leave-on-out cross-validation (LOOCV). In this case, k is set to the sample size and only one observation is the validation sample. In all procedures, the k resampled estimates of performance are summarized (e.g. by the mean and the standard error).

Since we apply multiple models to estimate the nuisance functions we create a weighted average among all models. Using stacking, we can find the optimal combination of a collection of prediction algorithms or even different settings within one model. In other words, we build a linear model that uses the outcome variable of the validation set as the dependent variable and all different base learners as the input variables. For the random forest we set the following tuning parameters: `n.trees=1000`, `min.node.size=10`.

2.2 Modified ML-methods

We now describe some methods that modify existing ML methods to estimate the CATE directly. In contrast to meta-learners that are flexible in the choice of the ML algorithm, these methods use a specific ML method (mostly tree-based algorithms). Packages or code in R are available for the causal forest (`grf`), the causal Boosting (<https://github.com/saberpowers/causalLearning>) and the causal BART (`install_github("vdorie/bartCause")`). Since causal

boosting is computationally expensive we do not consider this method in our analysis.

Causal Forest:

The causal forest method, part of the generalized random forest (GRF) by [Athey et al. \(2019\)](#) builds on a random forest algorithm to find neighborhoods in the covariate space. These neighborhoods are built by recursive splitting the covariates into subgroups while the criterion to do so is based on heterogeneity in treatment effects. The idea is to find leaves where the treatment effect is constant but different from other leaves. If we knew that $\tau(x)$ were constant over some neighbourhood $N(x)$, we could solve a partially linear model over $N(x)$ using the residual-on-residual approach (see e.g. [Robinson \(1988\)](#)): First we estimate $e(x) = \mathbb{E}[D_i|X_i = x]$ and second, $\mu(x) = \mathbb{E}[Y_i|X_i = x]$. We can use any non-parametric method like the lasso, random forests, boosting methods, neural networks and others. The final step is to estimate $\tau(x)$ over the neighbourhood $N(x)$:

$$\hat{\tau}(x) = \frac{\sum_{\{i: X_i \in N(x)\}} \{Y_i - \hat{\mu}(X_i)\} \{W_i - \hat{e}(X_i)\}}{\sum_{\{i: X_i \in N(x)\}} \{W_i - \hat{e}(X_i)\}^2}. \quad (5)$$

Note that this approach looks similar to the R-learner. [Chernozhukov et al. \(2018a\)](#) showed that when using any of the aforementioned ML methods to estimate the nuisance functions and then use the residual-on-residual approach to estimate the average treatment effect the following regularity condition holds:

Given that,

$$\mathbb{E} \left[\{\mu(X_i) - \hat{\mu}(X_i)\}^2 \right]^{\frac{1}{2}} \ll \frac{1}{n^{1/4}}, \quad \mathbb{E} \left[\{e(X_i) - \hat{e}(X_i)\}^2 \right]^{\frac{1}{2}} \ll \frac{1}{n^{1/4}}, \quad (6)$$

we get a central limit theorem such that $\sqrt{n}(\hat{\tau} - \tau) \Rightarrow \mathcal{N}(0, V)$. The treatment effect is the above setting, however, has to be constant. We can assume that with heterogeneous treatment effects there are subgroups such that the constant effect assumption holds. The question of how to find such accurate subgroups is exactly where the (causal) random forest comes into play. To create leaves that consist of observations with the same (average) treatment effect the splitting criterion has to rely on maximizing the heterogeneity in treatment effects between leaves (similar to maximize the variance between the leaves). Here we use again the method from equation (5). In observational studies where self-selection into treatment is present, the first splits might not be a good representation of the treatment effect rather than differences due to confounding variables. To overcome this problem [Athey et al. \(2019\)](#) suggest applying local-centering. This means that we use the residuals of the outcome and treatment variable as data instead of the original values. Therefore one has to train two nuisance functions beforehand to predict the conditional mean which is used to create the residuals. While machine learning methods rely on sample splitting to avoid overfitting the causal random forest integrates this

via an honesty condition. A tree is honest if, for each training sample i it only uses the response Y_i to estimate the within-leaf treatment effect or to decide where to place the split, but not both.

So far we have looked how a single tree is build and how the final treatment effect can be estimated. To extend this procedure to multiple trees lets view a forest as a weighting function:

$$\hat{\mu}(x) = B^{-1} \sum_{b=1}^B \sum_{i=1}^n Y_i \frac{1\{X_i \in L_b(x)\}}{|L_b(x)|} = \sum_{i=1}^n Y_i \underbrace{B^{-1} \sum_{b=1}^B \frac{1\{X_i \in L_b(x)\}}{|L_b(x)|}}_{\alpha_i(x)}. \quad (7)$$

Instead of seeing a forest as a double average over observations within a leaf and the B single trees, we can integrate the first sum to be a weighted average over all X_i that fall into the leaf $L_b(x)$ and divide by the total number of observations within the leaf ($|L_b(x)|$). This weighted average tells us how often Y_i falls into a certain leaf and hence the weight that we have to apply to control for the different proportions. The weights can be represented as $\alpha_i(x)$. We can now use these weights to weight each observation in a generalized method of moments estimator where we apply a linear model, regressing the residuals of D_i on the residuals of Y_i and weight by α_i . This is how we get the CATE using a random forest. The algorithm is implemented in the `grf` package. See [Friedberg et al. \(2018\)](#) for an extension of this approach to local linear forests.

Causal Boosting:

An alternative to random forest based causal inference is given by [Powers et al. \(2018\)](#) who introduces boosted trees and causal multivariate adaptive regression splines (MARS). By iteratively fitting weak learners to the residuals of a model, an approximation of the function is build. The idea is to fit a causal tree in the style of [Wager and Athey \(2018\)](#) while setting the basis function $\hat{G}(x, D)$ to zero. Now we estimate the residuals by $Y_i - \epsilon \times \hat{g}_k(X_i, D_i)$ and update $\hat{G}_k = \hat{G}_{k-1} + \epsilon \times \hat{g}_k$. k defines the terminal nodes from the tree and ϵ is the learning rate parameter. After K iterations we return $\hat{G}_K(x, D)$. Estimating the CATE is done by setting D to 1 for the treated observations and 0 for the control-group observations, such that:

$$\hat{\tau}(x) = \hat{G}_K(x, 1) - \hat{G}_K(x, 0). \quad (8)$$

Like in the causal forest the problem remains how to control for overfitting. Especially boosting methods are prone to overfit the data since the trees are not built independently. While a random forest would benefit from using more trees over which to average, in gradient boosting the number of trees is an important tuning parameter that needs to be controlled. In supervised ML we would ideally apply cross-validation. In our case, the parameter of interest is the CATE and we do not observe the true value for each observation. Hence,

cross-validation does not apply here. Instead, we can do something like the honest approach from the causal forest.

[Powers et al. \(2018\)](#) propose to split the data into two distinct sets. The training set is used to build the causal boosting. Using the split-points and split-variables from the training-set we use the covariates from the validation set, lets call it X_v , for validation and get new estimates based on D_v and Y_v for each terminal node. This procedure is done for any of the K trees, using again the residuals (this time from the validation set tree) to reestimate the terminal nodes of the next causal tree. This allows estimating a validation error for each of the original K models. The overall validation error for a causal boosting model is given by the differences of the CATE from the original vs. the validation trees.

Causal BART:

While the (causal) boosting relies on multiplying each sequential tree by the learning rate (ϵ), the idea developed by [Chipman et al. \(2010\)](#) is to estimate a posterior distribution of the prediction by explicitly setting priors for the trees and ensemble structure (e.g. the depth of the tree, the probability of a new split). Using a Bayesian approach allows for a broader set of information than the point estimate from regression and classification methods. The Bayesian Additive Regression Trees (BART) approach is a combination of three methods: Using gradient boosting trees, a Bayesian framing for each individual tree, and Markov chain Monte Carlo (MCMC) sampling to do backfitting (using additive and generalized additive models for posterior sampling). [Hill \(2011\)](#) proposes to use such nonparametric Bayesian models to estimate treatment effects. Given strong ignorability, one way to estimate treatment effect is to estimate the response function $\mu(X_i, D_i)$. This function is estimated in one step instead of estimating two functions. Hence, the prior is set directly for the response surface. This approach is also called the S-learner - train one function and set D_i to 1 and 0 for each observation to get estimates for both potential outcomes. [Hahn et al. \(2020\)](#) extends the idea of using a Bayesian approach to estimate treatment effects but expresses the response surface as:

$$E[Y_i | x_i, D_i = d_i] = \mu\{x_i, \hat{e}(x_i)\} + \tau(x_i) d_i, \quad (9)$$

where $\hat{e}(x)$ is the estimated propensity score and the functions $\mu(\cdot)$ and $\tau(\cdot)$ are independent BART priors. The inclusion of the estimated propensity score can be seen as a covariate dependent prior to control for confounding bias. The method is specially designed to estimate the CATE from observational studies with small effect sizes and heterogeneous effects. The package we use is build on the model by [Hill \(2011\)](#) (`install_github("vdorie/bartCause")`). A package that implements the method proposed by [Hahn et al. \(2020\)](#) is in development (`install_github("socket778/XBCF")`). This package is also available for Python. Note that the causal BART produces credible intervals as a contrast to confidence intervals. They are estimated from the posterior probability function and hence rely on the prior distribution while confidence intervals are based on data only. We will only use the term confidence interval

on all methods, however, we do mean credible intervals for the causal BART and (frequentists) confidence intervals for all other methods.

2.3 Sample splitting and cross-fitting

To aim for a consistent estimator, we need to assume certain complexity conditions on the nuisance functions. Specifically, we want them to be smooth (i.e. differentiable) and the entropy of the candidate nuisance functions to be small enough to fulfill Donsker conditions (e.g. if we assume Lipschitz parametric functions or VC classes). In high-dimensional settings ($p > n$) or when using ML methods that are complex or adaptive, the Donsker conditions might not hold; see, for example, [Robins et al. \(2013\)](#), [Chernozhukov et al. \(2016\)](#) and [Rotnitzky et al. \(2017\)](#). As [Chernozhukov et al. \(2018a\)](#) noticed, verification of the entropy condition is so far only available for certain classes of machine learning methods, such as lasso and post-lasso. For classes that employ cross-validation or for hybrid methods (like the SuperLearner), it is likely difficult to verify such conditions. Luckily, there is an easy solution available: sample splitting. When splitting the sample we can use independent sets for estimating the nuisance functions and constructing the treatment estimation equation. By using different sets, we can treat the nuisance functions as fixed functions which allow avoiding conditions on the complexity. It also allows us to use any ML method such as random forest or boosting or even an ensemble of different methods. The sample splitting approach to avoid smoothness conditions dates back at least to [Bickel \(1982\)](#) and was extended to also use cross-fitting by [Schick \(1986\)](#). For the meta-learners, we have to do sample splitting and cross-fitting manually while the causal forest as well as the causal boosting relies on honest-estimation and does sample splitting by default.

To overcome a potential loss in efficiency, since only a subset of the data is used when estimating the CATE, cross-fitting is an increasingly popular approach to combine ML methods with semi-parametric estimation problems; see, for example, [Chernozhukov et al. \(2018a\)](#), [Newey and Robins \(2018\)](#) and [Athey and Wager \(2017\)](#). Cross-fitting works like this: We split the data into two parts, subset A and M. We use data A to train the nuisance functions and then estimate the parameter of interest using the subset M. Now we switch the roles of the sets, using subset M for training and subset A for estimation. The two results are then averaged. If we want to use this procedure to make predictions, we would build two prediction models based on the roles of the samples and average the resulting values for each observation.

We give an example of the benefit from cross-fitting in Figure 4. We show the MSE from the true treatment effect for a single estimator ($\hat{\tau}_s(x)$ with dots) and the cross-fit estimator ($\hat{\tau}_{cf}(x)$ with triangles) based on a 50:50 sample split. We used the R-learner as the meta-learner and create 50 Monte Carlo replications of the data using the same data generating process (DGP) which simulates a RCT and has the following properties: $N = 2000$, $X = \mathbb{R}^{10}$, $e_0(X) = 0.5$, and $\tau(x) = X_1 + \mathbf{1}(X_2) + W$ with $W \sim \mathcal{N}(0, 0.5)$. Using cross-fitting

decreases the MSE compared to the single estimator in about 90% of the cases. We also find that the variance is smaller compared to the single estimator.

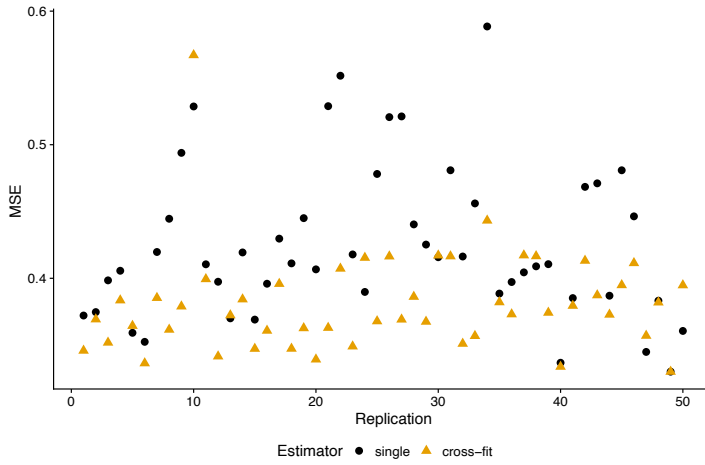


Fig. 4: Single vs. cross-fit estimation of CATE. \mathcal{Q}_{CF}

In this tutorial, we apply 5-fold sample splitting and use 80% of the data for training the models and 20% to evaluate the CATE. We rely on bootstrapping on the training data to create confidence intervals. Cross-fitting is achieved by splitting the training data again into two parts. We repeat this process 20 times and estimate the CATE on the test set for each repetition. We then take the median on the, already cross-fitted, estimates from the test-data for each observation. An alternative is also to take the mean. See [Jacob \(2020\)](#) for a Monte Carlo study about the implications of different sample-splitting, cross-fitting, and averaging approaches for meta-learner methods.

3 Empirical Examples

To illustrate the methods presented in the previous sections we consider two empirical examples. In the first example, we examine the effect that microcredits have on the total amount of loans, resulting from a randomized experiment in Morocco. In the second example, we study the effect of 401(k) eligibility on accumulated assets. This example deviates from random treatment assignment and contains self-selection into a treatment. While all methods condition on the observed pre-treatment variables to estimate the CATE, using the latter example we have to control for the covariates in all steps to eliminate the self-selection bias.

3.1 Effect of microcredits on borrowing


We start with an empirical dataset to analyze the effect of microcredit availability on borrowing activities such as the amount of loans (see Crépon et al. (2015) for a recent study using this dataset). Looking beyond the ATE and finding heterogeneous treatment effects is important to target specific groups and to make better policies. The allocation of treatment was randomized between 162 villages in Morocco. The villages were divided into pairs with similar observable characteristics. Then the treatment was randomly assigned to either one of the pair while the counterpart was assigned to the control group. Treatment as microcredit availability in this context means that between 2006 and 2007 a microfinance institution started operating only in the treated villages. In 2009, 5551 households were surveyed as a follow-up study. We use the results from this survey to estimate conditional average treatment effects using different methods and also show some strategies to get some insight into which characteristics are responsible for heterogeneity in treatment effects. We select the following pre-treatment covariates that are observed characteristics for each household such as head age, the number of adults and number of children, the total number of members in a household, indicators for households doing animal husbandry, other non-agricultural activity, household spouse responded to the survey, the education of the head, having an and outstanding loan over the last year. Table 3 shows the mean value for some covariates. They are categorized by all observations, the treated and the control group. Given these unconditional means, we see that the amount of loans for the treatment group is much higher (2,930) than for the control group (1,802). We also see that the mean of the characteristics is quite balanced across the two groups. This reassures us that the treatment assignment was randomly selected and that there are no confounding variables that lead to self-selection into treatment. While there are small differences in some covariates this is not concerning since all methods that we use here use the propensity score or condition on the covariates to estimate the treatment effect only on similar subgroups. For example, more people in the treatment group already have a loan in the last twelve months. We can estimate the probability of being in the treatment group given this variable and reweigh the treatment and control group to adjust for these differences. The dataset and R-code for the microcredit analysis can be found here  [Qemp](#).

Table 4 shows a summary for the heterogeneous treatment. Namely the effect for the 20% least affected, the ATE, and the effect for the 20% most affected observations. Especially for the quantiles, we find differences between the estimates given the methods that we consider. This holds for the lower 20% where the effect ranges from 450 to 1140 as well as for the 20% most affected with the lowest treatment effect from the DR-learner with 1300 and the highest estimate from the causal BART with 2500. The high value in the upper quantile from the causal BART is because it predicts some extreme values at the tail of the distribution. The T-learner has the highest variance in terms of treatment

Table 3: Descriptive statistics of households (mean)

	All	Treated	Control
<i>Outcome Variable</i>			
Total Amount of Loans	2,359	2,930	1,802
<i>Baseline Covariates</i>			
Number of Household Members	3.879	3.872	3.886
Number of Children	1.2664	1.261	1.272
Head Age	35.976	35.937	36.014
Declared Animal Husbandry Self-employment Activity	0.415	0.426	0.404
Declared Non-agricultural Self-employment Activity	0.146	0.129	0.164
Borrowed from Any Source	0.210	0.224	0.196
Spouse of Head Responded to Self-employment Section	0.067	0.074	0.061
Member Responded to Self-employment Section	0.044	0.048	0.041

Table 4: CATE results for different methods.

Category	Method	20% Least	ATE	20% Most
Meta-Learner	DR-learner	1043.6	1222.0	1299.7
	R-learner	1140.4	1247.7	1342.5
	T-learner	449.4	1391.5	2274.3
	X-learner	733.9	1282.2	1791.7
Modified ML	Causal BART	743.8	1409.3	2502.7
Methods	Causal Forest	1127.6	1323.6	1608.6

effect with a range from 450 amount of loans to 2270 loans on average for the specific quantiles. The ATE is around 1300 for all methods and there is no big difference between the methods. Figure 5 shows the treatment effect for each observation, sorted by the size of the effect. We also show 95% confidence intervals (CI). They are estimated via bootstrapping with $B = 500$ replications. Here we adopt the procedure for the construction of CI's from [Künzel et al. \(2019\)](#). We first split our entire dataset into a training and test-set. We use the training set for bootstrapping by creating a sample from the training data of the same size with replacement. For each meta-learner and each bootstrap sample from the training data, we use the test set to estimate the confidence intervals. In total, we end up having B estimates for each observation. Now we calculate the standard deviation ($\hat{\sigma}$) for each observation. The lower and upper bound is: $[\hat{\tau}_{med}(x) - q_{\alpha/2}\hat{\sigma}; \hat{\tau}_{med}(x) + q_{1-\alpha/2}\hat{\sigma}]$. Especially for the meta-learners, we have a high variance between the bootstrapped samples indicating that even if the CATE is different there might not be a significant heterogeneity. This is also in line with the estimates from the causal BART and the causal forest that show tighter bounds but also an almost flat CATE curve. To calculate the CATE, denoted by $\hat{\tau}_{med}(x)$, we use the whole training data, not a bootstrapped version. We then split the training data again into two parts, one to train the nuisance function and one to predict (for example, the propensity score). We

build the transformed outcomes based on these two samples and then regress them on the covariates to predict on the test set. We repeat this procedure 20 times and take the median. This can be seen as the cross-fitting approach with median averaging as described in [Jacob \(2020\)](#).

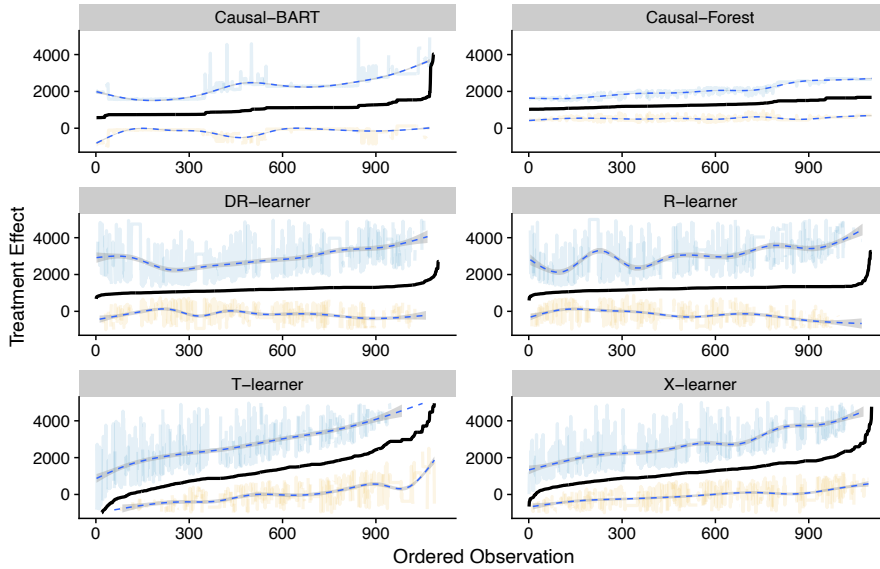


Fig. 5: Microcredit: Observations sorted by level of treatment effect. [O_{sort}](#)

Figure 5 shows quite similar values for at least four of the methods. Only the T- and X-learner have heavier tails and even predict a negative effect for some observations. Given the wide confidence intervals, this might not necessarily be true. The most homogeneous prediction comes from the causal forest.

If we believe that there is at least some difference in the effect between the least and most affected observations then we can look at the average characteristics of these groups to understand what are potential drivers for the heterogeneity. Here we adopt a simple approach introduced by [Chernozhukov et al. \(2018b\)](#), namely the Classification Analysis. The idea is to regress the least and most affected groups on some pre-chosen characteristics with G_q being the observations given a specific group of the treatment effect:

$$\delta_{least} = \mathbb{E}[g(X)|G_{least}] \quad \text{and} \quad \delta_{most} = \mathbb{E}[g(X)|G_{most}].$$

Here we focus on the head age, the probability of being self-employed in a non-agricultural sector, and whether someone had an outstanding loan over the past 12 months (borrowed from any source). In Table 5 we show the average of the characteristics for the two groups as well as if there is a significant difference between the groups. We show results for two methods,

the doubly-robust meta-learner (DR-learner) and the causal forest. The full Table that includes all methods can be seen in the Appendix (9). For both methods, we find a significant difference in head age, the probability of being self-employed in a non-agricultural sector, and the probability of having a loan. The most affected people seem to be younger, while older people are the least affected. Low values in the head age arise since many people did not respond to that question. However, we believe that the non-respondents are missing at random. This allows us to interpret the difference between the two. Looking at employment we find that people who benefit most from microcredits are those who are working in the non-agricultural sector. Agriculture equipment might be more expensive which is why microcredit might not be the first choice. We also find that people who already have a loan (with a higher probability) are less affected by microcredit availability.

Table 5: Classification results for DR-learner and causal forest.

	DR-learner			Causal Forest		
	Most Affected	Least Affected	Difference	Most Affected	Least Affected	Difference
Head age	25.64 (22.63,28.66)	46.25 (43.24,49.26)	-20.61 (-24.87,-16.34) [0.000]	3.82 (2.12,5.53)	57.21 (55.54,58.12)	-53.39 (-55.00,-51.49) [0.000]
Non-agricultural self-employed	0.380 (0.333,0.427)	0.018 (-0.029,0.065)	0.362 (0.295,0.429) [0.000]	0.222 (0.139,0.264)	0.029 (-0.038,0.041)	0.192 (0.121,0.232) [0.000]
Borrowed from any source	0.122 (0.069,0.175)	0.317 (0.263,0.370)	-0.195 (-0.270,-0.119) [0.000]	0.025 (-0.048,0.048)	0.271 (0.241,0.329)	-0.247 (-0.351,-0.219) [0.000]

Notes: 90% confidence interval in parenthesis and p-values in brackets.

3.2 Effect of 401(k) eligibility on accumulated assets

While the microcredit data is based on a randomized controlled trial, the eligibility of a 401(k) pension plan is not. Only some firms offer access to a 401(k) and hence there is self-selection into treatment. It might be the case that more educated people chose firms that provide a 401(k) pension plan and that they have higher financial assets in the first place. [Poterba and Venti \(1994\)](#) argue, that conditioning on observed characteristics, like the income, can restore the random assignment mechanism. The dataset we use is the same as in [Chernozhukov and Hansen \(2004\)](#) which is based on the 1991 Survey of Income and Program Participation. We are interested in the question if 401(k) eligibility, our treatment variable, has an impact on accumulated assets (here we use the net financial assets as the outcome variable). We control for income and other variables related to the job choice that may have an impact on treatment assignment and assets. In total, we have 9915 observations and 13 covariates consisting of age, family size, income, years of education, and indicator variables for married, two-earner status, defined benefit pension status, homeownership,


and IRA participation. We split the dataset into different parts and use 2000 observations to estimate the final treatment effect. The dataset and R-code for the 401(k) analysis can be found here  [emp](#).

Table 6: Descriptive statistics of observations (mean)

	All	Treated	Control
<i>Outcome Variable</i>			
Net financial assets	18,052	30,347	10,788
<i>Baseline Covariates</i>			
Age	41.06	41.48	40.81
Income	37,201	46,862	31,494
Years of education	13.21	13.76	12.88
Proportion of being married	0.60	0.67	0.56
Proportion of two-earners	0.38	0.48	0.31
Proportion of home-ownership	0.63	0.74	0.57

Table 6 shows the mean values for the net financial assets and for some pre-treatment covariates. The amount of assets is higher in the treatment group than in the control group. Concerning the self-selection into treatment we see that some characteristics are different between the treatment and control group. For example, the proportion of home-ownership, years of education, and income is higher for treated people. There are further reasons to believe that such characteristics are positively correlated with financial assets. In this case, we have to control for such variables to account for the self-selection into treatment.

Table 7: CATE results for different methods.

Category	Method	20% Least	ATE	20% Most
Meta-Learner	DR-learner	4998	7120	9806
	R-learner	4250	7410	11320
	T-learner	-4171	7579	25326
	X-learner	-285	7631	18648
Modified ML	Causal BART	2466	9055	21525
Methods	Causal Forest	5210	8228	12360

Table 7 shows the estimated CATE for the 20% least affected and 80% most affected as well as the ATE. The ATE is positive and ranges from 7,120 to 9,055, depending on the method. Its variance between the methods is quite low, compared to the estimates for the least and most affected groups. While the T- and X-learner predict a negative effect from 401(k) eligibility on financial assets for the lowest group, all other methods predict a positive effect. The highest

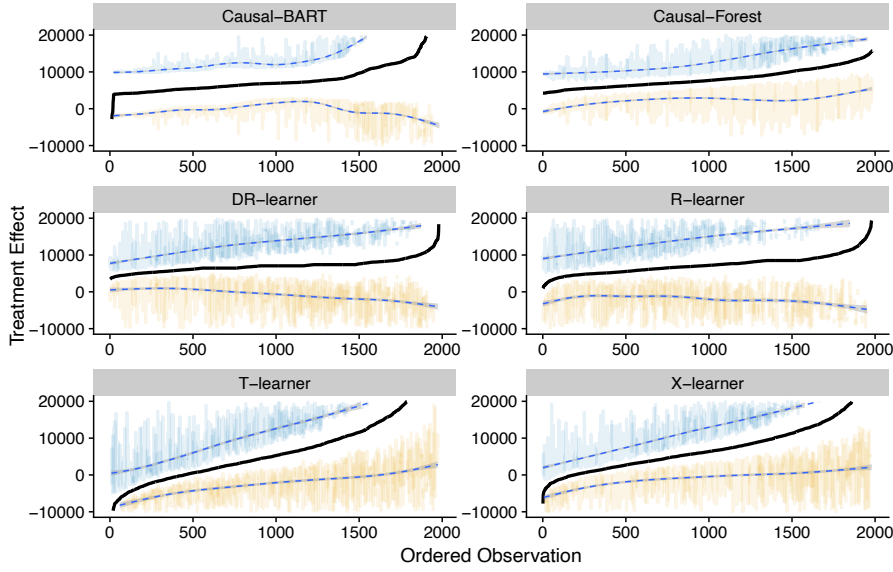


Fig. 6: 401k: Observations sorted by level of treatment effect. \mathbf{Q}_{sort}

affected group has values from 9,806 (from the DR-learner) to 25,326 (from the T-learner). The causal forest predicts values with the lowest heterogeneity. Except for the causal forest, all other learners predict extreme values in the tails of the CATE. If we would use a majority vote from all the methods to interpret the estimated effects, then it is reassuring that everyone has a positive effect from the 401(k) eligibility as can be seen in Figure 6. Given the wide confidence intervals, the evidence of treatment effect heterogeneity is not so clear.

4 Simulated Data

Since the true treatment effect is never known beforehand, we provide a simulation to evaluate different approaches in terms of performance for parameter estimation. The data-generating process allows controlling the number of observations, the dimensionality, and the distributions of the variables. The possibility to specify datasets for different simulations and scenarios helps to investigate the methods used in this tutorial. Note that simulated data often miss realistic data structures. An alternative is to rely on synthetic data where only the treatment effect is artificially added. Since a simulation study in the type of a Monte Carlo study is not the main focus of this tutorial we will only use two simulated data generating processes. The purpose is to give an idea of how to simulate data and test different methods. Instead of relying on purely artificial data, [Wendling et al. \(2018\)](#) creates synthetic data based

on real covariates and treatment assignment mechanism. Only the outcome is simulated based on non-parametric models of the real outcome.

4.1 Data Generating Process

The basic model used in this tutorial is a partially linear regression model based on [Robinson \(1988\)](#) with extensions:

$$Y = \tau_0 D + g_0(X) + U, \quad \mathbb{E}[U|X, D] = 0, \quad (10)$$

$$D = m_0(X) + V, \quad \mathbb{E}[V|X] = 0, \quad (11)$$

$$\tau_0 = t_0(Z) + W, \quad \mathbb{E}[W|Z] = 0, Z \subset X. \quad (12)$$

Let Y be the outcome variable. τ_0 is the true treatment effect or population uplift, while D is the treatment status. The vector $X = (X_1, \dots, X_p)$ consists of p different features, covariates, or confounders, while the vector Z is a subspace of X and represents the variables on which the treatment effect is dependent. U , V and W are unobserved covariates which follow a random normal distribution $= N(0, 1)$.

Equation 11 is the propensity score. In the case of completely random treatment assignment, the propensity score is constant for all units, and if equally distributed then $\hat{m}_0(X_i) = 0.5$. The covariates X are generated from a random multivariate normal distribution $(N(0, 1))$. Note that all values are continuous. In business applications, discrete values (categorical variables) are very common. For the data generation process as well as for the evaluation of most models it would make no difference if such variables are present. This is because vanilla machine learning methods can handle categorical variables quite well. An exception is the causal forest where one has to use one-hot encoding, transform the variable into dummies. Next, we describe the generation of the functions in detail.

Covariates (X)

1. Generate a random positive definite covariance matrix Σ based on a uniform distribution over the space $p \times p$ of the correlation matrix. Let $p = 20$.
2. Scale the covariance matrix. This equals the correlation matrix and can be seen as the covariance matrix of the standardized random variables $\Sigma = \frac{X}{\sigma(X)}$.
3. Generate random normal distributed variables $X_{N \times p}$ with mean = 0 and variance = Σ .

The function $g_0(X)$ is calculated via a linear function with interaction terms and contains the following covariates:


$$g_0(X) = X_1 + X_2 + X_3 \otimes X_4. \quad (13)$$

Next, a description of how to build the function $m_0(X)$ as well as how to create a heterogeneous treatment effect is given. A varying treatment effect implies that its strength differs among the observations and is therefore conditioned on some covariates X . Regarding the treatment assignment (D) two options are considered. Option 1 assumes D to be completely randomly assigned among the observations. In this case, D is just a vector of random numbers with values 0 or 1. In the second option, the treatment assignment is dependent on the covariates. The binary treatment assignment is generated through a Bernoulli function. This implies per default a sort of uncertainty or random error. Even if the probability from the propensity score is at 90% for $D = 1$ there is still a 10% chance that it is generated to be zero. The functions are generated as follows:

<u>Treatment Assignment ($m_0(X, \tau)$)</u>
<p><u>Option 1: m_0</u></p> $D \stackrel{ind.}{\sim} \text{Bernoulli}(m_0), \quad \text{with } m_0 = c \in (0, 1)$
<p><u>Option 2: $m_0(X)$</u></p> <ol style="list-style-type: none"> 1. Dependence is linear: $a(X) = X_1 + X_2 + X_3 - X_4$. 2. Calculate the probability distribution for the vector a from the normal distribution function: $m_0(X) = \Phi\left(\frac{a - \mu(a)}{\sigma(a)}\right) = \frac{1}{2} \left[1 + \text{erf}\left(\frac{a - \mu(a)}{\sigma(a)\sqrt{2}}\right) \right]$ 3. Apply a random number generator from a Binomial function $B\{N, m_0(X)\}$ with probability for success = $m_0(X)$. This creates a vector $D \in \{0; 1\}$ such that $D \stackrel{ind.}{\sim} \text{Bernoulli}\{m_0(X)\}$.


Regarding the treatment effect, two different options are considered. First, τ_0 depends linear on covariates X , and second, τ_0 has a non-linear, more complex form concerning the covariates. In both settings, we can examine heterogeneous treatment effects. The vector $b = \frac{1}{l}$ with $l \in \{1, 2, \dots, p\}$ represents weights for every covariate.

<u>Treatment Effect (τ_0)</u>
<u>Option 1: linear dependence</u>
$\tau(X) = X_1 + X_2 + X_3 + X_4 + W$ with $W \sim \mathcal{N}(0, 0.5)$.
<u>Option 2: non-linear dependence</u>
$\tau(X) = \sin(X_{1:3} \times b_{1:3}) + \cos(X_4)$.

The simulated data that include the true treatment effect can be found here:  [sim](#).

4.2 Results

To evaluate the different methods we consider two data generating processes (DGP). Setting 1 is a randomized controlled trial with a constant propensity score of 0.3 (this is more realistic than 0.5 since treatment is often expensive). The treatment effect depends linear on covariates (setting 1 equals option 1). In setting 2 we consider confounding variables, namely that the treatment probability now depends on covariates (linear) while the treatment effect is calculated via non-linear functions of the covariates. This mimics option 2. In both settings we set $N = 1000$ and $p = 20$. Table 8 shows the mean squared error (MSE) for all considered methods and both settings.

Table 8: CATE results for different methods.  [res](#)

Category	Method	MSE Setting 1	MSE Setting 2
Meta-Learner	DR-learner	0.47	1.01
	R-learner	0.47	0.45
	T-learner	0.57	2.20
	X-learner	0.49	0.65
Modified ML	Causal BART	0.33	0.20
Methods	Causal Forest	0.60	0.39

Notes: The estimated CATE is retrieved by taking the median over 20 repetitions with different sample splits.

Figure 7 and 9 show the sorted treatment effect heterogeneity with 95% confidence intervals for setting 1 and 2, respectively. While the causal BART method produces the lowest MSE it has high credible intervals. The same is true for the causal forest, which performs almost as well as the causal BART but again has a higher variance compared to the DR-learner. This shows that the

calculation of CI's and their interpretation might not always be a measure of the accuracy in these specific cases. Figure 12 in the Appendix shows boxplots of all methods and their variation. The blue line indicates the true ATE, hence we can see how accurate all methods are to predict the ATE. Figure 8 and 10 shows scatterplots for the estimated vs. the true CATE. The blue line indicates a linear regression estimate with pointwise confidence intervals (around the mean) for each method. As we have seen from the MSE, the causal BART method performs best along with the whole interval and in both settings. The DR-learner and T-learner have the highest variance with observations outside the interval of the true treatment effects.

One observation from this simulation is clearly that the meta-learners can improve in terms of MSE the simpler the functions are. We note that the results heavily depend on the choice of the ML methods one uses. If the data structure is quite complicated and non-linear, a lasso model might not be the best choice. While we used a possible selection of the lasso and the random forest we would need to validate how accurate the estimation of the nuisance function is and, in the case of the SuperLearner which model was picked. Here we find that for setting 1 the lasso was chosen in 90% of the cases while in setting 2 a combination of the lasso and the random forest was chosen. Including more ML methods could improve the prediction accuracy depending on the data generating process.

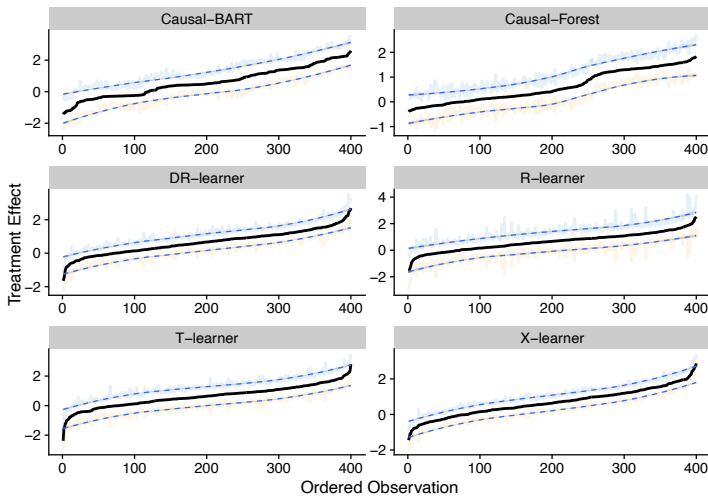


Fig. 7: Setting 1: Observations sorted by treatment effect. [Q51](#)

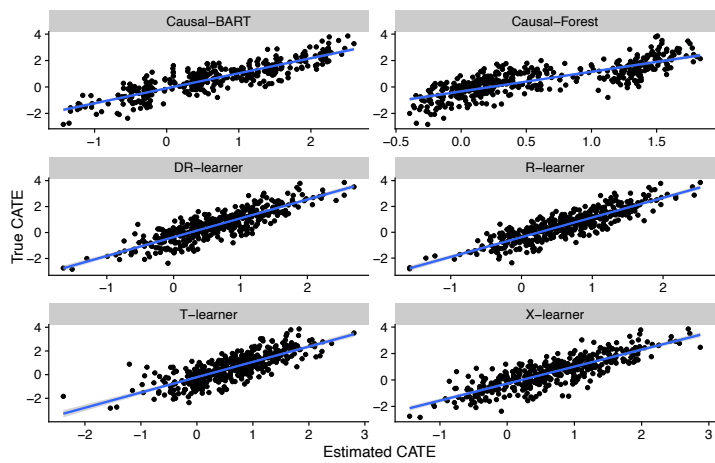


Fig. 8: Setting 1: Scatterplot of estimated and true CATE. \mathcal{Q}_{T1}

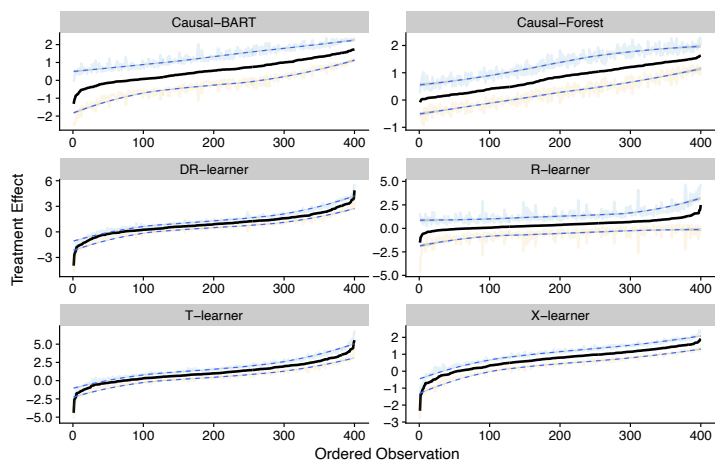


Fig. 9: Setting 2: Observations sorted by treatment effect. \mathcal{Q}_{S2}

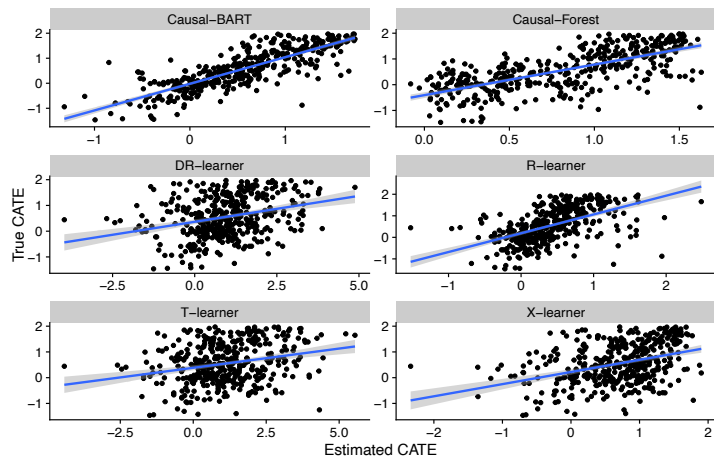


Fig. 10: Setting 2: Scatterplot of estimated and true CATE. \mathcal{O}_{T2}

5 Conclusion

In this tutorial, we present recent novel methods to estimate the conditional average treatment effect using machine learning methods. We categorize the methods into two branches. First, so-called meta-learners, that make use of off-the-shelf machine learning methods by using a transformed outcome to estimate the CATE. They are flexible in the choice of the machine learning method as long as they converge with a specific rate. Research showed that one can use the lasso, random forest, boosting methods, and neural networks. The second branch contains machine learning methods that are specific designed to estimate the ATE or the CATE. They often rely on trees or an ensemble of trees like the random forest. While there are packages available for many methods, the use of meta-learners needs special care since we have to transform the outcome first to estimate the CATE. We, therefore, provide R-code for many of such meta-learners and show how they can be used. We also demonstrate how to use the second branch of methods by integrating the packages in R-code that uses the same data structure as the meta-learners.

To demonstrate the strength and differences of all the methods that we consider we present four examples. Two empirical examples, the first from a randomized control trial and the second from an observational study. The third and fourth example contains simulated data where the true treatment effect can be observed and hence compared with the estimates from all the methods. In the empirical examples, we find strong evidence of positive treatment effects for each observation while significant heterogeneity in the effects is not that clear. This is mainly if we base the conclusion on calculated confidence intervals via the bootstrap or credible intervals from the causal BART. We do, however, find differences in the width of the confidence intervals and also in the CATE prediction among the methods. These differences also occur in the simulated data. We, therefore, recommend that practitioners should not rely on only one method but rather use multiple methods and compare the results. One should also carefully think about the different tuning-parameters that can be set when using machine learning methods. Depending on the method there can be a variety of options to consider. We try to avoid the problem of manually selecting such parameters through cross-validation and the selection of different ML methods for each nuisance function. Sample splitting and averaging is a further necessary step to get robust estimates among the methods.

Conflict of interest

The authors declare that they have no conflict of interest.

References

- Athey S, Wager S (2017) Efficient policy learning. ArXiv preprint, <https://arxiv.org/abs/1702.02896>
- Athey S, Wager S, Tibshirani J (2019) Generalized random forests. *Annals of Statistics* 47(2):1148–1178, DOI 10.1214/18-AOS1709
- Bickel PJ (1982) On adaptive estimation. *Annals of Statistics* 10(3):647–671, DOI 10.1214/aos/1176345863
- Chernozhukov V, Hansen C (2004) The effects of 401 (k) participation on the wealth distribution: an instrumental quantile regression analysis. *Review of Economics and Statistics* 86(3):735–751, DOI 10.1162/0034653041811734
- Chernozhukov V, Escanciano JC, Ichimura H, Newey WK, Robins JM (2016) Locally robust semiparametric estimation. ArXiv preprint, <https://arxiv.org/abs/1608.00033>
- Chernozhukov V, Chetverikov D, Demirer M, Duflo E, Hansen C, Newey W, Robins J (2018a) Double/debiased machine learning for treatment and structural parameters. *The Econometrics Journal* 21(1):C1–C68, DOI 10.1111/ectj.12097
- Chernozhukov V, Demirer M, Duflo E, Fernandez-Val I (2018b) Generic machine learning inference on heterogeneous treatment effects in randomized experiments. ArXiv preprint, <https://arxiv.org/abs/1712.04802>
- Chipman HA, George EI, McCulloch RE, et al. (2010) Bart: Bayesian additive regression trees. *The Annals of Applied Statistics* 4(1):266–298, DOI 10.1214/09-AOAS285
- Crépon B, Devoto F, Duflo E, Parienté W (2015) Estimating the impact of microcredit on those who take it up: Evidence from a randomized experiment in morocco. *American Economic Journal: Applied Economics* 7(1):123–50, DOI 10.1257/app.20130535
- EconML MR (2019) EconML: A Python Package for ML-Based Heterogeneous Treatment Effects Estimation. <https://github.com/microsoft/EconML>, version 0.x
- Fan Q, Hsu YC, Lieli RP, Zhang Y (2019) Estimation of conditional average treatment effects with high-dimensional data. ArXiv preprint, <https://arxiv.org/abs/1908.02399>
- Foster JC, Taylor JM, Ruberg SJ (2011) Subgroup identification from randomized clinical trial data. *Statistics in medicine* 30(24):2867–2880
- Friedberg R, Tibshirani J, Athey S, Wager S (2018) Local linear forests. arxiv. ArXiv preprint, <https://arxiv.org/abs/1807.11408>
- Hahn PR, Murray JS, Carvalho CM (2020) Bayesian regression tree models for causal inference: Regularization, confounding, and heterogeneous effects. *Bayesian Analysis* DOI 10.1214/19-BA1195
- Hansotia B, Rukstales B (2002) Incremental value modeling. *Journal of Interactive Marketing* 16(3):35, DOI 10.1002/dir.10035
- Hill JL (2011) Bayesian nonparametric modeling for causal inference. *Journal of Computational and Graphical Statistics* 20(1):217–240

- Jacob D (2020) Cross-fitting and averaging for machine learning estimation of heterogeneous treatment effects. ArXiv preprint, <https://arxiv.org/abs/2007.02852>
- Kennedy EH (2020) Optimal doubly robust estimation of heterogeneous causal effects. ArXiv preprint, <https://arxiv.org/abs/2004.14497>
- Knaus MC, Lechner M, Strittmatter A (2020) Machine Learning Estimation of Heterogeneous Causal Effects: Empirical Monte Carlo Evidence. *The Econometrics Journal* DOI 10.1093/ectj/utaa014
- Künzel SR, Sekhon JS, Bickel PJ, Yu B (2019) Metalearners for estimating heterogeneous treatment effects using machine learning. *Proceedings of the National Academy of Sciences* 116(10):4156–4165, DOI 10.1073/pnas.1804597116
- van der Laan MJ (2010) Targeted maximum likelihood based causal inference: Part i. *The international journal of biostatistics* 6(2)
- Lunceford JK, Davidian M (2004) Stratification and weighting via the propensity score in estimation of causal treatment effects: A comparative study. *Statistics in Medicine* 23(19):2937–2960, DOI 10.1002/sim.1903
- McCaffrey DF, Ridgeway G, Morral AR (2004) Propensity score estimation with boosted regression for evaluating causal effects in observational studies. *Psychological methods* 9(4):403
- Newey WK, Robins JR (2018) Cross-fitting and fast remainder rates for semiparametric estimation. ArXiv preprint, <https://arxiv.org/abs/1801.09138>
- Nie X, Wager S (2017) Quasi-oracle estimation of heterogeneous treatment effects. ArXiv preprint, <https://arxiv.org/abs/1712.04912>
- Polley EC, Rose S, Van der Laan MJ (2011) Super learning. In: *Targeted learning*, Springer, pp 43–66, DOI 10.1007/978-1-4419-9782-1_3
- Poterba JM, Venti SF (1994) 401 (k) plans and tax-deferred saving. In: *Studies in the Economics of Aging*, University of Chicago Press, pp 105–142
- Powers S, Qian J, Jung K, Schuler A, Shah NH, Hastie T, Tibshirani R (2018) Some methods for heterogeneous treatment effect estimation in high dimensions. *Statistics in Medicine* 37(11):1767–1787, DOI 10.1002/sim.7623
- Robins JM, Zhang P, Ayyagari R, Logan R, Tchetgen ET, Li L, Lumley T, van der Vaart A (2013) New statistical approaches to semiparametric regression with application to air pollution research. *Research report (Health Effects Institute)* (175):3
- Robinson PM (1988) Root-n-consistent semiparametric regression. *Econometrica: Journal of the Econometric Society* pp 931–954, DOI 10.2307/1912705
- Rotnitzky A, Robins J, Babino L (2017) On the multiply robust estimation of the mean of the g-functional. ArXiv preprint, <https://arxiv.org/abs/1705.08582>
- Rubin DB (1980) Randomization analysis of experimental data: The fisher randomization test comment. *Journal of the American Statistical Association* 75(371):591–593, DOI 10.2307/2287653
- Schick A (1986) On asymptotically efficient estimation in semiparametric models. *Annals of Statistics* 14(3):1139–1151, DOI doi:10.1214/aos/1176350055

- Sharma A, Kiciman E, et al. (2019) DoWhy: A Python package for causal inference. <https://github.com/microsoft/dowhy>
- Wager S, Athey S (2018) Estimation and inference of heterogeneous treatment effects using random forests. *Journal of the American Statistical Association* 113(523):1228–1242, DOI 10.1080/01621459.2017.1319839
- Wendling T, Jung K, Callahan A, Schuler A, Shah N, Gallego B (2018) Comparing methods for estimation of heterogeneous treatment effects using observational data from health care databases. *Statistics in medicine* 37(23):3309–3324, DOI 10.1002/sim.7820
- Wyss R, Ellis AR, Brookhart MA, Girman CJ, Jonsson Funk M, LoCasale R, Stürmer T (2014) The role of prediction modeling in propensity score estimation: an evaluation of logistic regression, bcart, and the covariate-balancing propensity score. *American journal of epidemiology* 180(6):645–655, DOI 10.1093/aje/kwu181
- Zimmert M, Lechner M (2019) Nonparametric estimation of causal heterogeneity under high-dimensional confounding. ArXiv preprint, <https://arxiv.org/abs/1908.08779>

A Additional Plots

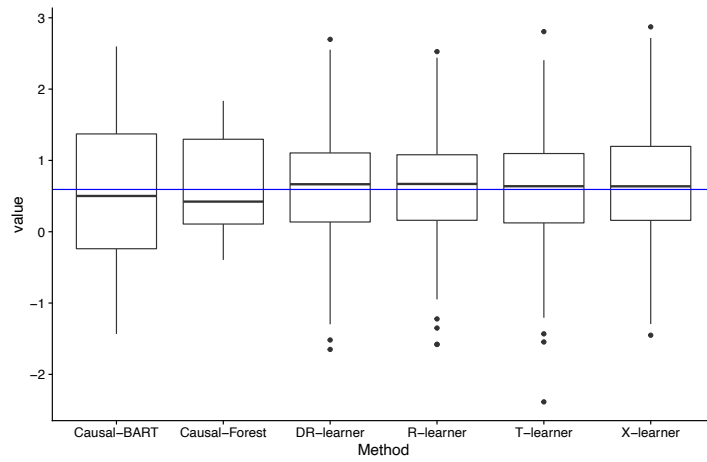


Fig. 11: Setting 1: Boxplots of different Methods. Blue line shows true ATE.

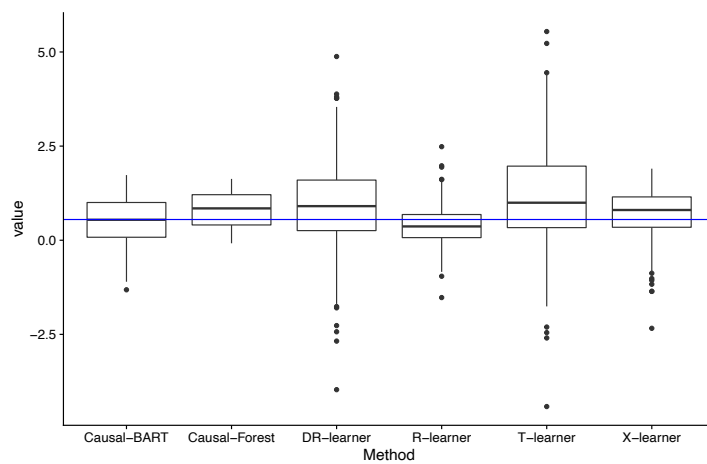
[Q_{B1}](#)

Fig. 12: Setting 2: Boxplots of different Methods. Blue line shows true ATE.

[Q_{B2}](#)

B Tables

B.1 Classification results for the microcredit example

Table 9: CLAN results for different methods.

	DR-learner			R-learner			T-learner		
	Most Affected	Least Affected	Difference	Most Affected	Least Affected	Difference	Most Affected	Least Affected	Difference
Head age	25.64 (22.63,28.66)	46.25 (43.24,49.26)	-20.61 (-24.87,-16.34)	28.46 (25.45,31.47)	46.80 (43.80,49.80)	-18.34 (-22.59,-14.09)	26.43 (23.35,29.51)	44.86 (41.79,47.94)	-18.43 (-22.78,-14.08)
Non-agricultural	-	-	[0.000]	-	-	[0.000]	-	-	[0.000]
self-employed	0.380 (0.333,0.427)	0.018 (-0.029,0.065)	0.362 (0.295,0.429)	0.305 (0.255,0.354)	0.077 (0.027,0.127)	0.228 (0.157,0.298)	0.389 (0.341,0.438)	0.032 (-0.017,0.080)	0.357 (0.289,0.426)
Borrowed from any source	0.122 (0.069,0.175)	0.317 (0.263,0.370)	-0.195 (-0.270,-0.119)	0.245 (0.186,0.305)	0.303 (0.244,0.362)	-0.058 (-0.141,0.026)	0.145 (0.091,0.199)	0.290 (0.236,0.343)	-0.145 (-0.221,-0.069)
	-	-	[0.000]	-	-	[0.350]	-	-	[0.000]
	X-learner			Causal BART			Causal Forest		
	Most Affected	Least Affected	Difference	Most Affected	Least Affected	Difference	Most Affected	Least Affected	Difference
Head age	26.39 (23.35,29.43)	44.81 (41.77,47.85)	-18.42 (-22.72,-14.13)	19.49 (16.82,22.16)	52.76 (50.09,55.42)	-33.27 (-37.04,-29.50)	3.82 (2.12,5.53)	57.21 (55.54,58.12)	-53.39 (-55.00,-51.49)
Non-agricultural	-	-	[0.000]	-	-	[0.000]	-	-	[0.000]
self-employed	0.416 (0.368,0.464)	0.018 (-0.030,0.066)	0.398 (0.331,0.466)	0.235 (0.191,0.280)	0.045 (0.001,0.090)	0.190 (0.127,0.253)	0.222 (0.139,0.264)	0.029 (-0.038,0.041)	0.192 (0.121,0.232)
Borrowed from any source	0.122 (0.068,0.176)	0.335 (0.289,-0.137)	-0.213 (-0.289,-0.137)	0.131 (0.077,0.185)	0.330 (0.276,0.385)	-0.199 (-0.276,-0.122)	0.025 (-0.048,0.048)	0.271 (0.241,0.329)	-0.247 (-0.351,-0.219)
	-	-	[0.000]	-	-	[0.000]	-	-	[0.000]

Notes: Averages are taken from the mean of the CATE over 500 bootstrap iterations.

Algorithm 6: Bootstrap Confidence Interval

- 1 x : features of the training data,
- 2 w : treatment assignments of the training data,
- 3 y : observed outcomes of the training data,
- 4 p : point of interest)
- 5 $S_0 = \{i : w_i = 0\}$
- 6 $S_1 = \{i : w_i = 1\}$
- 7 $n_0 = \#S_0$
- 8 $n_1 = \#S_1$
- 9 **for** b *in* $\{1, \dots, B\}$ **do**
- 10 $s_b^* = c(\text{sample}(S_0, S_1))$
- 11 $x_b^* = x[s_b^*]$
- 12 $w_b^* = w[s_b^*]$
- 13 $y_b^* = y[s_b^*]$
- 14 $\hat{\tau}_b^*(p) = \text{learner}(x_b^*, w_b^*, y_b^*)(p)$
- 15 **end**
- 16 $\hat{\tau}(p) = \text{learner}(x, w, y)(p)$
- 17 $\sigma = \text{sd}(\{\hat{\tau}_b^*(p)\}_{b=1}^B)$
- 18 **return** $(\hat{\tau}(p) - q_{\alpha/2}\sigma, \hat{\tau}(p) + q_{1-\alpha/2}\sigma)$

IRTG 1792 Discussion Paper Series 2021



For a complete list of Discussion Papers published, please visit
<http://irtg1792.hu-berlin.de>.

- 001 "Surrogate Models for Optimization of Dynamical Systems" by Kainat Khowaja, Mykhaylo Shcherbatyy, Wolfgang Karl Härdle, January 2021.
- 002 "FRM Financial Risk Meter for Emerging Markets" by Souhir Ben Amor, Michael Althof, Wolfgang Karl Härdle, February 2021.
- 003 "K-expectiles clustering" by Bingling Wang, Yingxing Li, Wolfgang Karl Härdle, March 2021.
- 004 "Understanding Smart Contracts: Hype or Hope?" by Elizaveta Zinovyev, Raphael C. G. Reule, Wolfgang Karl Härdle, March 2021.
- 005 "CATE Meets ML: Conditional Average Treatment Effect and Machine Learning" by Daniel Jacob, March 2021.

IRTG 1792, Spandauer Strasse 1, D-10178 Berlin

<http://irtg1792.hu-berlin.de>

This research was supported by the Deutsche
Forschungsgemeinschaft through the IRTG 1792.