Random Forests for Estimation of Heterogeneous Treatment Effects from Observational Data: An Empirical Comparison of Generalized Random Forests and Transformation Forests

---

Master Thesis in Biostatistics (STA495)

by

Peter Meili

16-664-841

supervised by

Prof. Torsten Hothorn

University of Zurich, Department of Biostatistics

Zurich, March 2019

Abstract

The understanding of treatment effect heterogeneity became a major topic of research during the last few years. Because classical random forests are not equipped for that task, more specialized random forest methods were introduced lately.
In this thesis the focus lies on comparing causal forests (Wager and Athey, 2018) as a special case of generalized random forests (Athey et al., 2019), and transformation forests (Hothorn and Zeileis, 2017). Based on an extensive setup of simulated data both methods are compared in their predictive accuracy through the mean squared error as a performance measure. These data sets differ in ambivalent dimension as well as number of observations. Furthermore, the influence of different main effect- and treatment propensity functions is investigated for cases with and without orthogonal dependencies of predictor variables. In addition, the influence of number of fitted trees is looked at in a limited manner.
It is found, that for low dimensional data (paired with low numbers of observations) transformation forests tend to have a lower prediction error (up to 40%) than causal forests. With some exceptions causal forest only start to return similar results as transformation forests with high dimensional data sets. It is shown that the accuracy of the fit is highly dependent on the dimension and size of the training data set, while additionally influenced by the underlying distribution of the conditional outcomes. Only a small part could be accounted towards different main effect- and treatment propensity functions. Furthermore, the number of trees used for the fit has a minimal overall influence (while a substantial one for isolated instances).

Acknowledgement

I would like to thank my thesis supervisor Prof. Torsten Hothorn for his guidance during my work on this thesis as well as providing me with an interesting topic.
Additionally the whole staff of the biostatistics master program for their lectures and help in gaining an understanding in the field of biostatistics. In special Dr. Eva Furrer who always did her best to make this master program a great experience. As well as Muriel Buri who helped towards a better readability of the thesis.
Furthermore, I want to acknowledge some of my fellow students Samuel Pawel, Sandra Siegfried, Charlotte Micheloud, Maria Eleni Syleouni, Eleftheria Michalopoulou and Sascha Stutz for the help throughout the semesters, for the support, and for the experiences outside the scope of education.

Contents

Chapter 1

Introduction

Treatment effect heterogeneity understanding is crucial towards a better use and fit of models in a variety of fields like (but not limited to) personalized medicine. The general idea is that treatment effects are not constant for specific subgroups but may depend on patient characteristics (also called predictor variables). The detection of it has gained higher attention over the last years, due to the availability of extensive enough data sets on the matter.
To illustrate the difference between a standard clinical trial setup and a setup where heterogeneity can be detected a simple example is used. The notation will differ from the standard way, to match the notation in the rest of this thesis (which is based on Athey et al. (2019) and Wager and Athey (2018)). There are two treatments A and B (where A is control) and the outcome Y given the treatments. The assumed models are

$$\begin{aligned} Y \mid \text{treatment A} &\sim N(m, \sigma^2), \\ Y \mid \text{treatment B} &\sim N(m + \tau, \sigma^2), \end{aligned}$$

where m is the overall intercept (main effect) and τ the treatment effect. Whereas the overall intercept describes the prognosis for subjects under treatment A, the treatment effect describes the causal effect induced by switching to treatment B. In a standard clinical trial both groups would then be randomized for $\hat{\tau}$ estimation. In this setup the predictor variables x and prognostic variables z are not taken into account, hence heterogeneous treatment effects cannot be detected. If it is now assumed, that the main- and treatment effect are depending on predictor variables x and prognostic variables z the models change to

$$\begin{aligned} Y \mid \text{treatment A}, z &\sim N(m(z), \sigma^2) \\ Y \mid \text{treatment B}, x, z &\sim N(m(z) + \tau(x), \sigma^2). \end{aligned}$$

In the above model z is the prognostic variable used for the prognostic effect $m(z)$, while $\tau(x)$ represents the heterogeneous treatment effect. For this thesis x and z are the same, hence in further sections z will be replaced by x. The aim is to estimate $\hat{\tau}(x)$, potentially from observational data where propensities $\mathbb{P}[B \mid W(x)] = e(x)$ determine the probability of receiving treatment $B|X$. A special case of this would be clinical trials with balanced parallel groups, where $e(x) = 0.5$.

The classical random forest (Breiman, 2001) only estimates $\hat{m}(x)$ in the absence of any treatment. Because of that restriction, methods that are more elaborate are needed. An earlier introduced method to estimate $\tau(x)$ is virtual twins (Foster et al., 2011). The basic idea behind this approach is that for every patient with (for example) treatment $Y \mid A, X$, an outcome $Y \mid B, X$ is generated (and vice versa), where the difference of both gives $\hat{\tau}(x)$. To arrive at such counterfactual estimates one first fits a random forest through regressing the observed $Y_i^{(1)}$ against (X_i, W_i). In this context the superscript (1) refers to a treated subject as an example (for control $Y_i^{(0)}$ the same steps would apply). Afterwards, the original treatment group of the subject will be switched to its counterpart with $(1 - W_i)$. Following the above example with a treated subject, the altered $(X_i, 1 - W_i)$ are then run down the trained forest. This gives estimates for $\hat{Y}_i^{(0)}$ which is the counterfactual estimate of $Y_i^{(1)}$ allowing estimation of the treatment effect with $\hat{\tau}(x) = Y_i^{(1)} - \hat{Y}_i^{(0)}$.
The focus of this thesis lies on two recently suggested methods for estimation of heterogeneous treatment effects. Causal forests (Wager and Athey, 2018) which are a special case of generalized random forests (Athey et al., 2019), and transformation forests (Hothorn and Zeileis, 2017). The former explicitly targets observational data by incorporating propensities $e(x)$ while the latter was so far evaluated for randomized clinical trial data only. Transformation forests are an extension of model-based random forests (Seibold et al., 2018).
The main conceptual difference between causal forest and model-based forest lies in the workflows of both methods. For causal forest the workflow is:

1. Estimate propensities $e(x)$ with a random forest for binary treatment decisions.
2. Estimate the prognostic effect $m(x)$ with regression forest for Y (on the predictor variables).
3. Estimation of $\hat{\tau}(x)$ by running a generalized random forest on centered responses $Y - \hat{Y}(x)$ and centered treatment assignments $W - \hat{W}(x)$.

The main difference to transformation model-based forests is, that step (1.) is missing (but will be added herein) and that steps (2.) and (3.) are performed simultaneously by a model-based forest.
This thesis can be aligned with work by Lu et al. (2018), where the causal forest method has already been compared to a set of different random forest based procedures (like virtual twins). It was deemed to be in the middle- to lower field performance wise compared to the other six methods, depending on the simulation setup used. Model-based forests (and hence transformation forests) were not one of these other methods under test.

1.1 Project Aim

The aim of this master thesis is to adapt transformation forests to observational data by incorporating propensities (in the same way as causal forests), to predict heterogeneous treatment effects with high accuracy and consistency. Both of these methods rely on the already well known random forest approach introduced by Breiman (2001). The goal is to detect specific performance-wise differences between the two methods. Based on these

differences, suggestions should be possible concerning the specific benefits that each of the models supply in reference to the different setups on which the artificially generated data sets depend up on.
Because the whole simulation setup in this thesis is in majority based on the setup of Wager and Athey (2018) a limited reproduction, of the for this thesis important parts of the paper, is carried out. Based on a successful implementation the rest of the thesis then can be build up on this previous work, with knowledge of the correctness of it. It is additionally advantageous for a better understanding of the causal forest implementation and method through learning by doing.
These two methods are trained on an extensive setup of test data sets with a high number of replicas per data set. While the testing is done on a separate test set to get the out-of-sample performance. Test- and training data are generated artificially through different setups of the parameters of the underlying data generating distributions, as well as with the use of different distributions themselves. Additionally different combinations of treatment heterogeneity with and without confounding are used.
As already mentioned above, different distributions for the conditional outcomes are implemented. This is to test cases where outcomes are not normal, but are based on (as used in this thesis) a Weibull distribution. This change will influence the centering of the response in a way that $Y - \hat{Y}(x)$ is not possible to do anymore without using an additional procedure.
The main performance measure is the mean squared error between estimation and true treatment effect. The influence of treatment propensity functions $e(x)$ as well as main effect functions $m(x)$ is investigated as well as the effect the number of trees in the forest has on the accuracy of the fit.

Chapter 2

Theory

A theoretical overview about the random forest approach in general as well as the two specific methods used in this thesis will be given in this section. In a first step, the classical random forest (Breiman, 2001) is explained in Section 2.1 as a short introduction to the topic. The general idea of treatment effect estimation is shown in Section 2.2 which both methods depend up on. Following, causal forests (Wager and Athey, 2018) are introduced in Section 2.3 with a special focus on local centering. Transformation forests (Hothorn and Zeileis, 2017) follow in Section 2.4 with a detailed account on the score functions used. Because both methods rely on almost identical tree aggregation schemes the topic was bundled in Section 2.5, followed at last with a summary on the main differences of both methods in Section 2.6.

2.1 Random Forest

Because it is already a well known ensemble learning method, a broad overlook about random forest by Breiman (2001) will be given in the following without going into detail (for a more detailed explanation see for example Hastie et al. (2001)).
Random forest is a method, which is based on the averaging of many single decision trees. One has to differentiate between predictions which require classification, where $y \in \{0, 1\}$ or regression with $y \in \mathbb{R}$. Each tree is based on the bagging (bootstrap aggregation) technique.
Bagging is a method, which reduces variance for high-variance low-bias procedures like trees. It uses M randomly (with replacement) drawn subsamples out of the learning sample. A prediction (the bagging estimate) is then made for each of the bootstrap samples and averaged with the mean over all samples.
The variance of the average of n random variables (with positive correlation ρ) is (Hastie et al., 2001)

$$\rho\sigma^2 + \frac{1-\rho}{n}\sigma^2 \tag{2.1}$$

where n is a number of i.i.d. random variables. As the number of random variables increases the second term will asymptotically go to 0, which leaves $\rho\sigma^2$ as the remaining portion. Hence there is a limit to the reduction in variance (as number of variables

increases) due to correlation ρ. This is exactly where random forest takes over with the goal of reducing the correlation between pairs of variables.
"The idea of random forest [...] is to improve the variance reduction of bagging by reducing the correlation between the trees, without increasing the variance too much."(Hastie et al. (2001), 588). This is done through the (random choosing) process described in the second step of the following procedure. Additionally overfitting can also be avoided through the random choosing of the subsample to place the splits in step two.

1. Draw (with replacement) M bootstrap samples each of size n from the original training data set.

2. A fully grown regression tree (without pruning) is generated with every bootstrap sample. The splits are placed based on a random subsample of each bootstrap sample instead of the whole one. If the sample size is denoted with p and the size of the random subsample with p_r, it is usually $p_r << p$.

3. Grow as many trees as specified.

To obtain predictions on a new observation x in a regression setting, all individual tree predictions are averaged on the new x. This averaging is done with $1/B$ (where B is the number of trees) times the sum of the predictions in every single tree.
For classification, a majority vote among the nodes over all trees is carried out (similar to bagging) and the classifier with the most votes (among all subsamples) is chosen. Because the tree is not pruned, the bias can be kept minimal.

2.2 Treatment Effect Estimation

A treatment effect is the difference between two potential outcomes in which a patient is treated and not treated. If the trial design does not allow for such a setting, a direct estimation is not possible, hence a different approach is needed. The variables used for response, explanatory variables and treatment indicator are (Y_i, X_i, W_i) where $i = 1, \ldots, n$ (with n independent observations). Whereas $Y_i \in \mathbb{R}$ and $X_i \in \mathbb{R}$, treatment indicator $W_i \in \{0, 1\}$.
Formally, given the response vector Y_i and explanatory variables X_i, the treatment effect is defined as

$$\tau(x) = \mathbb{E}[Y_i^{(1)} - Y_i^{(0)} \mid X_i = x]. \tag{2.2}$$

The potential outcomes $Y_i^{(0)}$ and $Y_i^{(1)}$ of the response vector represent the two cases an individual would have experienced with and without receiving treatment. This leads to the above mentioned challenge, that only one outcome can be observed (either patient received treatment or control). Therefore $\tau(x)$ cannot be estimated directly from the observed data (X_i, Y_i, W_i).
The standard way to tackle this issue is, to assume unconfoundedness of the treatment assignment W_i with the potential conditional response $Y_i|X_i$. Because the nearby observations in X_i can now be treated as from a randomized experiment (due to independence), this assumption leads to consistency for $\tau(x)$.

Based on this unconfoundedness restriction, the treatment effect can be rewritten as (Wager and Athey, 2018)

$$\tau(x) = \mathbb{E}\left[Y_i\left(\frac{W_i}{e(x)} - \frac{1-W_i}{1-e(x)}\right) \mid X_i = x\right], \text{where } e(x) = \mathbb{E}[W_i \mid X_i = x] \tag{2.3}$$

where $e(x)$ is the treatment propensity. To show that this is equivalent to Equation (2.2), both cases of the treatment indicator can be tried. For the case where a treated subject is given ($W_i = 1$), the second term will be 0, hence one will end up with $Y_i/e(x)$ (which is the $Y_i^{(1)}$ from before). If an untreated one is given ($W_i = 0$), the first term will be 0 and the second one will simplify to $Y_i/(1-e(x))$ which is the $Y_i^{(0)}$. Now, one only has to estimate the treatment propensity $e(x)$ and with the use of Equation (2.3) the treatment effect could be estimated.

2.3 Causal Forest

This method seeks to address "[...] the fear that researcher will iteratively search for subgroups with high treatment levels, and then report only the results for subgroups with extreme effects [...]"(Wager and Athey (2018), 2). Additionally a basic problem with classical approaches like k-nearest neighbors is, that they do not perform well when the number of predictor variables is increased. Because the causal forest uses a data-driven way for calculating the weights (similar to random forest), which nearby observations receive, this problem can be solved.
Additionally Wager and Athey (2018) try to address the issue that the asymptotic as well as a suitable inference framework is missing for most random forest methods. Through introduction of constraints, rigorous asymptotic analysis of the forest is introduced with this method, namely asymptotic normality theory that allows for statistical inference to be done on the forest. The constraint, which needs to be fulfilled, is called "honesty". This constraint is based on the idea that the learning sample should be divided into two subsamples, of which one is used for growing the tree and the other for prediction in the leaves. Two procedures will be introduced later, which fulfill the honesty requirement (so called propensity trees and double sample trees).
These causal forests have three properties which allow for inference (for more details see Wager and Athey (2018) section 2.3, or Athey et al. (2019) section 3).

- Causal forests are consistent for $\tau(x)$.
- Predictions are asymptotically Gaussian and unbiased which can be formalized with $(\hat{\tau}(x) - \tau(x))/\sqrt{Var[\hat{\tau}(x)]} \Rightarrow N(0,1)$. This means that the difference of prediction and true treatment effect divided through the standard deviation of the predicted treatment effect is asymptotically standard normal distributed.
- Asymptotic variance can be accurately estimated for causal forest. This is done with the use of the infinitesimal jackknife method for random forest (Efron (2014), Wager et al. (2014))

Because there is no such thing as a free lunch, there are some restrictions on these causal forests. As mentioned before these trees are required to be honest to achieve consistency and centered asymptotic normality. The honesty criteria is defined by "A tree is honest if, for each training sample i, it only uses the response Y_i to estimate the within–leaf treatment effect $\tau(x)$ [...] or to decide where to place the splits, but not both" (Wager and Athey (2018), 8).
One way to implement this is the double–sample tree (procedure 1 in Wager and Athey (2018), or algorithm 1 in Athey et al. (2019)) which divides the training data samples into two halves. One half is used to place the splits, while the other half is used for the within-leaf estimation. The double–sample tree then estimates $\hat{\tau}(x)$ on the second half of the sample. This procedure makes the forest more sensitive to changes in the treatment effect (when treatment heterogeneity is present), when there is no confounding for the main effect and treatment propensity.
A second approach to satisfy the honesty criteria is the use of propensity trees (procedure 2 in Wager and Athey (2018)), where Y_i is not taken into account when placing splits. Instead of the response, the treatment indicator W_i is used to train a classification tree first. This means that the classification tree is trained on a subsample of the (X_i, W_i) pairs. This procedure goes through without sample splitting and should be beneficial for reducing bias caused by variation in $e(x)$.
On a model level, the following score function estimating equation is used

$$f(m, \tau) = y - (m + \tau w), \tag{2.4}$$

where m (which is plug-in $\hat{m}(x)$ in this thesis) as the conditional mean function, and τ as the treatment effect are the unknown parameters. The model assumes normality of the conditional outcomes. The normal log-likelihood is $\ell(\tau) = 2^{-1}(y - \tau w)^2$, which leads to the scores

$$s_\tau = \begin{pmatrix} (y - \tau_1 w)w \\ (y - \tau_2 w)w \\ \vdots \\ (y - \tau_N w)w \end{pmatrix}$$

while the L2 loss function for estimation of the parameters is $L(m, \tau) = (y - (m + \tau w))^2$. The splits are induced through minimizing an error term (see Athey et al. (2019), section 2.2 for further explanation). The following procedure is used for estimation of $\hat{\tau}$.

1: **procedure** CAUSAL FOREST(Y, X, W)
2: $W \sim X$ ▷ Estimate $\hat{e}(x)$ by classification forest
3: $Y \sim X$ ▷ Estimate $\hat{m}(x)$ by regression forest
4: $Y - \hat{m}(x) \sim [W - \hat{e}(x)]|X$ ▷ Estimate $\hat{\tau}(x)$ through local centering
5: **end procedure**

2.3.1 Local Centering

Although viable inference can be done on the parameters with the above discussed approach "performance of the forests can in practice be improved by first regressing out

the effect of the features X_i on all outcomes separately." (Athey et al. (2019), 21). This is achieved with centering of the treatment indicators W and the conditional outcomes $Y|X, W$. The centering idea was introduced by Robinson (1988) for normal linear models, where the focus was on prove of consistency and efficiency of such centered parameters. The implementation is straightforward with

$$W_{\text{centered}} = W - \hat{W}(x),$$
$$Y_{\text{centered}} = Y - \hat{Y}(x).$$

$\hat{W}(x)$ and $\hat{Y}(x)$ refer to the estimated treatment indicators and outcomes (see Section 2.6 for further clarifications). These are first estimated using two separate regression forests. The forests for prediction of $\hat{\tau}$ are then run on the centered outcomes instead of the original ones. Towards why one can center in the first place an explanation can be found in Athey et al. (2019) section 6.1.1.
The aggregation procedure of the single trees is explained in Section 2.5 because it is the same as for transformation forests. For further analysis and theory on causal forests and generalized random forests consult Wager and Athey (2018) and Athey et al. (2019). Whereas Athey et al. (2019) focus on the theoretical analysis of generalized random forests, Wager and Athey (2018) induce a more simulation based take.

2.4 Transformation Forests

Most regression and random forest methods give information about the conditional expectation $\mathbb{E}(Y|X = x)$ while the understanding of the full predictive distribution $Y|X$ is lacking. The reason is that the splits in the majority of forest methods are sensitive to mean changes, while changes in higher moments (like variance) are not picked up. Estimation of conditional distribution functions has already been done over ten years ago by Hothorn et al. (2004) and somewhat similar by Meinshausen (2006) with quantile regression forest.
To tackle this problem, transformation trees and its extension transformation forests were introduced by Hothorn and Zeileis (2017). This method can be considered as an "[...] adaptive local likelihood estimator of conditional distribution functions." (Hothorn and Zeileis (2017),1) that are sensitive to distributional changes. A second advantage of the method is its allowance for a broad number of classical inference procedures (e.g. variable importance or independence tests) because models are fully parametric in comparison to the causal forest ones.
Hothorn and Zeileis (2017) start with the introduction of a parametric family of distributions

$$\mathbb{P}_{Y,\Theta} = \{\mathbb{P}_{Y,\vartheta} \mid \vartheta \in \Theta\} \tag{2.5}$$

where Y is the target random variable, ϑ the parameters and Θ the parameter space. If predictors X are taken into account and the assumption holds that $\vartheta(x) \in \Theta$, the conditional distribution $\mathbb{P}_{Y|X=x}$ can be rewritten as $\mathbb{P}_{Y|\vartheta(x)}$. This means, that the conditional distribution $\mathbb{P}_{Y|X=x}$ is a member of the parametric family of distributions as seen

in Equation (2.5). If different parametrizations are put in place for the parametric family $\mathbb{P}_{Y,\Theta}$, implementation can be difficult, which is addressed with the use of transformation models.
A basic transformation model of the form $\mathbb{P}(Y \leq y) = F_Y(y) = F_Z(h(y))$ is introduced, where $h()$ is a monotone increasing transformation function. This setup allows for simple to complex transformation functions, which will enable a wide range of statistical applications. These models are used, so that the score contributions can be calculated for inducing the splits.
In contrast to generalized random forests, there is now no limitation to normality anymore, because $F_Z()$ can be any distribution. In the simulations, that will be discussed in Chapter 3, normal- and Weibull based outcomes are used, hence these two cases will be looked at here. In a first step, the following densities are defined

$$f(m, \sigma, \tau) = \phi(y/\sigma - m/\sigma - \tau w)/\sigma \tag{2.6}$$
$$f(\theta_1, \theta_2, \tau) = \exp(\theta_1 + \theta_2 \log(y) - \tau w - \exp(\theta_1 + \theta_2 \log(y) - \tau w))\theta_2 \tag{2.7}$$

where Equation (2.6) is the normal model while (2.7) the Weibull, where the true parameters $m(x), \sigma(x)$ and $\tau(x)$ may depend on x in a single forest. The loss function in the normal case is $L(m, \sigma, \tau) = (y/\sigma - (m/\sigma + \tau/\sigma W))^2$. As in the standard way the likelihoods need to be calculated to arrive at the scores. The log-likelihoods are

$$\ell(m, \sigma, \tau) = -2^{-1}(y/\sigma - m/\sigma - \tau w)^2 - \log(\sigma)$$
$$\ell(\theta_1, \theta_2, \tau) = \theta_1 + \theta_2 \log(y) - \tau w - \exp(\theta_1 + \theta_2 \log(y) - \tau w) + \log(\theta_2)$$

which (derived towards every unknown parameter) give the three dimensional scores. Because a full representation of the score matrix overflows the page, a simpler representation is chosen. Instead of the 3xN matrix including all the scores for $i = 1, \ldots, N$, a 1x3 is displayed.

$$s_{\text{Normal}} \begin{pmatrix} m \\ \sigma \\ \tau \end{pmatrix} = \begin{pmatrix} -\frac{1}{\sigma^2}(m + \tau\sigma w - y) \\ -\frac{1}{\sigma^3}(\sigma^2 + \tau\sigma w(y - m) - (y - m)^2) \\ -\frac{1}{\sigma} w(m + \tau\sigma w - y) \end{pmatrix}$$

$$s_{\text{Weibull}} \begin{pmatrix} \theta_1 \\ \theta_2 \\ \tau \end{pmatrix} = \begin{pmatrix} 1 - y^{\theta_2} \exp(\theta_1 - \tau w) \\ -y^{\theta_2} \exp(\theta_1 - \tau w) \log(y) + \frac{1}{\theta_2} + log(y) \\ y^{\theta_2} \exp(\theta_1 - \tau w) w - w \end{pmatrix}$$

To arrive at the 3xN matrix, the 1x3 matrix has to be transposed and the parameters made dependent on i. The splitting of the tree nodes is done with statistical tests based on these scores. To induce the splits the following H_0 is tested (Hothorn and Zeileis, 2017)

$$H_0 : s(\hat{\vartheta}^N_{ML}|Y) \perp X \tag{2.8}$$

where s are the score contributions of the transformation family and $\hat{\vartheta}_{ML}^N$ the unconditional maximum likelihood estimator not depending on x. Following the notation from before the score test can also be written as $H_0 : s(m, \sigma, \tau) \perp X$, where $(m, \sigma, \tau) \in \mathbb{R}^3$. This hypothesis tests if all models come from the same distribution and hence make the transformation forest sensitive to changes in those.
The following procedure is used for estimation of $\hat{\tau}$.

1: **procedure** "CAUSAL" TRANSFORMATION FOREST(Y, X, W)
2: $\quad W \sim X$ ▷ Estimate $\hat{e}(x)$ by classification forest
3: $\quad Y \sim [W - \hat{e}(x)]|X$ ▷ Estimate $\hat{\tau}(x)$ and $\hat{m}(x)$ simultaneously
4: **end procedure**

For further analysis and theory on transformation forests see Hothorn and Zeileis (2017) and Hothorn et al. (2019).

2.5 Aggregation of Trees

Both trees are aggregated in the same way over nearest neighbor weights which measure how often x_i falls into the same leave (or terminal node) as x. In the following it will be discussed on the transformation forest case from Hothorn and Zeileis (2017)
To measure the similarity of the two distributions $\mathbb{P}_{Y|X=x}$ and $\mathbb{P}_{Y|X=x_i}$, a so called "conditional weight function" $w_i^N(x)$ is introduced. This conditional weight function represents how "close" x (one observation x) and x_i (all other observations except x) are through counting how many times the i-th observation ends up in the same terminal node. With the use of this measure, as well as the log-likelihood contribution of the probability model from Equation (2.5), the following forest conditional parameter function can be defined (Hothorn and Zeileis, 2017):

$$w_{\text{Forest},i}^N(x) := \sum_{t=1}^{T} \sum_{b=1}^{B_t} I(x \in B_{tb} \wedge x_i \in B_{tb}) \tag{2.9}$$

$$\hat{\vartheta}_{\text{Forest}}^N(x) := \arg\max_{\vartheta \in \Theta} \sum_{i=1}^{N} w_{\text{Forest},i}^N(x) \ell_i(\vartheta) \tag{2.10}$$

The conditional weight function of the forest can be seen in Equation (2.9). N is the total number of observations in the learning sample and T is the number of trees in the forest. B_{tb} is the $b-$th terminal node in the $t-$th tree which, in every node, contains the parameter estimate $\hat{\vartheta}_{tb}^N$, and in every tree the conditional parameter function $\hat{\vartheta}_{\text{Tree}}^N(x)$. In less formal terms, the sum over all trees and all cells of each tree, where x and x_i are in the same terminal node (which are only then considered to be "close") gives the conditional weight function.
In Equation (2.10) the forest conditional parameter function can be seen which includes the conditional weight function from Equation (2.9). The conditional weights are multiplied with the unconditional log likelihood function contributions of the learning sample observations and summed up over the whole learning sample. Furthermore, the likelihood

is maximized to get the parameter estimate for $\hat{\vartheta}$. The ϑ parameter is in the normal case defined with $\vartheta = (m, \sigma, \tau)$, while with $\vartheta = (\theta_1, \theta_2, \tau)$ for the Weibull case.
The equivalent to Equation (2.10) for the generalized random forest case is (Athey et al., 2019)

$$(\hat{\theta}(x), \hat{v}(x)) \in \arg\min_{\theta,v} \{\| \sum_{i=1}^{n} \alpha_i(x) \psi_{\theta,v}(Y_i, W_i) \|\}. \tag{2.11}$$

The $\alpha(x_i)$ is the $w^N_{\text{Forest},i}(x)$ from the transformation forest case (for the formal definition of $\alpha(x_i)$ see Equation 3 in (Athey et al., 2019)). The v is an optional nuisance parameter while θ is the ϑ from above. Furthermore, instead of maximizing a likelihood the Euclidean distance is minimized.

2.6 Difference Between Causal Forest and Transformation Forest

This section aims to give a more straightforward explanation on what the differences of the aforementioned two methods actually are in regard to their calculation steps as well as the models used.
On the model level one has to compare the model Equation (2.4) for causal forest with the model equation for transformation forest. The main difference is, that causal forests use a least square score with respect to τ after centering, while transformation forests use a likelihood score with respect to m, τ, σ without centering of the response. Differences as seen in Table 2.1 arise for the underlying distribution, unknown parameters and capability of taking censored observations into account. As visible, for causal forest a normal distribution is assumed as underlying whereas in transformation forest any continuous distribution $F_Z()$ can be chosen. In the transformation forest context, two unknown parameters are estimated whereas for causal forest it is only one. Furthermore, censored observations can be taken into account in the transformation setup while not for causal forests.

Table 2.1: Comparison of model equations for causal forest (left column) with transformation forest (right column).

	$y - (m + \tau w)$	$m/\sigma + y/\sigma + \tau/\sigma w$
Distribution	$N \sim (\mu, \sigma)$	Any $F_Z(h(y))$
Scores	$\mathbb{R}$	$\mathbb{R}^3$
Estimated parameters	$\hat{\tau}$	$\hat{m}, \hat{\tau}$
Censoring possible	No	Yes

An overview about the calculation steps for causal- and transformation forest can be seen in Table 2.2, where the numbers in parentheses indicate the step. These are the steps introduced in both procedures in Section 2.3 and Section 2.4. For causal forest, one first estimates the treatment propensity and the main effect separately and uses the centered version of both for estimation of the treatment effect. For transformation forests only, the treatment propensity is needed and the estimation of main– and treatment effect happens simultaneously.

Table 2.2: Calculation steps for the treatment propensity, main effect and treatment effect for causal– and transformation forest. It is assumed, that response Y, predictor variables X and treatment indicator W are known.

	Causal forest	Transformation forest
$\hat{e}(x)$	(1) $W \sim X$	(1) $W \sim X$
$\hat{m}(x)$	(2) $Y \sim X$	
$\hat{\tau}(x)$	(3) $Y - \hat{m}(x) \sim [W - \hat{e}(x)]\|X$	
$\hat{\tau}(x), \hat{m}(x)$		(2) $Y \sim [W - \hat{e}(x)]\|X$

In a first step, the treatment propensity is estimated through regressing the treatment indicator on the predictor variables. This procedure is carried out for both methods as denoted with (1). In the causal forest framework, the main effect is estimated at (2) with regressing Y on the predictor variables X. The $\hat{e}(x)$ and $\hat{m}(x)$ are then used to calculate the centered outcomes $(Y_{\text{centered}}, W_{\text{centered}})$ on which the estimation of the treatment effect is done. For transformation forest, only the treatment assignments are centered, while the main effect $\hat{m}(x)$ will be estimated simultaneously with the treatment effect $\hat{\tau}(x)$ in (2), while both being reparametrized later.
The centering of both variables is used to improve the performance of the forest. Take for example the treatment propensity $e_i(x) = 0.9$. If $W_i = 1$ than $W_{\text{centered}}(x_i) = 0.1$, hence close to 0, while if $W_i = 0$ one would have $W_{\text{centered}}(x_i) = -0.9$. Therefore, if this is done for $i = 1, \ldots, N$ the W_{centered} becomes centered around zero with values between $[-1, 1]$. The responses Y behave similar, although the range of values for Y_{centered} is not bound to $[-1, 1]$.

Chapter 3

Simulations

The overall goal is to compare causal forests and transformation forests in their ability to predict the treatment effect with a minimal mean squared error. In a first step, the data generating process based on Athey et al. (2019) is shown in Section 3.1, followed by a detailed explanation on the calculation setup in Section 3.2. The results can be found in Section 3.3. Additionally in the same section, an analysis in to the influence of different treatment propensity- and main effect functions is shown, as well as a short discussion about the influence of different tree numbers.

3.1 Data Generating Process

The data generating process explained in this section is in most parts based on the similar setup of section 6.2 in Athey et al. (2019) as well as section 5.2 in Wager and Athey (2018). The full code can be found in Section A.5.1. The following three changes and extensions are made to the process to obtain a more general view:

- In addition to the use of the normal distribution as basis for the conditional outcomes $Y_i|X_i, W_i$ the Weibull distribution is used. This leads to a doubling of the total simulation setups.
- A periodic behavior of the treatment propensity is introduced using a sinus function on orthogonal predictor variables.
- A setup with a constant treatment effect is not taken into account as opposed to Wager and Athey (2018). Instead only setups with treatment heterogeneity paired with constant (and variable) main effects and constant (and variable) treatment propensities as well as setups with (and without) interaction between the latter two are considered.

First, the following three parameters with their definitions are introduced (Wager and Athey, 2018):

$$\begin{aligned} \text{Main effect: } m(x) &= 2^{-1}\mathbb{E}[Y^{(0)} + Y^{(1)}|X = x], \\ \text{Treatment effect: } \tau(x) &= \mathbb{E}[Y^{(1)} - Y^{(0)}|X = x], \\ \text{Treatment propensity: } e(x) &= \mathbb{P}[W = 1|X = x]. \end{aligned}$$

The predictor variables X_i are sampled from a uniform distribution as seen in Equation (3.1) (Athey et al., 2019). Treatment assignment $W_i \in \{0, 1\}$ is determined by sampling from a Bernoulli distribution in which the treatment propensity gives the probability for the assignment as seen in Equation (3.2) (Athey et al., 2019). The response given the predictor variables and the treatment assignment can be seen in Equations (3.3) (Athey et al., 2019) and (3.4). First, for the case where the response is based on the normal distribution and second where it is based on the Weibull distribution. In this special case here, the Weibull distribution with shape parameter $k = 1$ reduces actually to an exponential distribution. The standard deviation (as used in the normal distribution) was set to 1 for all setups.

$$X_i \sim U([0, 1]^p) \tag{3.1}$$

$$W_i | X_i \sim \text{Bernoulli}(e(X_i)) \tag{3.2}$$

$$Y_i | X_i, W_i \sim N(m(X_i) + (W_i - 0.5)\tau(X_i), 1) \tag{3.3}$$

$$Y_i | X_i, W_i \sim \text{Weibull}(1, \exp(m(X_i) + (W_i - 0.5)\tau(X_i))) \tag{3.4}$$

For generating the training data sets a total of 24 different setups as seen in Table 3.1 are used with variation in the main effect $m(x)$ and the treatment propensity $e(x)$. Four different functions for the treatment propensity $e(x)$ are paired with three different functions for the main effect $m(x)$, while the treatment effect function

$$\tau(x) = \zeta(x_1)\zeta(x_2)$$

is based on a $\zeta(x_i)$ function of x_1 and x_2. This function remains the same for all different setups through this thesis and is defined as

$$\zeta(x_i) = 1 + \frac{1}{1 + e^{-20(x_i - 1/3)}}, \text{where } i = \{1, 2\}.$$

To see how the random forest methods react to orthogonality in the predictor variables, $e(x)$ and $m(x)$ use either x_1 or x_3 as basis for calculations. Based on x_1 a dependent setting is tested (also x_2 could be used), because the treatment effect is also based on x_1 (and x_2), while x_3 is used for the orthogonal setting.
Simulations vary in terms of sample size $n = \{100, 250, 500, 1000\}$ as well as the dimension $d = \{10, 50, 500\}$ of the covariate matrix X. The maximum dimension 500 is set to test the case when there are more predictor variables than observations. Of every data set, 50 replicas are generated to then average the measured mean squared errors. After

aggregation (over all combinations of n and d) of the results, over the 50 replicas, there are always 12 mean squared errors for every single one of the 24 different setups.
Because there are 24 different setups and 12 possible combinations of n and d ($n = 4$ times $d = 3$) replicated 50 times, the total number of learn data sets generated is 14'400. Each of these data sets consists of the following three parts:

- Covariate matrix X with size n and dimension d.
- Vector of responses Y with length n.
- Vector of treatment indicators W (which are either 0 or 1 for treatment and control) with length n.

Table 3.1: The 24 different setups of the data generating process with different combinations of treatment propensity and main effect function. $\beta_{a,b}$ defines the β-density with the shape parameters a and b. Normal / Weibull are in reference to the distribution on which the conditional outcomes $Y_i|X_i, W_i$ are based.

Setup Normal / Weibull	$e(x)$	$m(x)$
1 / 13	0.5	0
2 / 14	0.5	$2x_1 - 1$
3 / 15	0.5	$2x_3 - 1$
4 / 16	$\frac{1}{4}(1 + \beta_{2,4}(x_1))$	0
5 / 17	$\frac{1}{4}(1 + \beta_{2,4}(x_1))$	$2x_1 - 1$
6 / 18	$\frac{1}{4}(1 + \beta_{2,4}(x_1))$	$2x_3 - 1$
7 / 19	$\frac{1}{4}(1 + \beta_{2,4}(x_3))$	0
8 / 20	$\frac{1}{4}(1 + \beta_{2,4}(x_3))$	$2x_1 - 1$
9 / 21	$\frac{1}{4}(1 + \beta_{2,4}(x_3))$	$2x_3 - 1$
10 / 22	$\sin(2\pi x_3)/4 + 0.5$	0
11 / 23	$\sin(2\pi x_3)/4 + 0.5$	$2x_1 - 1$
12 / 24	$\sin(2\pi x_3)/4 + 0.5$	$2x_3 - 1$

The sinus function for setups 10–12 as well as 22–24 is extended with $(/4 + 0.5)$ because $e(x)$ is further used as a probability as seen in Equation (3.2), to determine who is treated and who is not (treatment indicator W). With this scaling the former range $[-1, 1]$ of the values returned by the sinus function is restricted to $(0, 1)$ to be used as a probability.

Furthermore, this exact term was chosen to make the resulting value range similar to the other $e(x)$ functions based on the beta distribution. The reason why 0 and 1 are not included is, that it lead to computational problems inside the transformation function. Specifically for lower dimensions the mean squared error started to increase as the number of observations increased (while the dimension was fixed) which made little sense, because a decrease would have been expected.
For testing the estimations from the training data sets, a single test set is generated. The test matrix X_{test} has $n = 10'000$ and $d = 500$. For causal forest it contains only the predictor variables, while for transformation forest a treatment indicator $W = 1$ has to be added to the matrix.
The output of this data generating process consists of the following three parts:

- Argument matrix with 14400 rows and 4 columns with information about all possible combinations of setup, number n, dimension d, and replica count 1–50.
- List of 14400 learn data sets each containing X, Y, W.
- Test data set X_{test}.

3.2 Setup

First, the setup for the causal forest is discussed followed by the transformation forest setup. The full code for causal forest can be found in Section A.5.2, and for transformation forest in Section A.5.3.
It has to be differentiated between cases where the treatment propensity is constant at 0.5 and cases where it is not. If the treatment propensity is constant, the implementation with the corresponding `R`-functions differs. For causal forest, two different settings were tested (with and without honest tree splitting) while for transformation forest only one was taken into account.
The `causal_forest()` (Tibshirani et al., 2018) function is used for the similar named procedure. Specifically the following settings are put in to place (for all setups with $e(x) \neq 0.5$):

```
causal_forest(X = as.matrix(data[, grep("^X", colnames(data))]),
              Y = data$y, W = (0:1)[data$trt], num.trees = 250,
              sample.fraction = 0.632, ci.group.size = 1,
              min.node.size = 20, mtry = mtry,
              honesty = honestyFactor, W.hat = NULL)
```

Additionally, to take the underlying distribution of the conditional response in to account, a transformation on Y is done beforehand. Because the causal forest assumes a symmetric distribution, it will not be able to fit the Weibull (i.e. exponential) distribution properly, which is used for setups 13–24. Hence, for these setups, the Y responses are log-transformed.
In a first step, the `num.trees` is set to $B = 250$, while in a second step the `num.trees` is set to $B = 2000$ (in a reduced setting of only 25 replicas) to match the setup of Wager and Athey (2018).

The `sample.fraction` is set to 0.632 to match the setup in the transformation forest case. This is the size of the random subsample that is drawn without replacement from the training data to build each tree. For every tree a new subsample is drawn. Under the honesty criteria, this leads to a further split. Hence under honesty the sample fraction to place the splits, as well as the fraction for the within-leaf estimation, is $0.632/2 = 0.316$ of the training sample. As a direct consequence from setting the sample fraction > 0.5 the `ci.group.size` argument had to be set to < 2, which means that no confidence intervals are provided.
The `min.node.size` argument refers to the minimum number of observations in each tree leaf. This is set to 20 to match the `minbucket` argument for transformation forests. This value was chosen, to ensure enough observations in each node for fitting of the regression model in the transformation forest case.
The number of variables tried for each split, denoted with the `mtry` argument, is set as a floored square root of d (which is the dimension of the covariate matrix X). This is in accordance with the same setting in the classical `randomForest()` function.
With the `honestyFactor` argument the two different setups for the causal forest are defined. When the honesty factor is set to `TRUE` honest tree splitting is incorporated (for consistency this setups will be represented by `CF(honest)` in the rest of this thesis). For `FALSE` no honest tree splitting is used (this setting is represented by `CF(dishonest)` going forward)
The `W.hat` argument is needed, to consider the treatment propensity if it is known. As mentioned before, $e(x)$ is used as the probability for treatment in a Bernoulli distribution. Hence $\hat{W}(x)$ is the (known or estimated) probability for treatment, which is subtracted from the treatment indicator W in a sub step inside the `causal_forest` function. This leads to a so called centered treatment indicator $W_{\text{centered}} = W - \hat{W}(x)$. For the cases where $e(x)$ is known to be $= 0.5$, the argument has to be defined as `W.hat` $= 0.5$. If this is not done, then the propensity will be estimated inside the function, which will lead to a higher uncertainty for the estimates. If the treatment propensity is unknown then the `W.hat` argument can be omitted (or set to `NULL`), because $e(x)$ will be estimated inside the function with $\hat{W} = W \sim X$.
For the transformation forest the setup is a bit more extensive because the procedures from `causal_forest` should be matched as close as possible. This means, that W_{centered} has to be calculated in a separated step before fitting of the transformation forest to the training data.
Before that, the data needs to be brought into a convenient form for the transformation forest function. Once a normal linear model `Lm` is fitted for the case where the outcomes are based on the normal distribution as seen in Equation (2.6) and once a Weibull model as seen in Equation (2.7) with `Survreg` for the Weibull based outcomes.

```
if (attributes(data)$truth$mod == "normal") {
        m <- as.mlt(Lm(y ~ trt, data = data))
    } else {
        m <- as.mlt(Survreg(y ~ trt, data = data))
    }
```

The discrimination between cases where $e(x)$ is known to be equal to 0.5 and where it is

not has to be done, similar to causal forests. For the case where the treatment propensity is constant no further adjustments to W are necessary.
For the cases where the treatment propensity is not constant the treatment probabilities $\hat{W}(x)$ have to be estimated and then subtracted from the treatment indicator W. This is achieved with fitting an ordinary random forest model with W as outcome and X as predictors. Afterwards the predicted treatment probabilities $\hat{W}(x)$ are subtracted from the original treatment assignments W to arrive at W_{centered}:

```
#Calculate treatment probabilities.
rf <- randomForest(trt ~ ., data=data[,-which(colnames(data)=="y")],
                   ntree = 250)
#Subtract  treatment probability from treatment indicator.
data$trtA <- (0:1)[data$trt] - predict(rf, type = "prob")[,2]
```

After this first fit, the output models can be given over to the `traforest()` function which fits a transformation forest. In specific, the following settings are used:

```
tf <- traforest(m, formula = y | trtA ~ ., data = data, ntree = 250,
                minbucket = 20, mtry = mtry,
                control= ctree_control(teststat = "Quadratic",
                                       testtype = "Univariate",
                                       mincriterion = 0,
                                       saveinfo = FALSE))
```

Similar as for causal forest, `ntree` is set to $B = 250$ at first.
The `minbucket` argument as well as the `mtry` correspond to their equivalents in the causal forest and will not be discussed again.
The test statistic type (which is applied for variable selection) is set to `Quadratic`. The second available option `Maximum` is not taken into consideration in this extended setting. The difference between `Maximum` and `Quadratic` lies in their approach in dealing with the scores in the nodes. In the `Maximum` setting every single one of the multidimensional scores is taken into account separately, while with `Quadratic` all will be brought into a quadratic form, which makes the calculation simpler. Together with the two settings from before with and without honesty criteria three settings is reached. Each of them is used on each simulated training data set and evaluated with X_{test}.
Further specified is the univariate distribution as means to compute the test statistic distribution. The minimal test statistic value `mincriterion` to implement a split is set to zero. Which means, that 0 must be exceeded by the value of the test statistic (or 1-p-value). The last argument `saveinfo` refers to a possibility that additional information about variable selection can be stored or not. This is not needed and therefore set to `FALSE`.
The output of the transformation forest is a matrix containing the estimated intercept (the main effect $m(x)$), $\hat{Y}$ and treatment effect $\hat{\tau}(x)$. Because only the main- and treatment effect is of interest a reparametrization needs to be done to get rid of the Y. This is done through dividing both effects through the $\hat{Y}$ values.
In short, the simulation is setup as following. Three different settings are implemented of which two are based on causal forest (`CF(honest)`, `CF(dishonest)`) and one on transformation forest (`Quadratic`). The evaluation was carried out on two extensive data sets,

once with 50 replicas of each data set and once with 25. For the majority of this work the findings correspond to the data set with 50 replicas, while for some parts (influence of tree number) a data set with 25 replicas was used. While for the less extensive data set tree numbers of 250 and 2000 were tested only the case with 250 was looked at for 50 replicas.
The simulations are implemented in `R` (R Core Team, 2017), using base packages, as well as `trtf` (Hothorn, 2018*b*), `tram` (Hothorn, 2018*a*), `grf` (Tibshirani et al., 2018), `randomForest` (Liaw and Wiener, 2002) and `future.apply` (Bengtsson, 2018). The first two packages are used for the transformation forest, while `grf` contains the causal forest implementation. The `future.apply` package contains a very simple implementation for parallel processing of basic `apply` functions by just adding `future_` in front.

3.3 Results

In this section the following research question is answered:

- What are the specific performance-wise differences, dependent on different data generating processes based on different main effect– and treatment propensity functions, between casual– and transformation forests?

The successful reproduction of some parts of the simulation results from Wager and Athey (2018) can be found in Table A.1 and Figure A.1 in the appendix. They are not shown in this section because they do not offer any new results.
The MSE of the three methods (depending on combinations of d and n of the covariate matrix X) for setups 1–6 is displayed in Figure 3.1, for setups 7–12 in Figure 3.2, for setups 13–18 in Figure 3.3 and for setups 19–24 in Figure 3.4. Furthermore, aggregated MSEs based on the different functions of the setups as well as settings can be seen in Table A.3 in the appendix.
Under the simulation setup in Section 3.2 the true treatment effect has a mean of 2.77 with a variance of 0.98.

3.3.1 Simulation Results

If Table A.3 is consulted, a clear difference can be seen between the MSE values for normal– and Weibull based conditional responses. In specific the mean MSEs for setups 13–24 are between 2.67% for `CF(dishonest)` and 9.26% for `TF` higher than for setups 1–12. Furthermore, if the single setups are looked at, there is no instance where the Weibull (i.e. exponential) based setups achieve a better fit than the normal based setups. Overall the mean MSE of normal based outcomes is 0.73 with variance 0.09, while 0.76 for Weibull based (with $k = 1$) with variance 0.09.
For setups 1–12 the best overall fit is achieved with a MSE of 0.54 for setup 1 with transformation forest, while for causal forest the best fit can be seen for setup 10 with 0.68 at setting `CF(dishonest)`. On the other side of the spectrum, the least good is 0.92 for setup 5 with `CF(honest)` for causal forest, while for transformation forest there is 0.64 for setup 6.
For setups 13–24 it is almost identical. Best fit for setup 13 (which corresponds to setup 1) by transformation forest with a MSE of 0.58, while for causal forest setup 19 with

0.7, which is different to the first twelve (although setup 22 is a close second). The least favorable has a value of 0.94 for `CF(honest)` and setup 17 (which corresponds to setup 5) for causal forest, while transformation forest is at 0.71 for setup 20.
The following statements can be made in regard to Figure 3.1 up to Figure 3.4:

- Causal forests with honest tree splitting (setting `CF(honest)`) performs in most cases not as good as transformation forests, while sometimes equal to `CF(dishonest)`. This seems to have its roots in the fact that double sample trees (Wager and Athey, 2018), which incorporate honest tree splitting, are more sensitive to changes in the treatment effect. Although treatment heterogeneity is present in all setups, there is, in the majority of setups, additional confounding in $e(x)$ and $m(x)$. It is surprising that even for setup 1, where the first simulation setting of Athey et al. (2019) with no confounding but with treatment heterogeneity, is exactly mimicked, `CF(honest)` is still not performing better than `CF(dishonest)`.

- There is strong evidence that transformation forests achieve a better fit than causal forests without honest tree splitting for low dimensional data with a small number of observations (10:100 and 10:250) with a p-value of < 0.0001. Furthermore, strong evidence (p-value < 0.0001) for a consistent improvement of transformation forests against causal forests can be seen for all setups 1–12 and for most setups 13–24, with dimension 50.

- For high dimensional data sets with number of observations 100–1000 and for cases with more dimensions than predictor variables, only minor differences arise between transformation forests and causal forests on `CF(dishonest)`. This is as expected, because when number of observations increase the differences between different methods tend to diminish.

- Setting `CF(dishonest)` tends to have a better prediction for low dimension paired with high observation numbers (10:1000), except for setups 1–3 (and their equivalents 13–15). These six setups are all based on a constant treatment propensity of 0.5 which seems the transformation forest can handle better.

- The highest mean variance in the estimates has `CF(dishonest)` with 0.10 followed by `TF` with 0.08 and `CF(honest)` with 0.06. Although through visual inspection the IQR of the box plots is higher for `TF`, causal forests tend to have more outliers (especially for smaller data sets).

- Fits with the conditional response based on a Weibull (i.e. exponential) distribution always have a higher MSE then the normal based ones.

- The best fit is achieved for low number of dimensions paired with a high number in n. In this framework this would be 10:1000, which is as one would expect.

As seen in Table 3.2, there are some major differences in the computation time of the three different methods. The fit of transformation forest takes per data set a mean of $16.18s$ for setting `Quadratic`. The fit of causal forest takes per data set in the mean 0.32 seconds with $0.31s$ for `CF(honest)` and $0.34s$ for `CF(dishonest)`. Hence, the ratio between the mean computation time of transformation forest against causal forest is

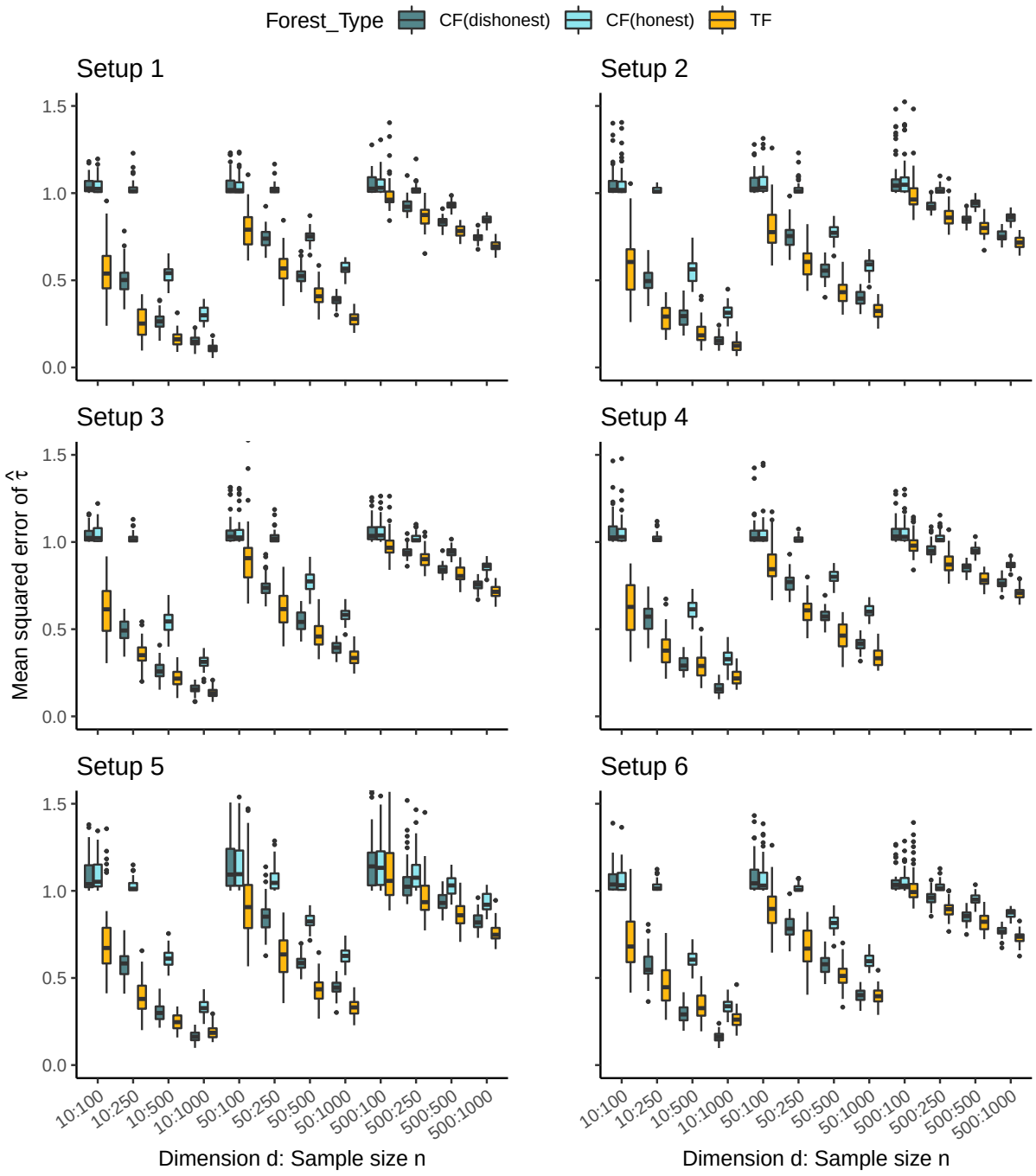

Figure 3.1: Mean MSE of $\hat{\tau}$ over 50 replicas for settings one to six for normal based outcomes, dependent on the twelve combinations of dimension and sample size. The settings `CF(dishonest)` and `CF(honest)` refer to the honesty criteria for causal forests while `TF` refers to the transformation forest.

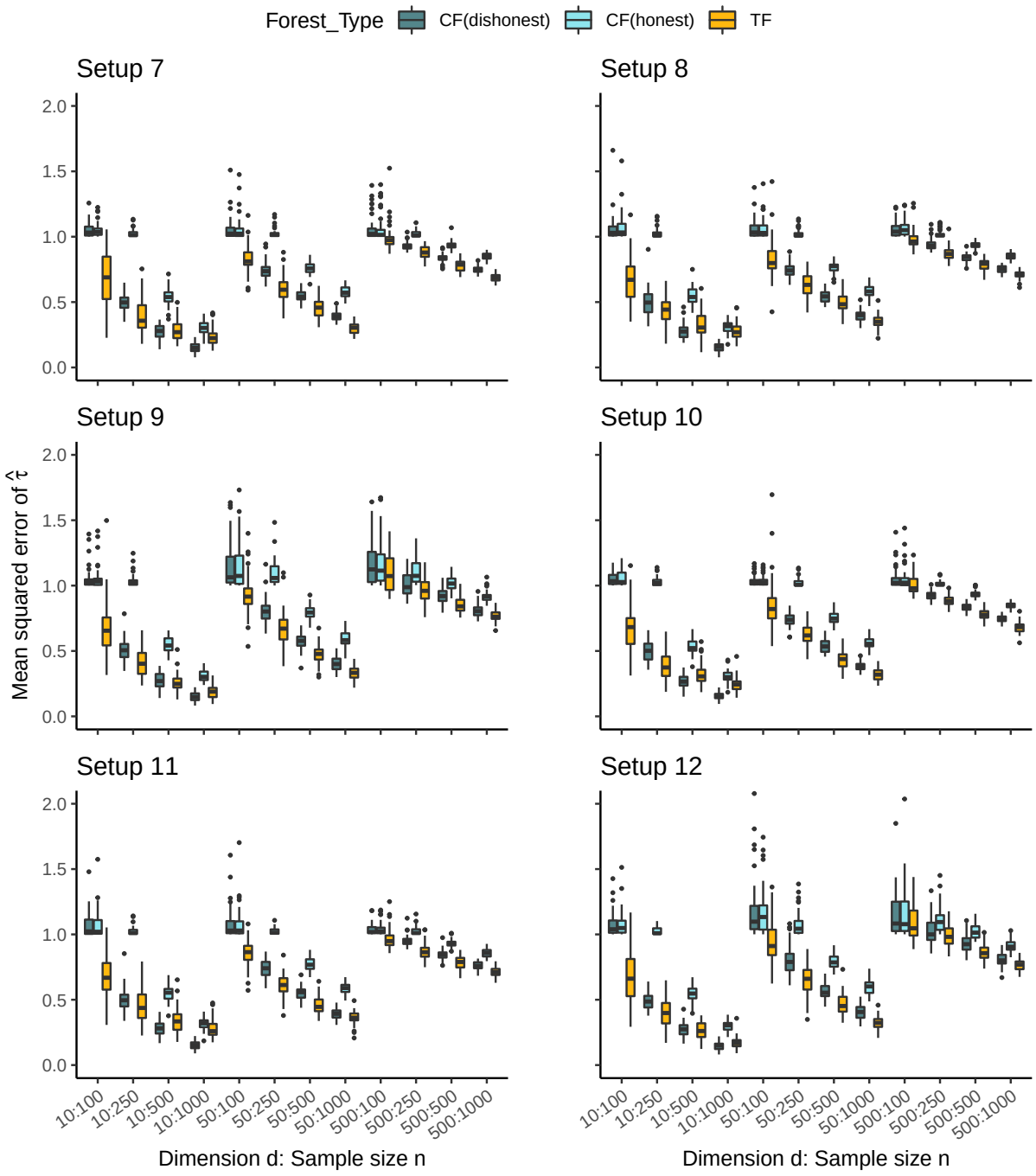

Figure 3.2: Mean MSE of $\hat{\tau}$ over 50 replicas for settings seven to twelve for normal based outcomes, dependent on the twelve combinations of dimension and sample size. The settings `CF(dishonest)` and `CF(honest)` refer to the honesty criteria for causal forests while `TF` refers to the transformation forest.

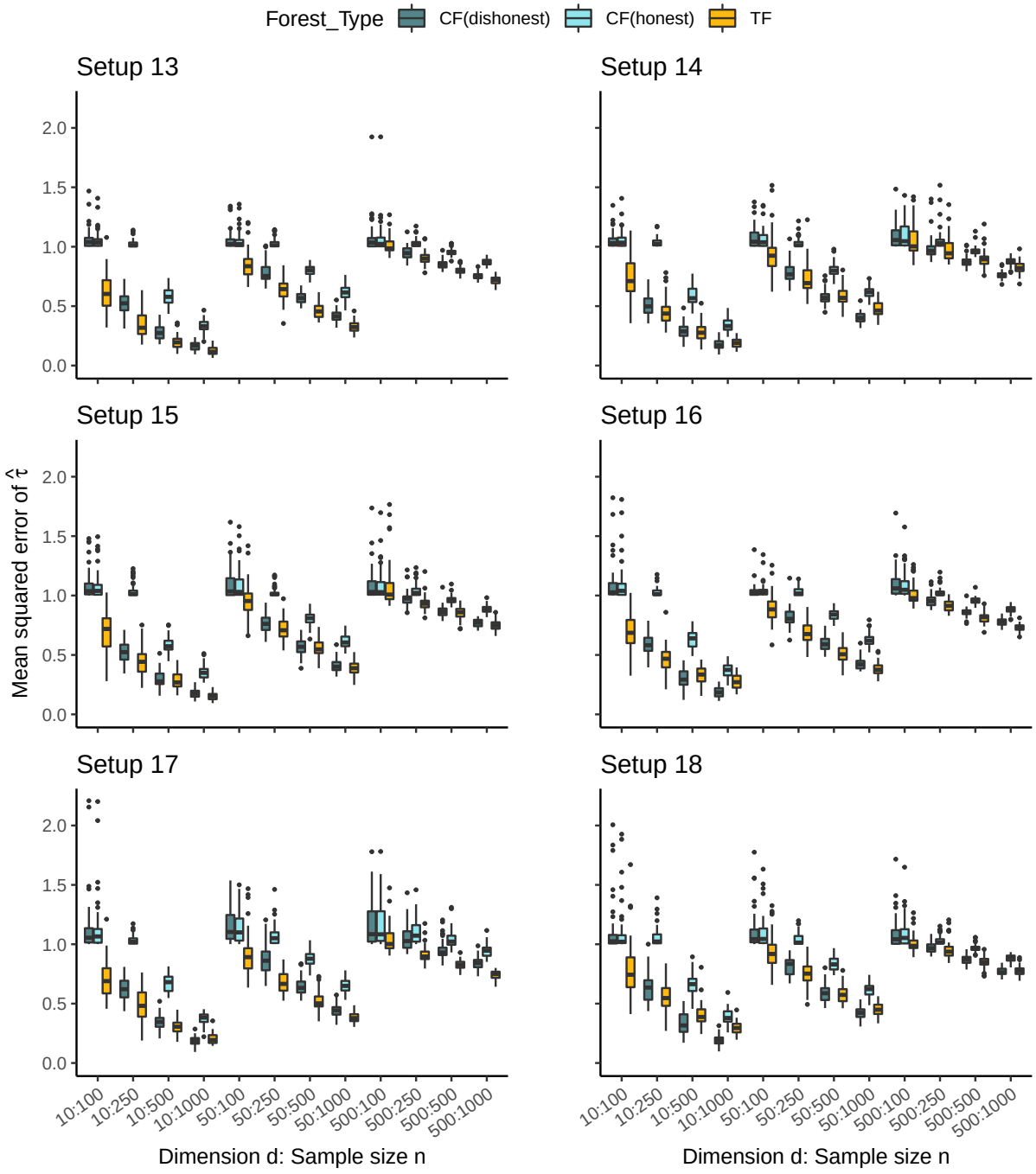

Figure 3.3: Mean MSE of $\hat{\tau}$ over 50 replicas for settings 13 to 18 for Weibull ($k = 1$) based (and logarithmic transformed) outcomes, dependent on the twelve combinations of dimension and sample size. The settings `CF(dishonest)` and `CF(honest)` refer to the honesty criteria for causal forests while `TF` refers to the transformation forest.

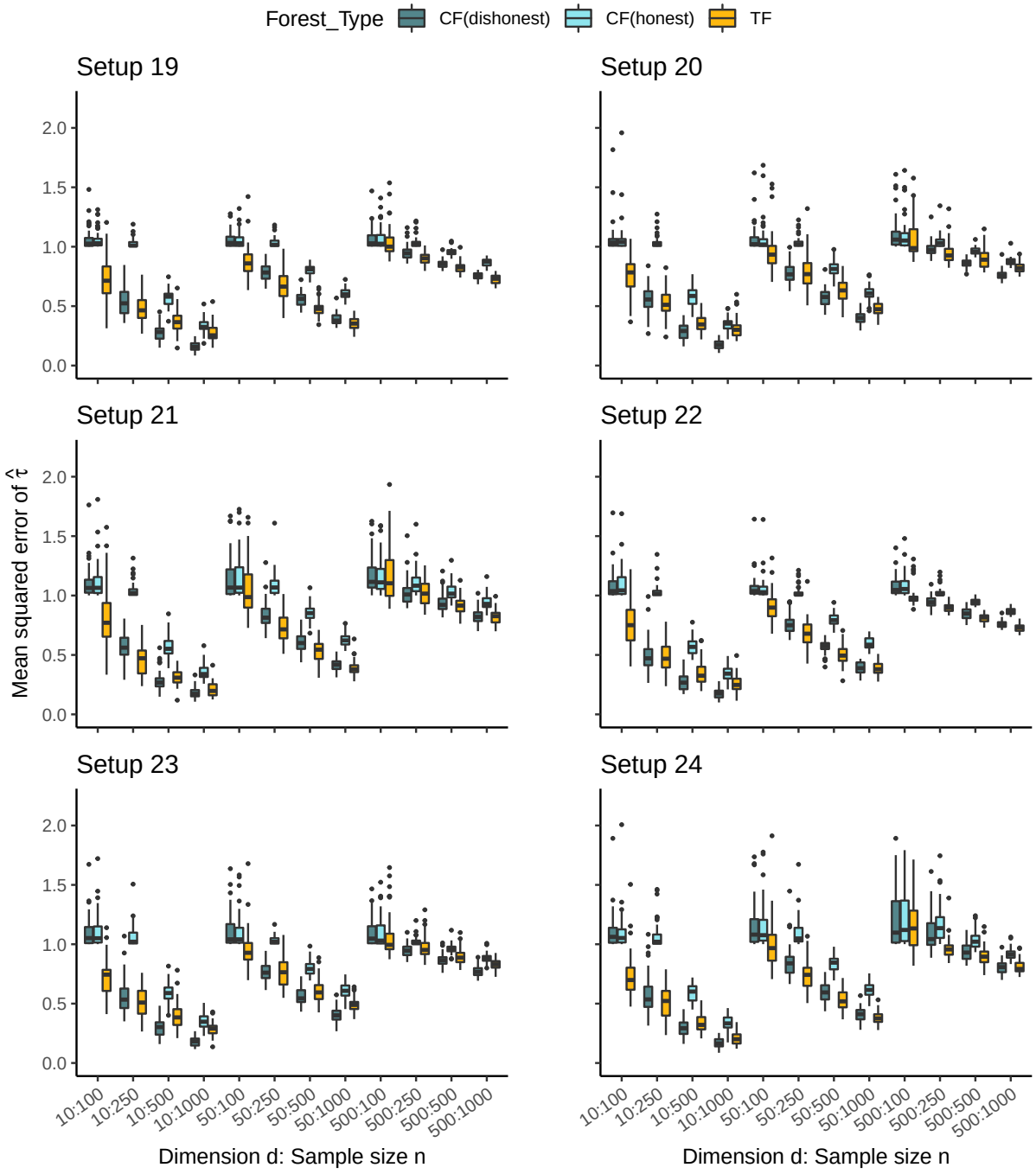

Figure 3.4: Mean MSE of $\hat{\tau}$ over 50 replicas for settings 19 to 24 for Weibull ($k = 1$) based (and logarithmic transformed) outcomes, dependent on the twelve combinations of dimension and sample size. The settings `CF(dishonest)` and `CF(honest)` refer to the honesty criteria for causal forests while `TF` refers to the transformation forest.

49.78. This means, that causal forest implementation is around 49.78 times faster than the transformation forest one for this particular setting.

Table 3.2: Computation time in hours for the whole training data set, once for 25 replicas and once for 50 replicas with 250 trees for both. Parallel computing was initialized on a server with a 12 core processor.

Number of trees:Replicas	TF	CF(honest)	CF(dishonest)	Total
250:50	64.7	1.24	1.35	67.29
250:25	37.6	0.91	0.67	39.18

It has to be noted first, that these numbers are not robust. The measured times are based on only one single run each, since for obvious reasons multiple runs, just to check the computation time, were not in the scope of this thesis. Second, a difference was to be expected, because the transformation forest estimates two parameters simultaneously, which is computationally more demanding then the causal forest case with its single parameter estimation.

3.3.2 Influence of Treatment Propensity

The influence of the four different treatment propensity functions (as seen in Table 3.1) on the overall fit is displayed in Figure A.2 in the appendix. Setting `CF(honest)` was intentionally left out because one would not want to use it for a fit in this setup, hence there is no benefit in discussing the behavior towards different treatment propensities. Through the aggregation process (every line in the figure represents three setups which were already aggregated over 50 replicas each) most of the variation will be eliminated. If a difference is visible, there is a high degree of certainty that the difference did not occur by chance. Furthermore, the possible confounding influence of the different main effect functions $m(x)$ does not impose a problem. Because every $m(x)$ function was paired with every single one of the $e(x)$ functions, a direct comparison can be made where the effect of $m(x)$ is already taken into account.
There seems to be a difference in the fit for transformation forests for normal- and Weibull (i.e. exponential) based conditional outcomes for data sets with $d = 10$. If $e(x) = 0.5$ the fit is better (with a lower MSE) than for cases with $e(x) \neq 0.5$. This corresponds to setups 1–3 and 13–15. Non the less this is the only visible difference that can be directly traced back to the difference in treatment propensities.
For causal forest there does not seem to be any evidence that the treatment propensity has any influence on the accuracy of the fit.

3.3.3 Influence of Main Effect

The influence of the three different main effect functions can be seen in Figure A.3 in the appendix. The same reasoning as to why one is able to compare the fit applies as in Subsection 3.3.2.

Similar as for the treatment propensity, there seems to be a difference for transformation forests, while causal forest do not show any kind of significant deviation between the three different main effect functions. For transformation forests small differences arise for higher dimensions and numbers of observations (above 10:1000). The best fit is achieved for instances where the main effect is zero, while setups where the main effect is orthogonal ($m(x) = 2x_3 - 1$) lead to a decrease in precision. But in general these are rather small differences as well as they do not hold for all the different instances of dimension and number of observations of the test data set.

3.3.4 Influence of Number of Trees

Because of computational reasons a run with a fit over 2000 trees (as in Wager and Athey (2018)) was not feasible in time for the case with 50 replicas. Non the less some statements can be made about the influence of the number of trees for a reduced setting. For an earlier fit, only 25 replicas were used where it was possible to fit both tree counts.
If Table A.2 is consulted it can be seen, that the fit improves for a majority of setups (as denoted in the positive numbers of the mean difference) if the number of trees fitted is increased eightfold, which is as expected. In this context mean difference is the difference in the MSE of a fit with 250 trees minus the fit with 2000 trees. Hence if positive numbers occur, the fit has improved. For causal forest there is a small increase for `CF(dishonest)` of below a halve percent, while for `CF(honest)` there is a mixed effect. For both settings one can also see a decrease in precision (meaning negative mean differences), especially for the setups based on Weibull(i.e. exponential). For transformation forest there is a mean gain of over 7% which is distributed very uneven. While setups (10–12 and 22–24) with treatment propensity based on the sinus function gain between 14% and 37%, the rest show improvement in the low one digit area (while one is even negative).
It can be said, that it has some merits to increase the number of trees in the transformation forest setting, but it depends on the values of the treatment propensity. While it is useful to increase for sinus based $e(x)$ it is not advantageous for constant treatment propensities like $e(x) = 0.5$. The adverse effect of this measure will be an increase in computation time by a factor of 3.97. For causal forests `ntree = 250` is sufficient, and an increase to `ntree = 2000` not necessary with the simulation setup used in this thesis.

Chapter 4

Conclusion

This thesis gives a detailed comparison between causal forests and transformation forests based on a simulated setting. An extensive set of combinations of treatment heterogeneity, paired with main effect-and treatment propensity functions, with and without orthogonal dependencies on the treatment effect function, is investigated. Both random forest methods show for certain settings a good performance on the overall prediction of a heterogeneous treatment effect. The general behavior of the fit is, that with increase of the number of observations the mean squared error goes down, which is as expected. Furthermore, if the outcomes are based on a skewed distribution one has to take appropriate steps before running the forests. If this is not done, the methods will not produce any viable predictions. Additionally, in the transformation forest case, the treatment propensity should lie in $(0, 1)$.
The major driver behind overall MSE differences is first and foremost the extent of the test data set, as well as (in a smaller part) the distribution of the conditional outcomes. Different treatment propensity- and main effect functions only have a diminishing overall influence, while for certain setups differences can arise. The same can be said for number of trees fitted, where it does not seem beneficial to increase for causal forest, it can have some merits in the transformation forest setting. It seems, that despite the substantial differences in the setups, transformation forests are up to around 40% better (than causal forests) in their predictive errors for low dimensions paired with a low number of observations. Because the economical aspect should not be overlooked, which means that smaller data sets are cheaper to obtain, it seems that transformation forests tend to be the way to go.

4.1 Outlook

Throughout the work on this thesis, the following areas came to mind, where further investigation on the topic could be carried out. Once a more in depth analysis of the optimal number of trees per forest, once a comparison to additional random forest based methods and additionally a fit with Weibull based outcomes where $k \neq 1$.
The investigation up on the number of trees was done very coarsely with only two instances of numbers (i.e. 250 an 2000). Although, some statements could be made, the question about what a sufficient number of trees would be cannot be answered with the data at hand. It may also depend on the corresponding setup. For example for setup one

with only treatment heterogeneity there may be a need for less trees then in a more complex setting with additional confounding. Especially for transformation forest, there is a computational argument to be made that the number of trees should be kept at a minimum.
As already mentioned, Lu et al. (2018) already did a comparison of a variety of different random forest methods (to be exact, seven), including causal forests. In a next step, the simulation experiments could be reproduced and run with causal forest, to see if the same outcome could be achieved. If so, transformation forests could be added to see how they perform in comparison to the other methods, to achieve a wider comparison than just between two methods. The level of insights one might gain from this will be less than with this current setup, since Lu et al. (2018) only use three different simulation models with two settings for sample size. Which leads to six cases which can be investigated (compared to the 24 in this thesis).
Because the Weibull model used in this thesis has rate parameter equal one the model is actually exponential. Since both random forest methods are better equipped to handle an exponential case, where $\mathbb{E}(X)$ is a simple function to estimate, a case with an actual Weibull distribution should be tried.

Bibliography

Athey, S., Tibshirani, J. and Wager, S. (2019), 'Generalized random forests', *The Annals of Statistics* **47**(2), 1148–1178.

Bengtsson, H. (2018), *future.apply: Apply Function to Elements in Parallel using Futures*. R package version 1.0.1.
URL: *https://CRAN.R-project.org/package=future.apply*

Breiman, L. (2001), 'Random forests', *Machine Learning* **45**(1), 5–32.

Efron, B. (2014), 'Estimation and accuracy after model selection', *Journal of the American Statistical Association* **109**(507), 991–1007.

Foster, J., Taylor, J. and Ruberg, S. (2011), 'Subgroup identification from randomized clinical trial data', *Statistics in medicine* **30**, 2867–80.

Hastie, T., Tibshirani, R. and Friedman, J. (2001), *The Elements of Statistical Learning*, Springer Series in Statistics, Springer New York Inc., New York, NY, USA.

Hothorn, T. (2018*a*), *mlt: Most Likley Transformations*. R package version 0.2-3.
URL: *https://CRAN.R-project.org/package=tram*

Hothorn, T. (2018*b*), *trtf: Transformation Trees and Forests*. R package version 0.3-3.
URL: *https://CRAN.R-project.org/package=trtf*

Hothorn, T., Lausen, B., Benner, A. and Radespiel-Tröger, M. (2004), 'Bagging survival tree', *Statistics in Medicine* **23**(1), 77–91.

Hothorn, T., Steffen, V., Seibold, H. and Korepanova, N. (2019), Survival forests under test: Impact of the proportional hazards assumption on prognostic and predictive models for als survival, Technical report. arXiv:1902.01587v1.

Hothorn, T. and Zeileis, A. (2017), Transformation forests, Technical report. arXiv:1701.02110.

Liaw, A. and Wiener, M. (2002), 'Classification and regression by randomforest', *R News* **2**(3), 18–22.
URL: *https://CRAN.R-project.org/doc/Rnews/*

Lu, M., Sadiq, S., Feaster, D. and Ishwaran, H. (2018), 'Estimating individual treatment effect in observational data using random forest methods', *Journal of Computational and Graphical Statistics* **27**(1), 209–219.

Meinshausen, N. (2006), 'Quantile regression forests', *Journal of Machine Learning Research* **7**, 983–999.

R Core Team (2017), *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria.
URL: *https://www.R-project.org/*

Robinson, P. (1988), 'Root-n-consistent semiparametric regression', *Econometrica* **56**(4), 931–54.

Seibold, H., Zeileis, A. and Hothorn, T. (2018), 'Individual treatment effect prediction for amyotrophic lateral sclerosis patients', *Statistical Methods in Medical Research* **27**(10), 3104–3125.

Tibshirani, J., Athey, S., Wager, S., Friedberg, R., Miner, L. and Wright, M. (2018), *grf: Generalized Random Forests (Beta)*. R package version 0.10.0.
URL: *https://CRAN.R-project.org/package=grf*

Wager, S. and Athey, S. (2018), 'Estimation and inference of heterogeneous treatment effects using random forests', *Journal of the American Statistical Association* **113**(523), 1228–1242.

Wager, S., Hastie, T. and Efron, B. (2014), 'Confidence intervals for random forests: The jackknife and the infinitesimal jackknife', *Journal of Machine Learning Research* **15**, 1625–1651.

Appendix A

A.1 Software

R version and packages used to generate this report:
R version: R version 3.4.2 (2017-09-28)
Base packages: grid, stats, graphics, grDevices, utils, datasets, methods, base
Other packages: gridExtra, ggpubr, magrittr, ggplot2, tram, trtf, partykit, mvtnorm, libcoin, mlt, basefun, variables, randomForest, grf, xtable, knitr
This document was generated on Mrz 30, 2019 at 09:38.

A.2 Limited Reproduction of Wager and Athey 2018

The results of some portions of the causal forest simulations from Wager and Athey (2018) were reproduced to a certain degree. Because the scope of this thesis is a comparison with another random forest method, the focus was only on the causal forest, while the additional analysis with k-NN was not taken into account. This reproduction is limited to the MSE and coverage results as seen in Table A.1 and the display (color dependent) of $\tau(x)$ and $\hat{\tau}(x)$ through the causal forest method in Figure A.1.
A different simulation setup was used for this than in the rest of this thesis to match the setup of Wager and Athey (2018) which is as follows:

- Setup: Setup 1 from Table 3.1 with heterogeneity in $\tau(x)$ while main effect and treatment propensity remain constant at 0 and 0.5.
- Procedure: Double sample trees based on the honesty criteria.
- Number of trees: $B = 2000$.
- Sample: The sample size is $n = 5000$ with $d = \{2, 3, 4, 5, 6, 8\}$ and $s = 2500$, where s is the size of the random drawn subsample from $\{1, \ldots, n\}$. This means, contrary to the rest of this thesis, one data set was split into training - and test data compared to the all new test set used in the thesis itself.

This setup describes the one shown in (28) on page 20 in Wager and Athey (2018), which holds for the reproduction of the table. For the figure reproduction (29) (on page 21 of Wager and Athey (2018)) was used with $n = 10'000$ and $d = \{6, 20\}$. The difference from

(29) to (28) stand from the fact that (29) has additional confounding in $m(x)$ and $e(x)$, similar to setup 5 form Table 3.1 with a minor change in the ζ function.
The following function setup was used:

```
causal_forest(X = as.matrix(data[, grep("^X", colnames(data))]),
              Y = data$y, W = (0:1)[data$trt], num.trees = 2000)
```

All other options were left at their default values, like the honesty criteria which is already implemented as `FALSE` (for double sample trees) and does not need to be specified.
To reproduce the coverage, a simple Wald confidence interval was used with a target coverage rate of 95% (because it was not clearly stated in Wager and Athey (2018) what confidence interval was used to calculate coverage). The confidence interval is based on the estimated values and variances $\hat{\tau} \pm q^z_{1-\frac{\alpha}{2}} \sqrt{\text{var}(\hat{\tau})}$ from the causal forest. Then it was simply counted how many times τ_i was covered by the interval gained from the estimated treatment effects $\hat{\tau}_i$.
If the values in Table A.1 are compared it seems that the reproduction was successful towards the MSE while not towards the coverage rate. For the MSE there is a maximal deviation of 50% for dimension six while the other values are either closer or right on point. It has to be said, that these values are all rounded to one significant number to match with the precision of the original mean squared errors. Due to this rounding, the variation was lost. Because it can be assumed that the original numbers were submitted to rounding too, it makes them roughly comparable. This difference between reproduction results and original might be due to differences in the starting numbers of the random algorithm.
A four to seven percentage point difference can be seen between the coverage rates from Wager and Athey (2018) and the reproduction in Table A.1. These differences might arise due to different approaches for the coverage rate calculations. Trend wise, there is the same decrease of 7% between the highest and lowest value, it seems that the behavior is reproducible but not the level.

Table A.1: Comparison between the MSE and the coverage of the causal forest method by Wager and Athey (2018) (page 22, table 2) and the reproduction, with honest tree splitting and heterogeneity in $\tau(x)$. Both aggregated over 25 replicas of a data set with $n = 5000$.

Dimension	MSE		Coverage	
	Original	Reproduction	Original	Reproduction
2	0.04	0.03	0.97	0.93
3	0.03	0.03	0.96	0.92
4	0.03	0.03	0.94	0.89
5	0.03	0.03	0.93	0.89
6	0.02	0.04	0.93	0.86
8	0.03	0.03	0.90	0.86

The reproduction of the figure on $\tau(x)$ and $\hat{\tau}(x)$ seems also to have been successful as seen in Figure A.1. The two figures of $\hat{\tau}(x)$ deviate marginally for $d = 20$ from the ones produced in Wager and Athey (2018) on page 23 while the figures for $\tau(x)$ seem rather congruent to its counterparts. Because the author did not have the original code, the figures from the paper could not be displayed here. Furthermore, the actual color grading is unknown, hence the sharpness of the color levels is a mere approximation.

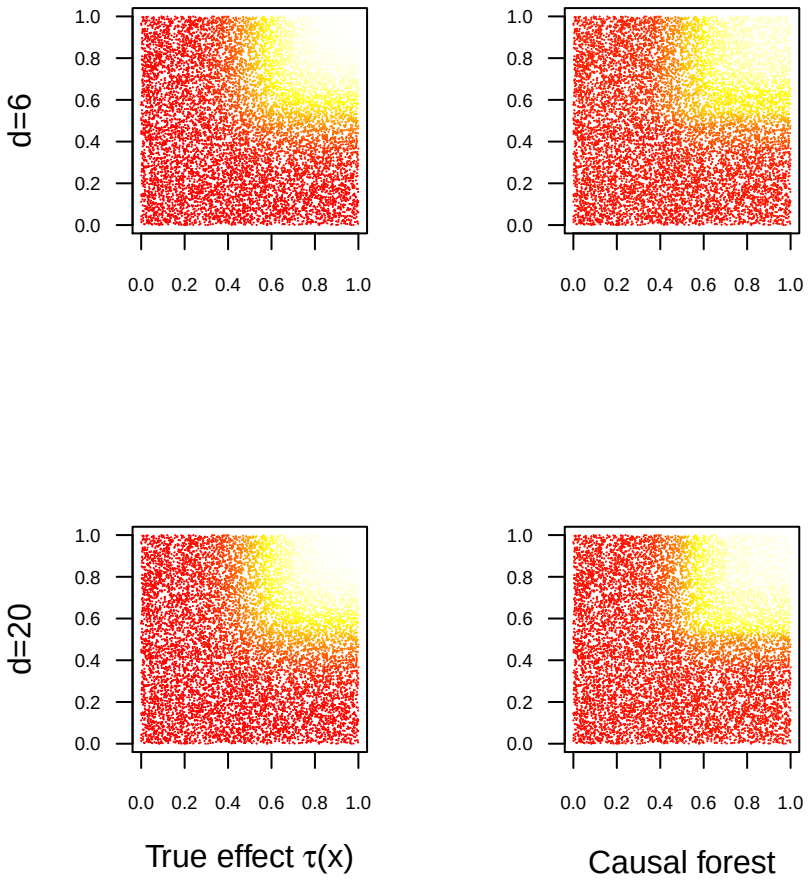

Figure A.1: Reproduction of figure 2 from Wager and Athey (2018), page 23 (without the k-NN method). True treatment effect and estimated treatment effect on $n = 10'000$ data points with dimension $d = 6, 20$ on the basis of setting (29) (Wager and Athey (2018), page 21). The test points are according to values x_1 (on the X-axis) and x_2 (on the Y-axis). The color represents the treatment effect, whereas red stands for a low effect and white for a high one.

A.3 Tables

Table A.2: Difference in the MSE of the fit dependent on the number of trees, where `ntree = 250` is used as base. The columns "Difference" refer to the MSE difference of the fit with MSE(`ntree = 250`) minus MSE(`ntree = 2000`), while the "%" column sets the relation between "Difference" and the MSE of the fit based on 250 trees. Based on 25 replicas of the whole process.

Setup	CF(dishonest)		CF(honest)		TF	
	Difference	%	Difference	%	Difference	%
1	0.0014	0.21	-0.0002	-0.02	0.0026	0.49
2	0.0040	0.58	-0.0008	-0.10	0.0049	0.87
3	0.0025	0.37	0.0014	0.17	0.0049	0.83
4	0.0037	0.53	0.0017	0.20	0.0104	1.72
5	0.0039	0.52	0.0019	0.21	0.0165	2.59
6	0.0030	0.43	0.0008	0.09	0.0091	1.43
7	0.0032	0.47	0.0019	0.23	0.0120	1.97
8	0.0031	0.45	0.0024	0.29	0.0141	2.29
9	0.0054	0.75	0.0001	0.01	0.0227	3.52
10	0.0040	0.60	0.0028	0.34	0.2102	26.27
11	0.0033	0.48	0.0013	0.15	0.1872	23.64
12	0.0005	0.06	0.0018	0.19	0.1063	14.54
13	0.0186	0.63	-0.0086	-0.29	0.0032	0.55
14	0.0659	0.56	0.0544	0.49	0.0049	0.72
15	0.0399	0.56	-0.0045	-0.07	0.0051	0.79
16	0.0065	0.25	0.0053	0.20	0.0163	2.55
17	0.0005	0.01	-0.0126	-0.21	0.0101	1.56
18	0.0064	0.11	-0.0437	-0.81	0.0116	1.65
19	0.0023	0.08	0.0038	0.12	0.0122	1.89
20	-0.0882	-0.71	0.0084	-0.15	-0.0018	-0.25
21	0.0111	0.27	0.0084	0.23	0.0217	3.13
22	-0.0001	-0.00	-0.0070	-0.20	0.3697	37.06
23	-0.0173	-0.14	-0.0711	-0.65	0.3203	31.49
24	-0.0211	-0.46	0.0267	0.61	0.1557	18.66
Mean	0.0026	0.27	-0.0021	0.04	0.0637	7.50

Table A.3: Aggregated MSE for every setup according to causal- and transformation forest, as well as the summary measure for the first- and second twelve. One number represents the mean of every twelve mean squared errors per setup. Based on 50 replicas.

Setups	CF(dishonest)	CF(honest)	TF
1	0.6866	0.8460	0.5398
2	0.6997	0.8608	0.5618
3	0.6916	0.8535	0.5932
4	0.7103	0.8670	0.5971
5	0.7648	0.9172	0.6299
6	0.7149	0.8703	0.6445
7	0.6897	0.8504	0.5912
8	0.6939	0.8549	0.6127
9	0.7348	0.8985	0.6355
10	0.6845	0.8455	0.6007
11	0.6964	0.8565	0.6140
12	0.7365	0.9003	0.6334
13	0.7066	0.8681	0.5817
14	0.7142	0.8773	0.6780
15	0.7187	0.8826	0.6501
16	0.7281	0.8864	0.6427
17	0.7852	0.9370	0.6434
18	0.7431	0.8989	0.6948
19	0.7045	0.8690	0.6431
20	0.7174	0.8788	0.7133
21	0.7598	0.9231	0.7019
22	0.7055	0.8721	0.6439
23	0.7209	0.8848	0.7106
24	0.7660	0.9282	0.6907
1-12	0.7086	0.8684	0.6045
13-24	0.7308	0.8922	0.6662

A.4 Figures

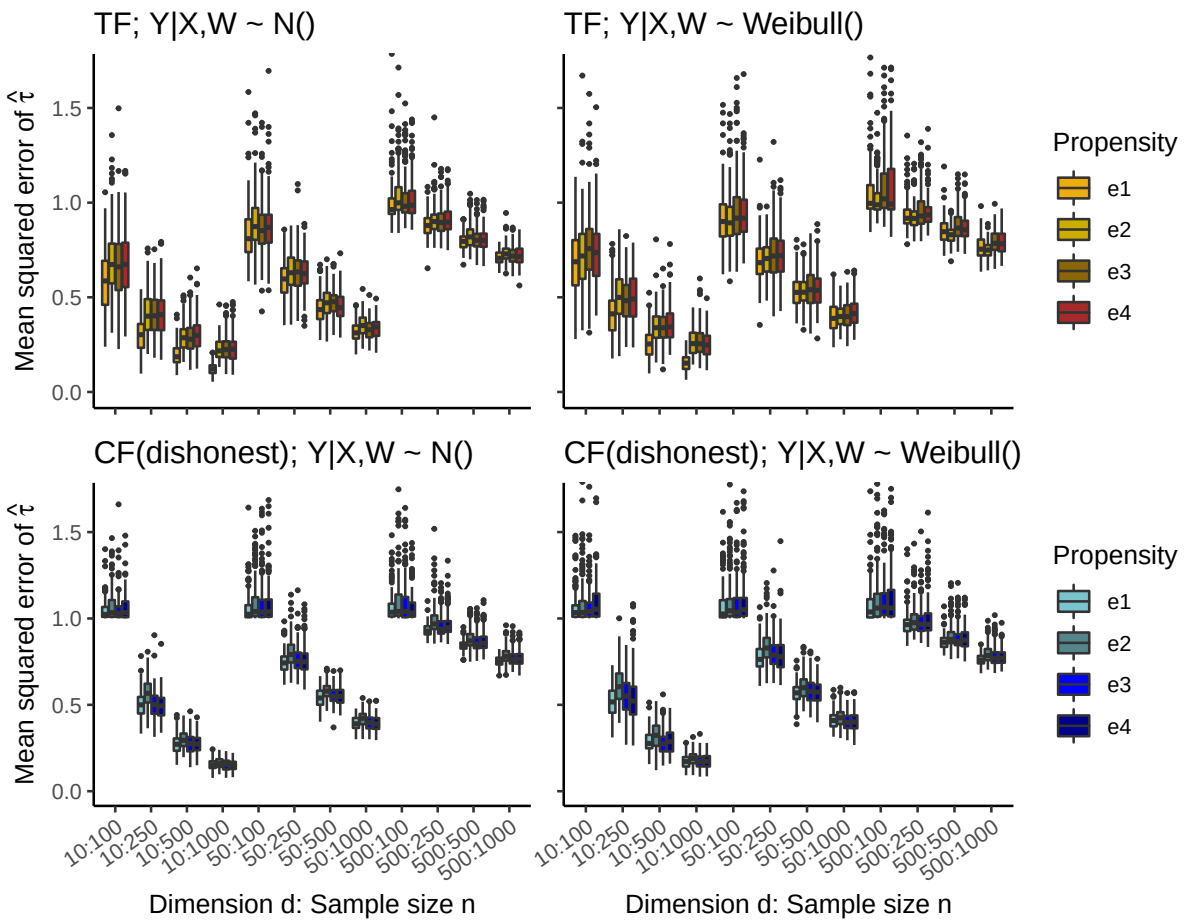

Figure A.2: Mean MSE of $\hat{\tau}$ over 50 replicas of the whole process, aggregated for settings with the same treatment propensity function $e(x)$. Divided towards dependence of the underlying distribution of the conditional response $Y|X, W$, as well as causal forest with `CF(dishonest)` and transformation forest. The following encoding scheme was used: e1: $e(x) = 0.5$, e2: $e(x) = \frac{1}{4}(1+\beta_{2,4}(x_1))$, e3: $e(x) = \frac{1}{4}(1+\beta_{2,4}(x_3))$ and e4: $e(x) = \sin(2\pi x_3)$.

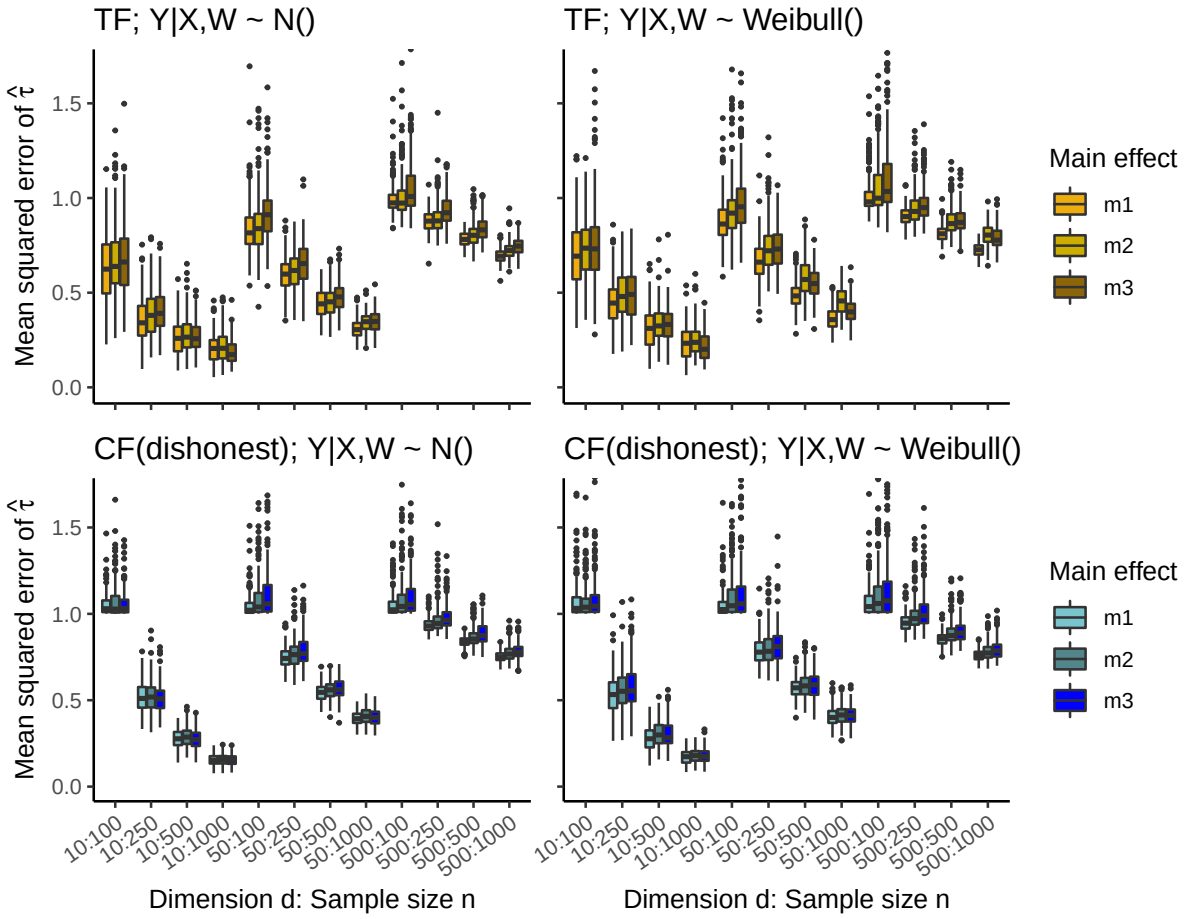

Figure A.3: Mean MSE of $\hat{\tau}$ over 50 replicas of the whole process, aggregated for settings with the same main effect function $m(x)$. Divided towards dependence of the underlying distribution of the conditional response $Y|X, W$, as well as causal forest with `CF(dishonest)` and transformation forest. The following encoding scheme was used: m1: $m(x) = 0$, m2: $m(x) = 2x_1 - 1$ and m3: $m(x) = 2x_3 - 1$.

A.5 Code

A.5.1 Data Generating Process

```
library("future.apply")
plan(multiprocess) #Initialize parallel

#DGP and Simulate.dgp function
dgp<-function(e=0.5, m=0, z=0, sd=1,model = c("normal", "weibull")){

  if (!is.function(efct <- e))
    efct <- function(x) return(rep(e, nrow(x)))
  if (!is.function(mfct <- m))
    mfct <- function(x) return(rep(m, nrow(x)))
  if (!is.function(zfct <- z))
    zfct <- function(x) return(rep(z, nrow(x)))
  #Tau(x) = zeta(x1) * zeta(x2)
  tfct <- function(x) zfct(x[,"X1"])*zfct(x[,"X2"])
  if (!is.function(sdfct <- sd))
    sdfct <- function(x) return(rep(sd, nrow(x)))

  model<-model
  ret <- list(efct = efct, mfct = mfct, tfct = tfct, sdfct = sdfct,
              model = model)
  class(ret) <- "dgp"
  return(ret)
}

simulate.dgp<-function(object, nsim = 1, seed = NULL, dim = 3) {
  if (!is.null(seed))
      set.seed(seed)

  x<-matrix(runif(nsim*dim,0,1),nrow = nsim, ncol = dim)
  colnames(x) <- paste("X", 1:ncol(x), sep = "")

  tX<-object$tfct(x)
  eX<-object$efct(x)
  mX<-object$mfct(x)
  sd<-object$sdfct(x)
  model<-object$model

  trt<-rbinom(nsim, size = 1, prob = eX)    #Wi|Xi ~Bernoulli (e(Xi))

  y <- switch(model,
      "normal" =rnorm(nsim, mean = mX+(trt-0.5)*tX, sd = sd),
      "weibull"=rweibull(nsim, shape = 1, scale = exp(mX+(trt-0.5)*tX))
```

```
  )

  df <- data.frame(x, y = y, trt = factor(trt))
  attributes(df)$truth <- object
  class(df) <- c("simdgp", class(df))
  return(df)
}

#Evaluate the truth
predict.simdgp <- function(object, newdata, ...) {
    atr <- attributes(object)$truth
    atr <- atr[sapply(atr, is.function)]
    sapply(atr, function(f) f(newdata))
}

#Define the different zeta, treatment propensity and main effect
#functions

#Zeta function
zF<-function(x){
  1+1/(1+exp(-20*(x-1/3)))
}

#Treatment propensity functions
eF_x1<-function(x){
  1/4*(1+dbeta(x[,"X1"],2,4))
}
eF_x3<-function(x){
  1/4*(1+dbeta(x[,"X3"],2,4))
}
eFs_x3<-function(x){
  sin(2*pi*x[,"X3"]) / 4 + .5
}

#Main effect functions
mF_x1<-function(x){
  2*x[,"X1"]-1
}
mF_x3<-function(x){
  2*x[,"X3"]-1
}


#Generate the data
setups <- vector(mode = "list", length = 1)
setups[[1]] <-dgp(e = 0.5,    m = 0,     z = zF, sd = 1,model ="normal")
```

```
setups[[2]] <-dgp(e = 0.5,    m = mF_x1, z = zF, sd = 1,model ="normal")
setups[[3]] <-dgp(e = 0.5,    m = mF_x3, z = zF, sd = 1,model ="normal")
setups[[4]] <-dgp(e = eF_x1,  m = 0,     z = zF, sd = 1,model ="normal")
setups[[5]] <-dgp(e = eF_x1,  m = mF_x1, z = zF, sd = 1,model ="normal")
setups[[6]] <-dgp(e = eF_x1,  m = mF_x3, z = zF, sd = 1,model ="normal")
setups[[7]] <-dgp(e = eF_x3,  m = 0,     z = zF, sd = 1,model ="normal")
setups[[8]] <-dgp(e = eF_x3,  m = mF_x1, z = zF, sd = 1,model ="normal")
setups[[9]] <-dgp(e = eF_x3,  m = mF_x3, z = zF, sd = 1,model ="normal")
setups[[10]]<-dgp(e = eFs_x3, m = 0,     z = zF, sd = 1,model ="normal")
setups[[11]]<-dgp(e = eFs_x3, m = mF_x1, z = zF, sd = 1,model ="normal")
setups[[12]]<-dgp(e = eFs_x3, m = mF_x3, z = zF, sd = 1,model ="normal")
setups[[13]]<-dgp(e = 0.5,    m = 0,     z = zF, sd = 1,model="weibull")
setups[[14]]<-dgp(e = 0.5,    m = mF_x1, z = zF, sd = 1,model="weibull")
setups[[15]]<-dgp(e = 0.5,    m = mF_x3, z = zF, sd = 1,model="weibull")
setups[[16]]<-dgp(e = eF_x1,  m = 0,     z = zF, sd = 1,model="weibull")
setups[[17]]<-dgp(e = eF_x1,  m = mF_x1, z = zF, sd = 1,model="weibull")
setups[[18]]<-dgp(e = eF_x1,  m = mF_x3, z = zF, sd = 1,model="weibull")
setups[[19]]<-dgp(e = eF_x3,  m = 0,     z = zF, sd = 1,model="weibull")
setups[[20]]<-dgp(e = eF_x3,  m = mF_x1, z = zF, sd = 1,model="weibull")
setups[[21]]<-dgp(e = eF_x3,  m = mF_x3, z = zF, sd = 1,model="weibull")
setups[[22]]<-dgp(e = eFs_x3, m = 0,     z = zF, sd = 1,model="weibull")
setups[[23]]<-dgp(e = eFs_x3, m = mF_x1, z = zF, sd = 1,model="weibull")
setups[[24]]<-dgp(e = eFs_x3, m = mF_x3, z = zF, sd = 1,model="weibull")

NSIM <- 50  #How many replicas per data set
args <- expand.grid(setup = 1:length(setups),
                    nsim = c(100, 250, 500, 1000),
                    dim = c(10, 50, 500),
                    repl = 1:NSIM)

set.seed(123)
learn_yxw <- lapply(1:nrow(args), function(i) {
  simulate(setups[[args$setup[i]]], nsim = args$nsim[i],
           dim = args$dim[i])
})

#Test matrix
testx <-
  matrix(runif(10000 * max(args$dim)), nrow = 10000)
colnames(testx) <- paste("X", 1:ncol(testx), sep = "")
```

A.5.2 Causal Forest

```
library("grf")

retGRFh <- retGRFd <- args

retGRFh$Class <- TRUE
retGRFd$Class <- FALSE

NumTrees<-250

fitfun <- function(d, honestyFactor = TRUE) {

  if(attributes(d)$truth$mod == "weibull"){
    d$y<-log(d$y)  #Transform skewed distribution
  }

  W05 <- max(abs(predict(d, newdata = testx)[, "efct"] - .5)) <
    .Machine$double.eps
  mtry <- floor(sqrt(ncol(d) - 2))

  #The setups where e=0.5:
  if(W05) {
    #ci.group.size must be less than 2 because with CI"s enabled,
    #the sampling fraction would have to be less than 0.5.
    cf <- causal_forest(X = as.matrix(d[, grep("^X", colnames(d))]),
                        Y = d$y, W = (0:1)[d$trt], W.hat = 0.5,
                        min.node.size = 20, sample.fraction = 0.632,
                        mtry = mtry, ci.group.size = 1,
                        num.trees = NumTrees, honesty=honestyFactor)
  }
  else{
    cf <- causal_forest(X = as.matrix(d[, grep("^X", colnames(d))]),
                        Y = d$y, W = (0:1)[d$trt],
                        min.node.size = 20, sample.fraction = 0.632,
                        mtry = mtry,ci.group.size = 1,
                        num.trees = NumTrees, honesty=honestyFactor)
  }

  that <- predict(cf, newdat = testx)$predictions
  tTruth<-predict(d, newdata = testx)[, "tfct"]
  mean((tTruth - that)^2)
}

set.seed(123)
retGRFh$MSE_t_CF <- future_sapply(learn_yxw[1:nrow(retGRFh)],
```

```
                                    fitfun, honestyFactor = TRUE)
retGRFd$MSE_t_CF <- future_sapply(learn_yxw[1:nrow(retGRFd)],
                                    fitfun, honestyFactor = FALSE)
```

A.5.3 Transformation Forest

```
library("randomForest")
library("trtf")
library("tram")
library("survival")

testxdf<-as.data.frame(testx)
testxdf$trt <- factor(c(0, 1))[2] #Set treatment indicator.

retYOUm <- retYOUq <-args
retYOUm$Class <- "maximum"
retYOUq$Class <- "quadratic"

NumTrees<-250

fitfun <- function(d, statFactor = c("maximum", "quadratic")) {

  W05 <- max(abs(predict(d, newdata = testx)[, "efct"] - .5)) <
    .Machine$double.eps
  mtry <- floor(sqrt(ncol(d) - 2))

  if(W05) { #e = 0.5:
    #Check if the conditional response is based on normal or Weibull.
    if (attributes(d)$truth$mod == "normal") {
        m <- as.mlt(Lm(y ~ trt, data = d))
    } else {
        m <- as.mlt(Survreg(y ~ trt, data = d))
    }
    tf <- traforest(m, formula = y | trt ~ ., data = d, ntree = NumTrees,
                    minbucket=20, mtry = mtry,
                    control= ctree_control(teststat = statFactor,
                                           testtype = "Univariate",
                                           mincriterion = 0,
                                           saveinfo = FALSE))

    #Transformation steps:
    cf <- predict(tf, newdata = testxdf, type = "coef")
    cf <- do.call("rbind", cf)
    cfout <- t(t(cf[,c(1, 3)] / cf[,2]) * c(-1, 1)) #Divide through Y
```

```
    colnames(cfout) <- c("m", "tau")
    that<-cfout[,"tau"]
  }

  else{ #Setups where e!=0.5
    #Calculate treatment probabilities.
    rf <- randomForest(trt ~ ., data=d[,-which(colnames(d)=="y")],
                       ntree = NumTrees)
    #Subtract  treatment probability from treatment indicator.
    d$trtA <- (0:1)[d$trt] - predict(rf, type = "prob")[,2]

    if (attributes(d)$truth$mod == "normal") {
      m <- as.mlt(Lm(y ~ trtA, data = d))
    } else {
      m <- as.mlt(Survreg(y ~ trtA, data = d))
    }

    tf <- traforest(m, formula = y | trtA ~ ., data = d, ntree = NumTrees,
                    minbucket=20, mtry = mtry,
                    control= ctree_control(teststat = statFactor,
                                           testtype = "Univariate",
                                           mincriterion = 0,
                                           saveinfo = FALSE))

    cf <- predict(tf, newdata = testxdf, type = "coef")
    cf <- do.call("rbind", cf)
    cfout <- t(t(cf[,c(1, 3)] / cf[,2]) * c(-1, 1))
    colnames(cfout) <- c("m", "tau")
    that<-cfout[,"tau"]
  }

  tTruth <- predict(d, newdata = testxdf)[, "tfct"]
  mean((tTruth - that)^2)
}


set.seed(123)
retYOUm$MSE_t_TF <- future_sapply(learn_yxw[1:nrow(retYOUm)],
                                  fitfun, statFactor = "maximum")
retYOUq$MSE_t_TF <- future_sapply(learn_yxw[1:nrow(retYOUq)],
                                  fitfun, statFactor = "quadratic")
```