CM 764 - Project: Evaluating BART and Synthetic Tree-Based Methods for the Estimation of Individual Causal Effects

Michael St. Jules

April 2017

Abstract

Bayesian Additive Regression Tree (BART) and Synthetic Random Forest methods are evaluated for the estimation of individual causal effects with nonbinary treatments, and on data with as little simulation as possible. For this purpose, data from a small randomized, cross-over, controlled trial is used. In general, of those methods evaluated, BART performed the best on the dataset.

Introduction and Background

Causal Inference

Causal inference is the field of statistics concerned with causality, with two broad goals:

(i) model selection, or estimating the parameters of distributions over random variables and which ones *cause* which variables, formalized as a directed edge from causes to their effects in a directed acyclic graph, and

(ii) the estimation of causal effects, i.e. the expected value of one random variable given that some are *set* to specific values, as a function of these values (and possibly others conditioned on).

Both are important in science, but the latter is crucial in determining which course of action to take in applications, e.g. which medical treatment to prescribe (if any at all) or what policies for a government to adopt. The "gold standard" in establishing causal effects in science is the randomized trial, an experiment, but for financial, time-constraint or ethical reasons, randomized trials may not be feasible. As such, observational studies may be preferred, and causal inference in this setting is much more challenging. This is the setting this paper is generally concerned with.

This project is formulated within the *Neyman-Rubin causal model* for counterfactual inference, generalized to the nonbinary treatment case as in (Hirano and Imbens 2004). It consists of the following random variables:

- *subjects* (or *individuals*, *units*, *contexts* or *patients*) $\mathbf{X} \in \mathcal{X}$ (not to be confused with the data matrix, notation which I also use later)
- *treatments* (or *interventions*) $\mathbf{T} \in \mathcal{T}$
- *potential outcomes* (or *counterfactuals*) $Y_{\mathbf{t}}$ for all $\mathbf{t} \in \mathcal{T}$.

Only $Y_{\mathbf{T}}$ is observed, i.e. we do not have access to the counterfactual outcomes $Y_{\mathbf{t}}$ for which $\mathbf{t} \neq \mathbf{T}$. This is *the fundamental problem of causal inference*. Our goal, then, is to estimate $Y_{\mathbf{t}}$ for these other values of $\mathbf{t}$. This is significantly more challenging with observational data, with individuals choosing their own treatments, than in the experimental setting, where subjects are randomly and independently assigned to treatment groups.

Although much work has been done in estimating *average treatment effects* (ATEs, or average causal effects, ACEs) in the binary treatment setting and some work on *average dose-response functions* (e.g. with propensity

scores (Hirano and Imbens 2004)), i.e. the quantities

$$\mathbb{E}[Y_1] - \mathbb{E}[Y_0] \text{ and } \mathbb{E}[Y_{\mathbf{t}}],$$

respectively, functions of $\mathbf{t}$ only, we are interested in the outcomes for *individuals* (e.g. individual treatment effects or ITEs, or individual causal effects, ICEs, and indidividual dose-response functions), i.e.

$$\mathbb{E}[Y_{\mathbf{t}}|\mathbf{X} = \mathbf{x}],$$

a function of both $\mathbf{x}$ and $\mathbf{t}$. Being able to estimate these accurately could, for example, lead to further personalized and automated medicine. In the binary treatment setting, the two treatments groups are referred to as *treatment*, with label 1, and *control*, with label 0.

In the *Neyman-Rubin causal model*, some assumptions are needed for these random variables. The first is *weak unconfoundedness* (Hirano and Imbens 2004): $\mathbf{x}, \mathbf{T}, Y_{\mathbf{t}}$ are said to be *weakly unconfounded* if

$$Y_{\mathbf{t}} \perp \mathbf{T} \,|\mathbf{X}\,, \quad \text{for all } \mathbf{t} \in \mathcal{T}.$$

That is, after controlling for $\mathbf{x}$ (potential confounders), the potential outcomes and the particular treatment received are independent. It is also referred to the *no hidden confounders* assumption.

Unfortunately, this assumption is not technically verifiable in practice in the observational setting, since, of course, $Y_{\mathbf{t}}$ is not observed for all $\mathbf{t} \in \mathcal{T}$. My simulation in this project, however, will satisfy it trivially, since the outcomes we sample $\mathbf{t}$ conditionally on

A second assumption is that each pair $(\mathbf{x}, \mathbf{t}) \in \mathcal{X} \times \mathcal{T}$ must be possible, i.e. has a positive probability or a positive density.

Overview

The methods I evaluated are those which seemed most promising in (Lu et al. 2017), judging by Figure 1 in that paper: Bayesian Additive Regression Trees (BART, with the function `bart` from the R package `BayesTree` (H. Chipman and McCulloch 2016)) and synCF (with the function `rfsrcSyn` from the R package `randomForestSRC` (H. Ishwaran and Kogalur 2017)). I also evaluated some slight modifications to these methods. The use of BART for causal inference is discussed in detail in (J. L. Hill 2011), which also mentions the possibility of its use for individual treatment effects. BART was one of the three overall winners of a recent causal inference competition, also performing the best on the individual prediction task. (J. Hill 2016) Other methods in (J. Hill 2016), (Shalit, Johansson, and Sontag 2016), (Alaa and Schaar 2017), (Athey and Imbens 2016), (Wager and Athey 2015) have also been used in the estimation of individual treatment effect, but only for binary treatments, and much of the code has been made specific to binary treatments. It is beyond the scope of this project to generalize them.

I present some background on BART and synthetic forests before proceeding to the experiments. Of those methods evaluated, BART seemed to have performed the best.

BART

Bayesian and Additive Regression Trees (*BART*) (H. A. Chipman et al. 2010) is a sum of trees regression model, which uses boosting with the Bayesian regularized trees from which it is composed. Its application to the estimation of causal effects is discussed first in (J. L. Hill 2011). The model is written as

$$Y = \sum_{j=1}^{m} f_j(\mathbf{x}) + \epsilon = \sum_{j=1}^{m} g(\mathbf{x}; T_j, M_j) + \epsilon\,, \qquad \epsilon \sim \mathcal{N}(0, \sigma^2)\,,$$

where each $f_j = g(-; T_j, M_j)$ is the predictor of a binary decision tree T_j and $M_j = \{\mu_{kj}|k\}$ is the set of values assigned to each terminal node, with μ_{kj} assigned to terminal node k. By default `ntree` $= m = 200$. A tree

T_j may have depth ≥ 2 to capture interactions between covariates. To regularize the model, data-informed priors are put on the parameters of the T_j, M_j and σ, and the goal is to be able to sample a model from the posterior

$$p((T_1, M_1), \ldots, (T_m, M_m), \sigma | \mathbf{X}, y) \ ,$$

to apply to new data $\mathbf{x}$. The prior is factored as

$$p((T_1, M_1), \ldots, (T_m, M_m), \sigma | \mathbf{X}, y) = \prod_{j=1}^{m} \Big[\Big(\prod_k p(\mu_{kj} | T_j) \Big) p(T_j) \Big] p(\sigma) \ .$$

Here, $p(T_j)$ is determined by

(i) the probability that a node at depth $d \geq 0$ is nonterminal, decreasing as d increases

(ii) the distribution on the splitting variable assignments at each interior node, chosen by default ('usequants = FALSE') to be uniform over the range of values taken by a variable, or uniform over the quantiles of that variable, i.e. uniform over the averages of pairs of successive values in the data ('usequants = TRUE'); and

(iii) the distribution on the splitting rule assignment in each interior node, conditional on the splitting variable, chosen by default to be uniform over the discrete set of splitting values.

The other priors are chosen as $p(\mu_{kj}|T_j) = \mathcal{N}(\mu_{kj} \ ; \ \mu_\mu, \sigma_\mu)$ and $p(\sigma)$ so that σ^2 is inverse chi-squared, to introduce conjugacy structure for ease of computation, with their hyperparameters estimated from the data.

The model is then fit by a *Bayesian backfitting Markov chain Monte Carlo algorithm*, a Gibbs sampler, by sampling (T_j, M_j) conditional on all other variables (including $\mathbf{y}$), for each j, and σ conditional on all other variables (including $\mathbf{y}$), and repeating this until convergence. In particular, sampling (T_j, M_j) can be done conditional only on the partial residual $\mathbf{R}_j = \mathbf{y} - \Big(\sum_{j' \neq j} g(\mathbf{x}_i); T_{j'}, M_{j'} \Big)_{i=1}^{N}$ and σ.

To predict the response for a new data point $\mathbf{x}$, we simply look at the prediction of one of the models near convergence (or multiple of them, and take their average).

The method I call `BART` below is simply BART with default values, in particular `usequants = FALSE`. `BART2` refers to BART with default values but quantile splitting, i.e. `usequants = TRUE`. I would have expected `BART2` to perform better, since `BART`'s uniform range splitting seems naive, and quantile splitting on the binary treatment variables should make no difference. However, this was not observed in the experiments, as `BART` generally performed better than `BART2`.

Synthetic forests

A synthetic forest (Ishwaran and Malley 2014) is a random forest precomposed with several other random forests. In particular, given a dataset $(\mathbf{X}, \mathbf{y}) = (\{\mathbf{x}_i\}_{i=1}^N, \{y_i\}_{i=1}^N)$ and a new data point to make a prediction on, $\mathbf{x}$, a synthetic forest is a random forest which takes as input training data $(\mathbf{X}, \mathbf{y})$, and new data $\mathbf{x}$, as well as the predictions, called *synthetic features*, made by several other random forests with different hyperparameters on this new data $\mathbf{x}$. In particular, the use of synthetic forests can replace hyperparameter tuning in random forests, but even more than this, it can choose predictions from the different random forests *locally* and *adaptively*. Synthetic forests are implemented by the function `rfsrcSyn` in the R package `randomForestSRC` (H. Ishwaran and Kogalur 2017).

A random forest is made up of `ntree` trees, with nodes of size at least `nodesize`≥ 1. At each node that can be split and still satisfy the `nodesize` constraint, a random subset of $1 \leq$`mtry` $\leq p$ features are chosen, where p is the total number of features, and the best split among splits using any of those `mtry` features is performed.

For synthetic forests, random forests are first constructed with `ntree` trees each, where by default `ntree`=1000, one forest for each value in a sequence of different node sizes, the argument `nodesizeSeq`, by default

`c(1:10,20,30,50,100)`. The ceiling of $p/3$ is used for the `mtry` values of the random forests, but a sequence `mtrySeq` of different values can instead be supplied (each value paired with each `nodesizeSeq` value). Node size is prioritized here since it acts as "a type of bandwidth smoothing parameter", with larger node sizes smoothing the predictions.

Then, another random forest is constructed, taking as input the dataset and the predictions of the previous random forests on the new data as input. It uses the same `ntree` value, but its own `nodesize` and `mtry` values. By default, `nodesize=5` and `mtry`=`ceiling`(p+`length(nodesizeSeq)`)/3, i.e. a third of the new number of features.

The data used in this project only has about 34 data points (depending on missing responses), so the larger `nodesizeSeq` default values will yield trivial trees. The trees themselves are constructed on bootstrap samples of the data.

The method I call `synth` here is just `rfsrcSyn` with default values. An important issue with `synth` here, however, is that it does not reweight treatment groups, so that any group which is underrepresented may be effectively ignored by never or rarely splitting on the treatment variable(s). As such, one might expect `synth` to have a relatively large average squared bias, and this does seem to be the case in my experiments (compard to `BART` and `synCF`, which I describe next).

Also using `rfsrcSyn` is the method `synCF`, which is a simple adaptation of the same method by the same name in (Lu et al. 2017) to 4 different treatments. I construct separate predictors for each treatment group, depending only on the data from the corresponding treatment group, and for prediction only on new data from the same treatment group. However, when no subject is in a particular treatment group for the given sample, I just use all of the data for that particular group.

This method may be less suitable for a large number of possible treatments, and especially continuous treatment variables, although in the latter case, treatments may potentially be grouped into a small number of intervals or blocks. In particular, we could use random or greedily chosen blocks, or perhaps equivalently, just use trees which start by splitting only on the treatment variables several times, before then splits on other variables (possibly again including the treatment variable). This is left to future work.

BART with synthetic features

I also implemented BART with synthetic features as in `synth`, i.e. the algorithm is exactly the same, with multiple random forests producing synthetic features, but instead of passing these synthetic features to another random forest, they are passed to BART. One might expect these synthetic features to overwhelm BART, with BART splitting on them at the (relative) expense of splitting on the treatment variable, and furthermore, the synthetic features themselves did not come from treatment-balanced algorithms, i.e. the treatment groups were not reweighted. Together, one should expect this algorithm to be more biased than `BART` alone, but less biased than `synth`, which is observed in the experiments. With default values (and `usequants = FALSE`), I call this method `BARTsynth`, and with default except for `usequants = TRUE`, I call this method `BARTsynth2`.

Experiments

I start with a randomized, cross-over, controlled trial (Chan et al. 2001), i.e. a study in which each subject received multiple treatments separately, with responses recorded for each treatment. Under the assumption that earlier treatments do not affect responses from later treatments, a subject's multiple responses may be treated like different potential outcomes, pretending that they only actually received one of the treatments. The only data which is simulated is the treatment assignment: each subject i received several treatments, but, according to some non-uniform probability distribution over treatments conditional on the characteristics of subjects, $\mathbf{x}_i$, we keep only the data corresponding to *one* of their treatments, say $\mathbf{t}_i^*$, so

that $\mathbf{t}_i^* \sim p(\mathbf{T}|\mathbf{x}_i) = P(\mathbf{T}|\mathbf{X} = \mathbf{x}_i)$ for the treatment assignment distribution $p(\mathbf{t}|\mathbf{x})$, which is the *generalized propensity score* (Hirano and Imbens 2004) for our simulated data. That is, we keep only $(\mathbf{x}_i, \mathbf{t}_i^*, y_i(\mathbf{t}_i^*))$, where $y_i(\mathbf{t})$ is the response of subject i to treatment $\mathbf{t}$. Hence, the training dataset is semi-simulated while the test set comes directly (or with minor modification) from the study. The goal is then, with this subset of the data, to predict each subject's responses to all of the treatments they received in the trial. As far as I am aware, this is the first use of a crossover study to evaluate methods for the estimation of individual treatment effects with access only to one potential outcome per subject (as is usually the case in observational studies). As such, this evaluation relies neither on full simulation as in (Lu et al. 2017) nor on the use of nearby surrogates for potential outcomes as in (Shalit, Johansson, and Sontag 2016). In future works, the treatment assignment distribution may come from a generalized propensity score estimated from a different study involving the same treatment and many of the same covariates. One other major difference between this project and (Lu et al. 2017) is the size of the training set: mine is about 34, while (Lu et al. 2017) use 500 and 5000.

Although initially interested in studying dose-effects with a continuous treatment variable, I've been unable to find such a crossover study, and have settled for a crossover study with each subject receiving 3 or 4 of at most 4 possible treatments was used, where the 4 possible treatments are pairs of binary variables (Chan et al. 2001). In this study, the treatments are exposure and position, (`EXP, POS`), denoting exposure (`EXP=1`) and non-exposure to pepper spray (`EXP=0`), and restrained prone (`POS=1`) and sitting (`POS=0`) positions after exposure (or non-exposure), and the responses of interest for this project are heart rate, respiratory rate, blood pressure (arterial, systolic and diastolic), tidal volume at 1 minute or 3 minutes after treatment, although various other measurements are taken in the study. The covariates of interest (besides treatment) are baseline measurements for the responses of interest, as well as demographic information, like age, sex, ethnicity, history of tobacco use, medical history, history of medication use, height, weight and body mass index. The assignemnt distribution is defined as $p(\mathbf{t}|\mathbf{x}) = p(\texttt{EXP}, \texttt{POS}|\mathbf{x}) = p(\texttt{EXP}|x)p(\texttt{POS}|\texttt{EXP},\mathbf{x})$, where $\log p(\texttt{EXP=1}|\mathbf{x})$ and $\log p(\texttt{POS=1}|\texttt{EXP},\mathbf{x})$ are linear in the variables being conditioned on and some interaction terms. The average (over all subjects) probability of `EXP=1` is around 0.11, while that of `POS=1` is around 0.26.

The measures used to evaluate each method are the *average squared bias* (which I often just call bias), the *average variance* and their sum, the *average prediction squared error*, or *APSE*. In this setting, the mean of each response is just the unique response (no $\mathbf{x}_i$ is repeated), so the variances of the responses are taken to be 0. In the causal inference literature (J. L. Hill 2011), the estimation of heterogeneous effects (PEHE) for binary treatments may be used, and this is just the MSE, mean squared error, or RMSE, root mean squared error, for the difference between treated and control responses for each subject, i.e. the average squared difference or the root of this, with the average taken over the values to be predicted.

In this project, I use the default values for all algorithms, unless otherwise specified (as described with `usequants`). Prediction is also done within sample rather than out-of-sample, i.e. all subjects are used for learning with only one treatment each, and predictions are made on the same subjects, for all treatments they received in the cross-over study, rather than separating the subjects into training and test subjects, and making predictions on test subjects, for all available treatments. Out-of-sample prediction errors would be interesting to compare methods with in future work. For this small dataset with $N = 34$ subjects, k-fold cross-validation for $k \geq 5$ (or just leave-one-out cross-validation with $k = N$) would be preferable, to not spread the data to thinly.

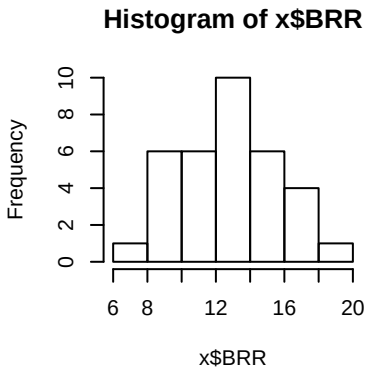
Preparing the Data and Treatment Assignment

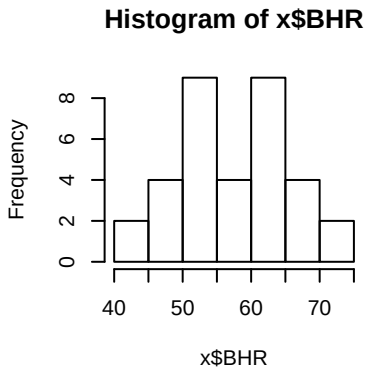
```
library(haven)
data <-
  read_por("~/Desktop/CM 764/Project/Pepper_spray/ICPSR_02961/DS0001/02961-0001-Data.por")
# fix directory so it doesn't depend on my computer

# For each subject, replace their baseline vital measurements
# (they repeat the measurements before each trial)
```

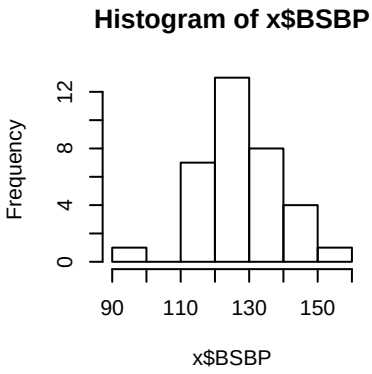
```r
# with their minimum baseline over all trials.
# This should hopefully take care of some treatment order effects

# The covariates are:
#BTV        "BASELINE TIDAL VOLUME"
#BRR        "BASELINE RESPIRATORY RATE"
#BHR        "BASELINE HEART RATE"
#BSBP       "BASELINE SYSTOLIC BLOOD PRESSURE"
#BDBP       "BASELINE DIASTOLIC BLOOD PRESSURE"
#BMAP       "BASELINE MEAN ARTERIAL PRESSURE"

# The covariates (dependent variables) will consist of the above (modified), and
#AGE        "AGE OF SUBJECT"
#SEX        "GENDER OF SUBJECT"
#ETH        "ETHNICITY OF SUBJECT"
#HT         "SUBJECT'S HEIGHT (IN METERS)"
#WT         "SUBJECT'S WEIGHT (IN KILOGRAMS)"
#BMI        "BODY MASS INDEX (KG/M2)"
#PMH        "PAST MEDICAL HISTORY"
#TOB        "TOBACCO USE HISTORY"
#MED        "HISTORY OF MEDICATION USE"

for (subj in data$SUBJ){
  data$BTV[data$SUBJ==subj] <- min(data$BTV[data$SUBJ==subj])
  data$BRR[data$SUBJ==subj] <- min(data$BRR[data$SUBJ==subj])
  data$BHR[data$SUBJ==subj] <- min(data$BHR[data$SUBJ==subj])
  data$BSBP[data$SUBJ==subj] <- min(data$BSBP[data$SUBJ==subj])
  data$BDBP[data$SUBJ==subj] <- min(data$BDBP[data$SUBJ==subj])
  data$BMAP[data$SUBJ==subj] <- min(data$BMAP[data$SUBJ==subj])
}

# View(data)

x.test <- data[c("SUBJ", "AGE", "SEX", "ETH", "HT", "WT", "BMI", "PMH", "TOB", "MED",
                 "BTV", "BRR", "BHR", "BSBP", "BDBP", "BMAP")] #baseline covariates
# but also SUBJ, for convenience, but SUBJ will be removed later
x <- unique(x.test) #get rid of duplicated rows

#test data x values to produce predicted y's
x.test <- cbind(x.test[,names(x)!="SUBJ"], EXP=data$EXP, POS=data$POS)
```

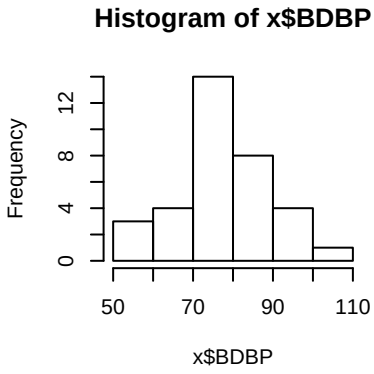
```r
# View the histograms for the baseline covariates corresponding to response covariates.
# We want to predict the response on the same power-scale as the corresponding
# baseline covariate, since rather than applying power transformations
# guided by the skew of the response, which may be the result of the biased
# treatment assignment and lead to poor generalization,
# I check for skew in the corresponding baseline covariates.
# Furthermore, many of these variables have been observed to be roughly
# normally distributed in the general population.
par(mfrow=c(2,3))
hist.default(x$BRR) #pretty well normal
hist.default(x$BHR) #symmetric but possibly two-modal
hist.default(x$BSBP) #pretty well normal
hist.default(x$BDBP) #pretty well normal
```

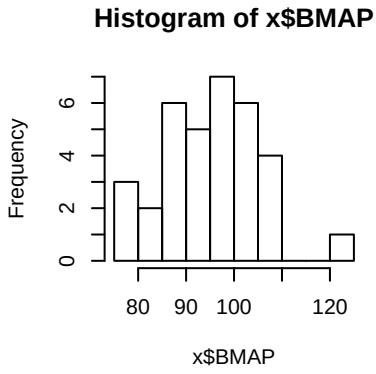
```r
hist.default(x$BMAP) #pretty well normal
hist.default(x$BTV) #slightly right-skewed (right-tailed)
```

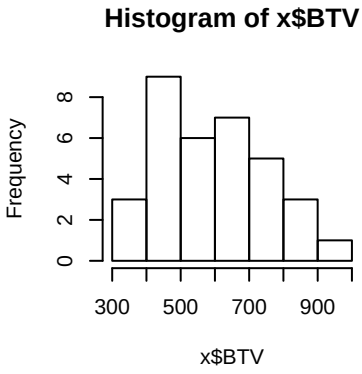




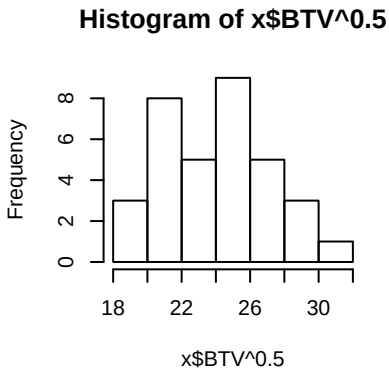









```r
# a power transform of ~0.5 would fix this
hist.default(x$BTV^0.5)
```



Now, define the simulated treatment assignment mechanism and how to sample from the data:

```r
# First, some global variables to avoid recomputing
max.AGE <- max(x$AGE)
min.AGE <- min(x$AGE)
max.WT <- max(x$WT)
min.WT <- min(x$WT)
#the log probability of EXP=1 will be linear (affine) in the following
exponent <- 3*(max.AGE-x$AGE)/(max.AGE-min.AGE) + (x$WT-min.WT)/(max.WT-min.WT) +
  5*(x$SEX==1) + 3*(x$ETH==2) + 2*(x$ETH==3) + 5*(x$TOB == 2) +
  3*((max.AGE-x$AGE)/(max.AGE-min.AGE)+1)*(3*(x$ETH==2)+2*(x$ETH==3))*
```

```
  (3*(x$SEX==1)+1)*(x$TOB == 2)
max.exponent <- max(exponent)
min.exponent <- min(exponent)
#i.e. log p(EXP=1) = a*exponent+b
#want max prob of EXP==1 to be 1/5, min to be 1/18, so fit a line:
#slope
a.EXP = (log(1/5)-log(1/18))/(max.exponent-min.exponent)
#intercept
b.EXP = log(1/5) - a.EXP*max.exponent

#log p(POS=1|EXP) = a*(exponent+2*EXP)+b
#want max prob to be 3/4, min to be 1/10
#slope
a.POS_EXP = (log(3/4)-log(1/10))/(max.exponent+2-min.exponent) #2 for 2*EXP
#intercept
b.POS_EXP = log(3/4) - a.POS_EXP*(max.exponent+2) #2 for 2*EXP

pEXP1 <- function(){
  exp(a.EXP*exponent+b.EXP)
}

pPOS1_EXP <- function(EXP){
  exp(a.POS_EXP*(exponent+EXP)+b.POS_EXP) #this was supposed to be
  #exponent+2*EXP, but it's too late to fix now
  #the distribution below is still a valid distribution
}

#sample treatments for each subject
treatment_dist <- function(x){
  x.EXP <- as.integer(runif(nrow(x)) <= pEXP1())
  x.POS <- as.integer(runif(nrow(x)) <= pPOS1_EXP(x.EXP))
  data.frame(EXP=x.EXP,POS=x.POS)
  # At least three possibilities for dealing with missing treatments in data:
  # (1) not care that some responses will be missing for some treatments (OK for trees?)
  # (2) keep reassigning until a valid treatment is obtained
  # (3) "round" to the nearest treatment:
  # If (0,0) or (1,1) is obtained but missing, flip a coin between (1,0) and (0,1),
  #  favouring (0,1) (e.g. 2/3) ?
  # If (0,1) or (1,0) is obtained but missing, flip to the other

  # For now, I'm using (1)
}


getSample <- function(y_name="RR_1", x.=x, data.=data,
                      t=NA, treatment_dist.=treatment_dist){
  if(is.na(t)){
     t <- treatment_dist.(x.)
  }
  y <- numeric(nrow(x.))
  y[] <- NA #fill with NAs
  j <- 1 #index in x
  #note that rows appear in the same order (increasing by SUBJ) in both x and data
```

```
  for(i in 1:nrow(data.)){
    if(data.[i,"SUBJ"]==x.[j,"SUBJ"] & all(data.[i, c("EXP","POS")] == t[j,c("EXP","POS")]))
      y[j] <- as.double(data.[i,y_name])
    if(i < nrow(data.) & data.[i+1, "SUBJ"] != x.[j,"SUBJ"]){
      j <- j+1
    }
  }

  cbind(x.[,names(x.) != "SUBJ"],t,y) #remove SUBJ
  #should x and t be combined into one variable?
}
```

At the very least, it's clear that the treatment assignment is not uniform, with some treatment groups much larger than others on average:

```
# marginal probability of EXP=1 (i.e. being pepper sprayed)
mean(sapply(1:5000, FUN=function(j){mean(treatment_dist(x)$EXP)}))
```

```
## [1] 0.1117
```

```
# marginal probability of POS=1 (i.e. being restrained)
mean(sapply(1:5000, FUN=function(j){mean(treatment_dist(x)$POS)}))
```

```
## [1] 0.2637647
```

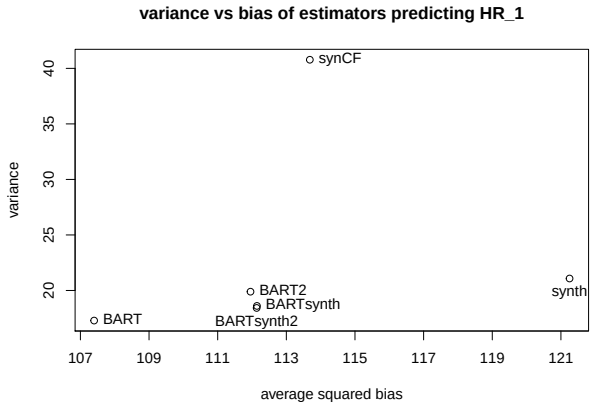
Results

We consider two partial orders on estimators here:

- we say estimator $\tilde{A}$ is *weakly preferred* to estimator $\tilde{B}$ if the average prediction squared error (APSE) of $\tilde{A}$ is strictly less than that of $\tilde{B}$.
- We say estimator $\tilde{A}$ is *clearly preferred* to estimator $\tilde{B}$ if both the average squared bias of $\tilde{A}$ is at most that of $\tilde{B}$, and the average variance of $\tilde{A}$ is at most that of $\tilde{B}$; and one of these inequalities is strict (or $\tilde{A}$ is weakly preferred to $\tilde{B}$)

In the following a *clear winner* is an estimator with both the least average squared bias and the least average variance, or one for which no other estimator is clearly preferred to it. A clear winner may not always exist. A *weak winner* is an estimator with the least APSE, or one for which no other estimator is weakly preferred to it.

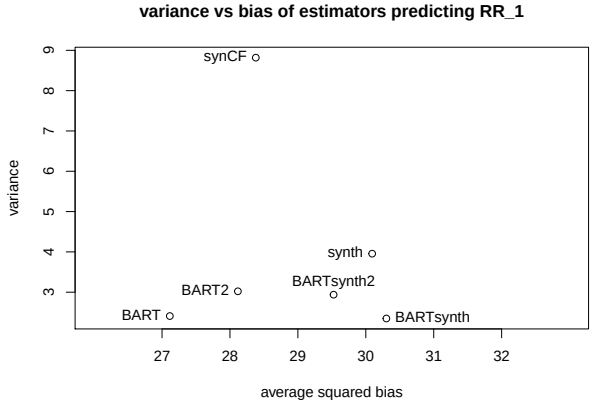
The following graph and estimates were produced with 100 samples:



```
##                bias2 variance     APSE
## BART       107.3996 17.28949 124.6891
## BART2      111.9560 19.89106 131.8471
## synth      121.2518 21.07799 142.3298
## synCF      113.6847 40.77653 154.4613
## BARTsynth  112.1439 18.60873 130.7526
## BARTsynth2 112.1353 18.42724 130.5625
```

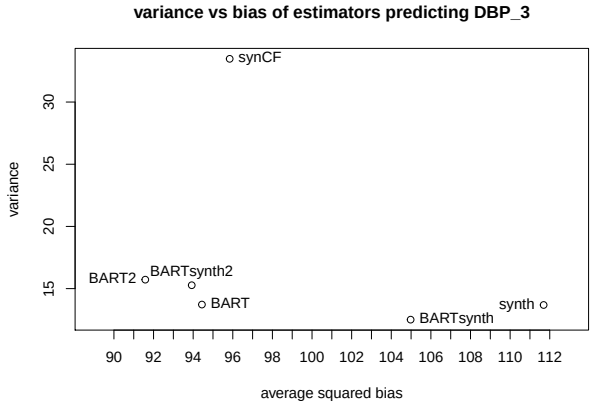
BART is the clear winner here. BART2, BARTsynth and BARTsynth2 performed significantly worse than BART, and synth and synCF performed significantly worse still.

The following graphs and estimates were produced with 50 samples each:




```
##                bias2 variance     APSE
## BART       27.11470 2.407885 29.52258
## BART2      28.11672 3.022351 31.13907
## synth      30.09379 3.955569 34.04936
## synCF      28.38078 8.818271 37.19905
## BARTsynth  30.30310 2.347645 32.65075
## BARTsynth2 29.52576 2.937958 32.46371
```

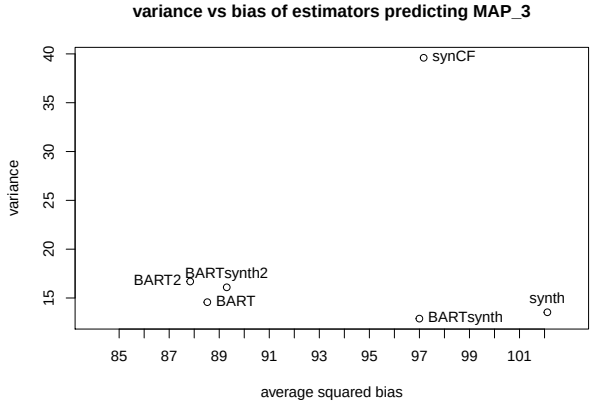
BART has the least average squared bias here by a fairly large margin, but a slightly higher variance than BARTsynth, so there is no clear winner. BART is a weak winner, and by a decent margin. synCF had a significantly higher variance than the others.




```
##                bias2 variance     APSE
## BART        94.44173 13.71698 108.1587
```

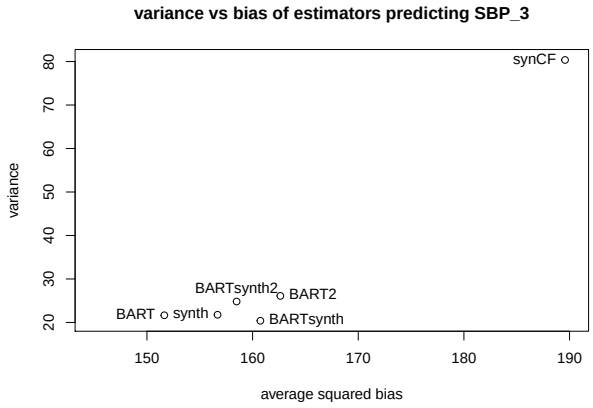
```
## BART2       91.58287 15.71932 107.3022
## synth      111.69140 13.68408 125.3755
## synCF       95.84338 33.47449 129.3179
## BARTsynth  104.97712 12.50366 117.4808
## BARTsynth2  93.92743 15.27268 109.2001
```

The are no clear winners here, but as BART, BART2 and BARTsynth2 are all close in average squared bias and average variance. Between them, APSE(BART2) < APSE(BART) < APSE(BARTsynth2), so BART2 is the weak winner.




```
##                bias2 variance     APSE
## BART        88.52395 14.57361 103.0976
## BART2       87.83971 16.69055 104.5303
## synth      102.11338 13.53934 115.6527
## synCF       97.16986 39.60143 136.7713
## BARTsynth   96.99828 12.88733 109.8856
## BARTsynth2  89.30043 16.09516 105.3956
```

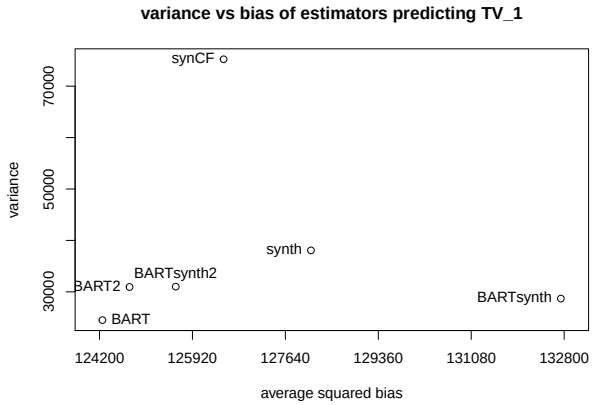
There is no clear winner here, but BART, BART2 and BARTsynth have no estimators clearly preferred to them. BARTsynth has the least variance, but there's a large trade-off, giving it a much larger bias. BART is the weak winner, followed closely by BART2.




```
##               bias2 variance     APSE
## BART       151.6443 21.65188 173.2961
## BART2      162.6264 26.10487 188.7313
## synth      156.6809 21.76141 178.4423
## synCF      189.5718 80.36604 269.9379
```

```
## BARTsynth  160.7367 20.38009 181.1168
## BARTsynth2 158.4871 24.81251 183.2997
```

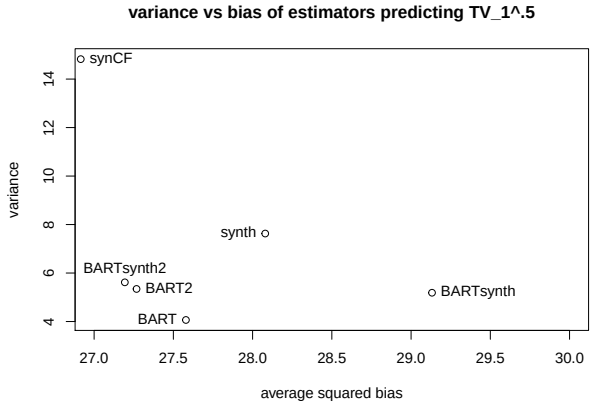
There is no clear winner here, but BART is almost one and has the least APSE by a decent margin. BART is the weak winner.




```
##                bias2 variance     APSE
## BART       124253.8 24501.16 148755.0
## BART2      124756.3 30942.41 155698.7
## synth      128113.9 38061.93 166175.8
## synCF      126496.7 75231.13 201727.9
## BARTsynth  132738.7 28681.35 161420.1
## BARTsynth2 125611.2 31009.21 156620.4
```

BART here is the clear winner. BART2 is clearly preferred to BARTsynth2, although the difference in variance may be smaller than the difference in sampling, and neither is clearly comparable to BARTsynth, but both are weakly preferred to it (lower APSE). synth and synCF have the worst APSE, and synCF is particularly bad because of its high variance.

In the last experiment, BTV is replaced with BTV^0.5; and TV_1, with TV_1^0.5:




```
##                bias2  variance     APSE
## BART       27.57933  4.063969 31.64330
## BART2      27.26774  5.343592 32.61133
## synth      28.07969  7.629227 35.70892
## synCF      26.91572 14.822367 41.73809
## BARTsynth  29.13208  5.187194 34.31927
```

```
## BARTsynth2 27.19383  5.616510 32.81034
```

synCF here is actually the least biased, but it has the worst variance, so it is not clearly comparable to any of the others. However, it does have the worst APSE, so all others are weakly preferred to it. BART, BART2, BARTsynth2 and synCF are not clearly comparable to one another, but in terms of APSE, from lowest/best to highest/worst, we have: BART, BART2, BARTsynth2, BARTsynth, synth, synCF.

Note that in all of the above experiments the average squared bias was always much greater (over 5 times greater in most cases) than the average variance. I suspect the main reason for this is because I defined the expected value of the response for individuals with covariates $\mathbf{x}_i$ to be y_i, i.e. the unique response corresponding to those particular covariates, and in doing so, the average squared bias absorbed the average variance of the response y which would have been observed under repitition of the experiment with the same subjects, setting it to 0 here. That is, the bias term is computed from differences between the average predicted value and a *noisy* response. Rather than using the responses directly, their values could have been smoothed locally.

Another possible reason is how few subjects are actually in the study: about 34, depending on whether some have missing responses or not.

All together, BART was a weak winner in all but one case (`BP_3`), in which it followed closely in second. BART was also a clear winner in two casse (`HR_1, TV_1`), and *nearly* a clear winner in two other cases (`RR_1, SBP_3`). synCF almost always performed poorly due to its high variance, and synth only performed well on `SBP_3`. synth was usually more biased than BART, BART2 and synCF, as might be expected. BART2 and BARTsynth2 performed well in comparison, and BARTsynth only slightly worse than BART2 and BARTsynth2. As such, for the particular dataset in this project, I would order the estimators from best to worst the following way:

1. BART
2. BART2 and BARTsynth2
3. BARTsynth
4. synth
5. synCF

In terms of computation time, the BART-based methods were a couple of times slower.

Conclusion

In this project, I compared the two main contenders, BART and synCF (and some variants), in (Lu et al. 2017) in for the estimation of individual causal effects on a small minimally-simulated dataset. My evaluation lead to the opposite ranking between them: in (Lu et al. 2017), synCF performed slightly better than BART, but here, BART performed better than synCF, and sometimes by quite large margins.

Directions for future work include:

- out-of-sample evaluations,
- continuous treatment variables, and
- the adaptation of other methods currently implemented only for binary treatments to nonbinary treatments.

References

Alaa, Ahmed M, and Mihaela van der Schaar. 2017. "Bayesian Inference of Individualized Treatment Effects Using Multi-Task Gaussian Processes." *ArXiv Preprint ArXiv:1704.02801*.

Athey, Susan, and Guido Imbens. 2016. "Recursive Partitioning for Heterogeneous Causal Effects." *Proceedings of the National Academy of Sciences* 113 (27). National Acad Sciences: 7353–60.

Chan, Theodore C., Gary M. Vilke, Jack Clausen, Richard Clark, Paul Schmidt, Thomas Snowden, and Tom Neuman. 2001. "Impact of Oleoresin Capsicum Spray on Respiratory Function in Human Subjects in the Sitting and Prone Maximal Restraint Positions in San Diego County, 1998." *Inter-University Consortium for Political and Social Research*. doi:10.3886/icpsr02961.v1.

Chipman, Hugh A, Edward I George, Robert E McCulloch, and others. 2010. "BART: Bayesian Additive Regression Trees." *The Annals of Applied Statistics* 4 (1). Institute of Mathematical Statistics: 266–98.

Chipman, Hugh, and Robert McCulloch. 2016. *BayesTree: Bayesian Additive Regression Trees*. https://CRAN.R-project.org/package=BayesTree.

Hill, Jennifer. 2016. "2016 Atlantic Causal Inference Conference Competition: Is Your Satt Where It's at?" *2016 Atlantic Causal Inference Conference*. http://jenniferhill7.wixsite.com/acic-2016/competition.

Hill, Jennifer L. 2011. "Bayesian Nonparametric Modeling for Causal Inference." *Journal of Computational and Graphical Statistics* 20 (1). Taylor & Francis: 217–40.

Hirano, Keisuke, and Guido W Imbens. 2004. "The Propensity Score with Continuous Treatments." *Applied Bayesian Modeling and Causal Inference from Incomplete-Data Perspectives* 226164. Chichester: Wiley & Sons: 73–84.

Ishwaran, H., and U.B. Kogalur. 2017. *Random Forests for Survival, Regression and Classification (Rf-Src)*. https://CRAN.R-project.org/package=randomForestSRC.

Ishwaran, Hemant, and James D Malley. 2014. "Synthetic Learning Machines." *BioData Mining* 7 (1). BioMed Central: 28.

Lu, Min, Saad Sadiq, Daniel J Feaster, and Hemant Ishwaran. 2017. "Estimating Individual Treatment Effect in Observational Data Using Random Forest Methods." *ArXiv Preprint ArXiv:1701.05306*.

Shalit, Uri, Fredrik Johansson, and David Sontag. 2016. "Estimating Individual Treatment Effect: Generalization Bounds and Algorithms." *ArXiv Preprint ArXiv:1606.03976*.

Wager, Stefan, and Susan Athey. 2015. "Estimation and Inference of Heterogeneous Treatment Effects Using Random Forests." *ArXiv Preprint ArXiv:1510.04342*.