Department of Mathematics

TUM School of Computation, Information and Technology

Technical University of Munich

Applying double machine learning and BART methods to the American Causal Inference Conference 2022 Data Challenge

Ruixuan Zhu

Thesis for the attainment of the academic degree

Master of Science

at the TUM School of Computation, Information and Technology of the Technical University of Munich

Supervisor:

Prof. Mathias Drton

Advisor:

M.Sc. Jun Wu

Submitted:

Munich, 31. March 2023

I hereby declare that this thesis is entirely the result of my own work except where otherwise indicated. I have only used the resources given in the list of references.

Ruixuan Zhu

Munich, 31. March 2023 Ruixuan Zhu

Abstract

During the master thesis, Bayesian Additive Regression Tree (BART), Bayesian Causal Forest(BCF), and Double Machine Learning(DML) are applied to solve American Causal Inference Conference 2022 Data Challenge. Bayesian Causal Forest(BCF) is a variant of the Bayesian Additive regression tree (BART) model. The R language is used for all implementations. For evaluation of the performances of these three models, Root Mean Squared Error(RMSE), uncertainty interval coverage, uncertainty interval width, and absolute bias are employed as metrics. Root Mean Squared Error(RMSE) and uncertainty interval coverage are emphasized among the four metrics since they are highlighted by the Data Challenge host. The evaluations show that the three models all have a good performance regarding Root Mean Square Error(RMSE) and the two BART-based models have much better performances than Double Machine Learning(DML) in terms of uncertainty interval coverage. Within BART-based models, Bayesian Causal Forest(BCF) outperformed Bayesian Additive Regression Tree(BART). Moreover, the two BART-based models outperformed Double Machine Learning(DML) significantly concerning the subgroup estimands, which is crucial for dealing with treatment effect heterogeneity.

Contents

List of Figures

List of Tables

List of Algorithms

1 Introduction

American Causal Inference Conference 2022 Data Challenge was held by the Society for Causal Inference. It is an annual causal inference competition, and a team at Mathematica led by Mariel Finucane and Dan Thal was running the most recent competition. The Mathematica team has generated thousands of datasets resembling policy assessments and integrated hidden causal relationships. Participants will compete to identify the best state-of-the-art methods for measuring these impacts and assessing which social policies are making a difference to the individuals and communities they serve. The annual Data Challenge provides an opportunity to compare causal inference methods across different data generation processes (DGPs) and they will propose a challenging problem each year. The emphasis of the 2022 Data Challenge was on the time-varying property of the given datasets.

Bayesian addition tree (BART) and double machine learning are the two dominant baseline models among all submissions to the American Causal Inference Conference 2022 Data Challenge. We use three causal inference models in the thesis: Bayesian Addition Regression Tree (BART), Bayesian Causal Forest (BCF), and Double Machine Learning. Based on the root mean squared error (RMSE) results, they all performed well in providing us with estimates of sample average treatment effects on the treated (SATT). Furthermore, the Bayesian causal forest outperformed the other two models. Considering the submission results of Mathematica's American Causal Inference Conference 2022 Data Challenge, my implementation results are still among the top-performing methods.

2 Problem Setting and Related Work

Causal inference is about counterfactual predictions. The causal inference model predicts what would happen to the same unit if faced with a counterfactual situation. Most causal inference statisticians define a causal effect as a comparison of what happens in two or more different states. One is fact and the other is counterfactual.

Definition 1 ⟨unit [21]⟩

A unit is the atomic research object in the treatment effect study [21].

Definition 2 ⟨Treatment [21]⟩

Treatment refers to the action that applies to a unit [21].

Definition 3 ⟨Potential Outcome⟩

We define the causal effect of treatment via potential outcomes. For a binary treatment $Z \in \{0, 1\}$, we define potential outcomes $Y_i(1)$ and $Y_i(0)$ corresponding to the the ith unit [19].

1. $Y_i(1) = Y(Z = 1)$ *is the outcome if the ith unit is under treatment.*
2. $Y_i(0) = Y(Z = 0)$ *is the outcome if the ith unit is under control.*

The fundamental problem with causal inference is that we can only observe one of the potential outcomes for each unit, and the other outcome is counterfactual. Therefore, only one of $Y_i(1)$ and $Y_i(0)$ can be observed at a time [19]. We can build causal inference models to predict the unobserved potential outcomes $Y_i(1)$ or $Y_i(0)$ and infer the causal effects.

2.1 Data

Since 2016, the American Causal Inference Conference (ACIC) has hosted a data challenge in which teams compete to estimate causal impacts on simulated datasets based on real-world data from fields such as health care or education [20]. The competition provides a ground for state-of-art causal inference methods that have the potential to revolutionize program evaluation [20]. The ACIC 2022 Data Challenge is designed to help understand which methods provide the most accurate and sophisticated estimates of policy effectiveness. The challenge's organizing committee designed the datasets to reflect data from evaluations of large-scale U.S. healthcare system interventions aimed at reducing Medicare expenditures [20]. The outcome of interest in the challenge is Medicare spending.

The ACIC 2022 Data Challenge consists of 200 realizations of 17 data generating processes (DGPs), with each realization producing a new sample of practices. Mathematica conceals the data generation processes (DGPs), and we have no idea which datasets share a common DGP. It is frequently unclear whether there is measured (or unmeasured) confounding in a real-world observational study.

We have 3400 datasets in total, with each dataset containing 500 practices. Practices decide whether or not to participate in the intervention. This means that in a treated practice, all patients are treated, while in an untreated practice, all patients are untreated. The data has a longitudinal structure, with patients being observed annually over time. Despite the fact that patients enter and exit the sample throughout the four-year observation period, all primary care practices are observed for the entire four-year period. The first two years are a baseline period during which no intervention is provided. The intervention is then initiated in treated practices at the start of Year 3. They will continue to receive the intervention until the end of Year 4.

The variables that are shared by 3400 datasets are described below:

id.practice: Practice identifier; Range from 1 to 500 in each dataset

Z: Treatment variable; Indicator for whether practice is in the treatment group (Z = 1) or control group (Z = 0)

year: Observation year; range from 1 to 4

post: Indicator for whether the intervention has begun for treated practices. post = 1 in Years 3 and 4, post = 0 in Years 1 and 2.

Y: Outcome variable (monthly Medicare expenditures for patients, in a year).

X1, X2, X3, X4, X5: Unordered categorical and binary practice-level covariates used to define subgroup SATT estimands.

X6, X7, X8, X9 : Additional practice-level covariates.

V1, V2, V3, V4, V5: Continuous, unordered categorical, and binary patient-level covariates

n_patients: the number of patients in each practice at a certain year

Each of the 3400 datasets has an id.practice value ranging from 1 to 500. Id.practice value i corresponds to a year between 1 and 4 and treatment status Z_i. We have the outcome Y_t and the number of patients $n\_patients_{t,i}$ at year t. Table 2.1 depicts these relationships.

$id.practice$	Z	$year$	Y	$n\_patients$	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
i	Z_i	1	Y_1	$n\_patients_{1,i}$	$\cdots$
		2	Y_2	$n\_patients_{2,i}$	$\cdots$
		3	Y_3	$n\_patients_{3,i}$	$\cdots$
		4	Y_4	$n\_patients_{4,i}$	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Table 2.1 Table to show the data block of id.practice i in one of the 3400 datasets.

2.2 Assumptions

Assumption 1 ⟨Stable Unit Treatment Value Assumption (SUVTA) [21]⟩

The potential outcomes for any unit do not vary with the treatment assigned to other units [21]. Also, there are no different forms or versions of each level of treatment with different potential outcomes for different entities [21].

Assumption 2 ⟨Ignorability⟩

The data generating processes are free of unmeasured confounding. In the longitudinal context, this means there are no unobserved covariates that relate to both treatment assignment and to the change in the untreated potential outcome $Y_{it}(0)$ from the period before the intervention (Years 1 and 2) to *the intervention period (Years 3 and 4). However, that ignorability does not preclude confounding by trends in the observed outcomes during the period before the intervention, $Y_{i2} - Y_{i1}$.*

$$\begin{aligned} z_i \perp (Y_{t,i}(1), Y_{t,i}(0)) \mid x_i, &\quad \textit{for } t \in \{3,4\}. \\ z_i \perp Y_{t_1,i}(0) - Y_{t_0,i}(0) \mid x_i, &\quad \textit{for } t_1 \in \{3,4\},\ t_0 \in \{1,2\}. \end{aligned} \tag{2.1}$$

Assumption 3 ⟨Positivity/Overlap⟩

For all i such that $z_i = 1$, we assume overlap for the treatment group:

$$0 < \mathbb{P}(z_i = 1 \mid x_i) < 1.$$

2.3 Estimands

The data challenge's targeted estimands are sample average treatment effects on the treated (SATTs). In total, we need to calculate 15 SATT statistics for each dataset: one overall SATT, two year-specific SATTs, and 12 conditional SATTs defined by the 2+3+2+3+2 levels of the categorical variables $X1$,$X2$, …, $X5$. To clearly define SATTs, we will first go over the definitions of causal effects ATE and ATT. The causal effect is the comparison between the potential outcomes under treatment and under control for the same unit or a common set of units [21].

Definition 4 ⟨Average Treatment Effect (ATE) [19]⟩

Individual treatment effects are calculated from the difference between the outcome with treatment and the outcome without treatment. The Average Treatment Effect(ATE) is the expected treatment effect across every unit in the population. We calculate the ATE by taking the average of all individual treatment effects.

$$ATE = \mathbb{E}[Y(Z=1) - Y(Z=0)]. \tag{2.2}$$

Definition 5 ⟨Average Treatment effect on the Treated group (ATT) [19]⟩

The Average Treatment Effect on the treated group(ATT) is the expected treatment effect across every unit in the population which is exposed to treatment. We calculate the ATT by taking the average of all individual treatment effects which are under treatment.

$$ATT = \mathbb{E}[Y(Z=1) \mid Z=1] - \mathbb{E}[Y(Z=0) \mid Z=1]. \tag{2.3}$$

Corollary 1

Under Assumptions 1- 3, we have:

$$\mathbb{E}[Y_i(1) - Y_i(0) \mid X_i = x] = \mathbb{E}[Y_i \mid X_i = x, Z_i = 1] - \mathbb{E}[Y_i \mid X_i = x, Z_i = 0]. \tag{2.4}$$

Proof. Under Assumption 1 and 2,

$$\begin{aligned}\text{LHS of Equation 2.4} &= \mathbb{E}[Y_i(1) \mid X_i = x] - \mathbb{E}[Y_i(0) \mid X_i = x] \\ &= \mathbb{E}[Y_i(1) \mid X_i = x, Z_i = 1] - \mathbb{E}[Y_i(0) \mid X_i = x, Z_i = 0] \\ &= \mathbb{E}[Y_i \mid X_i = x, Z_i = 1] - \mathbb{E}[Y_i \mid X_i = x, Z_i = 0] \\ &= \text{RHS of Equation 2.4}.\end{aligned} \tag{2.5}$$

□

Corollary 1 is very important. It gives us the foundation for calculating SATTs in the empirical setting. In each dataset, for each practice, we only have a unique Z_i value, so we do not know the value of LHS of Equation 2.4. However, we could compute the RHS via modeling

In order to calculate SATTs, we also need to take the variable n.patients into consideration. It serves as a weighting variable to define SATTs. With the notation of N_t:

$$N_t = \sum_{i:Z_i=1} n\_patients_{t,i}, \tag{2.6}$$

where $n\_patients_{t,i}$ denotes the number of patients in the ith practice at year t, we have the following definitions.

Definition 6 ⟨The overall SATT⟩

The overall SATT corresponding to all observations on the treated group across year3 and year4 is given as follows:

$$SATT_{overall} = \frac{1}{\sum_{t=3}^{4} N_t} \sum_{t=3}^{4} \sum_{i:Z_i=1} n\_patients_{t,i}(Y_{t,i}(1) - Y_{t,i}(0)). \tag{2.7}$$

Definition 7 ⟨SATT by year⟩

The yearly SATT corresponding to all observations on the treated group across the year3 or year4 is given as follows:

$$SATT_{yearly} = \frac{1}{N_t} \sum_{i:Z_i=1} n\_patients_{t,i}(Y_{t,i}(1) - Y_{t,i}(0)). \tag{2.8}$$

Definition 8 ⟨SATT for subgroup S⟩

The subgroup SATT corresponding to all observations on the treated group and subgroup S across year 3 and year 4 is given as follows. For example, the subgroup S could be $\{i \in [1, 500] \mid X_2 = A\}$.

$$SATT_{subgroup} = \frac{1}{\sum_{t=3}^{4} N_t(S)} \sum_{t=3}^{4} \sum_{i:Z_i=1, i \in S} n\_patients_{t,i}(Y_{t,i}(1) - Y_{t,i}(0)). \tag{2.9}$$

Here, $N_t(S) = \sum_{i:z_i=1, i \in S} n\_patients_{t,i}$

In addition to SATTs, we must calculate the corresponding 90% uncertainty intervals in the 2022 ACIC Data Challenge. Because the Bayesian Additive Regression Tree (BART) and Bayesian Causal Forest (BCF) models are based on Bayesian statistics, the SATTs calculated from these two models are based on the posterior distribution. The uncertainty intervals are then known as credible intervals. The following is the definition:

Definition 9 ⟨Credible interval⟩

Let θ *be a random variable, then a credible interval of size* $1 - \alpha$ *is an interval* (a, b) *such that:*

$$\mathbb{P}(a \leq \theta \leq b \mid x) = 1 - \alpha. \tag{2.10}$$

However, the double machine learning model is not based on Bayesian statistics and the corresponding derived uncertainty intervals are referred to as confidence intervals.

2.4 Evaluation Metrics

To evaluate the implementation results, we primarily use root mean squared error (RMSE) and uncertainty interval coverage. Mathematica's presentation at ACIC 2022 also emphasizes these two metrics. In addition, we investigate bias and uncertainty interval width in Section 6.1.

Definition 10 ⟨Root Mean Square Error (RMSE) [19]⟩

Root Mean Square Error(RMSE) is a standard way to measure a model's error in predicting quantitative data. It is the standard deviation of the prediction errors.

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}. \tag{2.11}$$

Here, n: number of observations; $\hat{y}_i$: predicted value, y_i: observed value

Definition 11 ⟨Uncertainty interval coverage rate⟩

Assume $y_1, y_2, \ldots, y_n$ are the ground-truth values of a statistic, and $[a_1, b_1], \ldots, [a_n, b_n]$ are their corresponding uncertainty interval calculated through modelling, then the uncertainty interval coverage rate is:

$$ci\_coverage = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}\{y_i \in [a_i, b_i]\}. \tag{2.12}$$

2.5 Propensity Score

Definition 12 ⟨propensity score [19]⟩

The propensity score is the conditional probability that a unit could be assigned to the treated group based on the observed covariates [19].

$$e(x) := \mathbb{P}(Z_i = 1 \mid X_i = x). \tag{2.13}$$

Nowacki et al. summarized a conclusion in a published article that adding propensity scores to pure prediction models does not improve predictive performance [16]. However, as suggested by Hahn et al., it is a common practice to include the propensity score as a covariate in a causal inference model [8].

Definition 13 ⟨propensity score as an additional covariate⟩

The mathematical model to predict outcomes with propensity score $e(x)$ is given as follows:

$$Y_i = g(X_i, Z_i, e(X_i)) + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, \sigma^2) \; i.i.d. \tag{2.14}$$

In Equation 2.14, a Bayesian non-parametric prior of $g(\cdot)$ provides modelling flexibility, while the propensity score covariate provides the anchor for robustness [21].

In the implementation of the BART model and double machine learning for the 2022 ACIC Data Challenge, we add propensity score $e(x)$ as an additional covariate. By definition, the propensity score $e(x)$ is part of the Bayesian Causal Forest (BCF) model.

As a result, it is necessary to consider a baseline model to estimate propensity score $e(x)$ for all observations (X, y). Since the treatment variable in the problem setting of the 2022 ACIC Data Challenge is a binary variable, it is possible to implement any binary classification model to estimate propensity scores $e(x)$, such as the Logit model and the Prohit model. In our implementations, we use the BART Classification model to estimate propensity score $e(x)$ to get more precise predictions.

3 Bayesian Additive Regression Tree (BART)

A decision tree is a flowchart-like structure that consists of a root node, branches, branch nodes, and leaf nodes. Predictor space is a p-dimensional space comprising all possible values of the p covariates $\{x_1, x_2, \ldots, x_p\}$ that describe the observations we have. A decision tree divides the predictor space into multiple distinct and non-overlapping regions $\{R_1, R_2, \ldots, R_n\}$ and a new observation x will be assigned to one of the regions $\{R_1, R_2, \ldots, R_n\}$ based on its corresponding values of the p covariates. A decision tree could also be mathematically expressed as a function $g(\cdot)$ that defines binary split rules $\{x_k \in A\}$ vs $\{x_k \notin A\}$ which induce partition over predictor space. When evaluating the value of a decision tree T, give the input value x, one would take x and move down the tree T until reaching one of the leaf nodes. BART is a Bayesian method using sums of regression trees. Regression trees are decision trees where the target variables can take continuous values instead of the class labels in the leaf nodes.

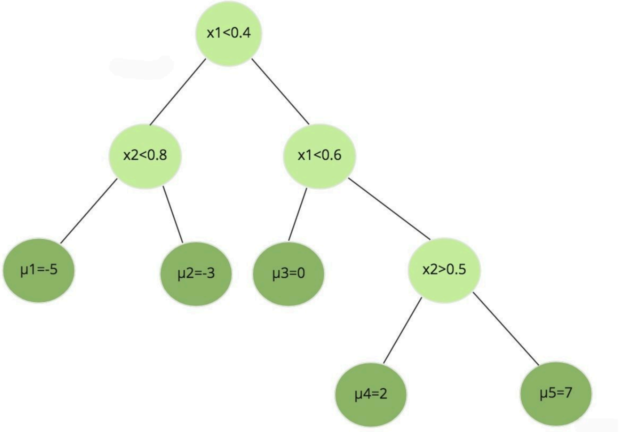

Figure 3.1 An example of decision tree

In Figure 3.1, assuming that there are p predictors, the decision tree divides the covariate space into five distinct regions $\{R_1, R_2, R_3, R_4, R_5\}$, where $R_1 = \{x = (x_1, x_2, x_3, ..., x_p) \mid x_1 < 0.4, x_2 < 0.8\}$ and so on. Therefore it could be expressed as a function $g(x) = \sum_{l=1}^{5} \mu_l \mathbf{1}(x \in R_l)$, where μ_l is the mean of all outcome values y_i which are assigned at leaf node l.

The partition of covariate space which corresponds to the decision tree in Figure 3.1 is illustrated in Figure 3.2.

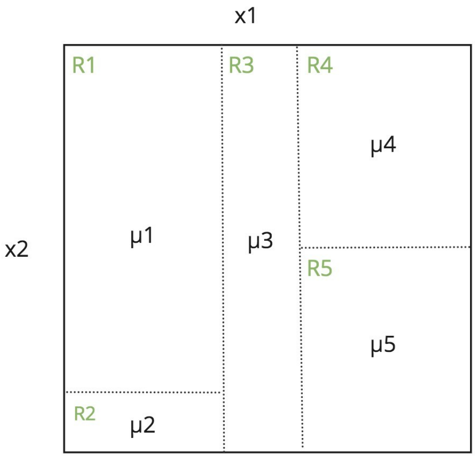

Figure 3.2 Illustration of partition of covariate space

Bayesian Additive regression tree (BART) is a Bayesian non-parametric model for causal inference which is first introduced by Chipman et al. [4]. It is a sum-of-trees model for approximating an unknown function $f(\cdot)$. Each tree in the model acts as a weak learner and explains only part of the result due to regularization. Considering the common regression framework:

$$y = f(x) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2), \tag{3.1}$$

we could define the BART model as below:

$$y = \sum_{j=1}^{m} g_j(x; (T_j, \mu_j)) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2), \tag{3.2}$$

where:

m: the total number of regression trees,

T_j: the jth binary regression tree,

μ_j: represents leaf nodes of the jth binary regression tree, assuming that the number of leaf nodes is L_j, then the vector of leaf node means $\mu_j = (\mu_{j1}, \ldots, \mu_{jL_j})$,

g_j: function mapping each leaf node to the set of predictors $x = \{x_1, \ldots, x_p\}$.

The pair (T_j, μ_j) defines the structure of the jth tree. We can deduce from the definitions above that $f(x) = \mathbb{E}[y \mid x]$ is the sum of all the leaf nodes assigned to x by a series of functions $\{g_1, \dots, g_m\}$. In equation 3.2, the number of trees m is a hyperparameter that should be pre-specified before the experiment, while $\theta \coloneqq \{(T_1, \mu_1), \dots, (T_m, \mu_m), \sigma\}$ is the set of parameters to be determined during the experiment. Compared to ordinary decision tree models, BART is a Bayesian model based on Bayes' Theorem. Hence, building the BART model requires two steps:

Step 1: Specify the prior distribution for all the parameters in set θ.

Step 2: Draw posterior distributions using Gibbs sampler and Metropolis-Hastings algorithm.

We will show the details of the two steps in the following part.

3.1 Prior Specification

To construct priors for the parameter set θ, it is trivial to make independence assumptions. Here, it is assumed that all the trees in the model are independent, i.e. $T_i \perp T_j$, and all the leaf nodes within the same tree and between trees are also independent of each other. As a result, we could decompose the prior distribution of the parameters:

$$
\begin{aligned}
p((T_1, \mu_1), \dots, (T_m, \mu_m), \sigma) &= [\prod_{j=1}^{m} p(T_j, \mu_j)] p(\sigma^2) \\
&= [\prod_{j=1}^{m} p(T_j) p(\mu_j \mid T_j)] p(\sigma^2) \\
&= [\prod_{j=1}^{m} p(T_j) [\prod_{i=1}^{L_j} p(\mu_{ji} \mid T_j)]] p(\sigma^2)).
\end{aligned}
\tag{3.3}
$$

3.1.1 The Leaf Node Prior

We assume the normal distribution for each single leaf node, i.e. $\mu_{ji} \mid T_j \sim \mathcal{N}(\mu_\mu, \sigma_\mu^2)$. The prior of $f(x)$ follows $\mathcal{N}(m\mu_\mu, m\sigma_\mu^2)$. The hyperparameter μ_μ, σ_μ are set by solving the following equations:

$$
\begin{aligned}
y_{max} &= m\mu_\mu + k\sqrt{m}\sigma_\mu, \\
y_{min} &= m\mu_\mu - k\sqrt{m}\sigma_\mu,
\end{aligned}
\tag{3.4}
$$

where y_{max}, y_{min} are derived from the observation data, k is a hyperparameter which could be tuned [21].

In software implementations, it used linear transformations to set μ_μ be zero. In the following sections, we also assume μ_μ to be zero for the sake of simplicity in posterior calculations. Therefore, we could refer to the leaf node prior as:

$$\mu_j \mid T_j \sim \mathcal{N}(0, \tau I_{L_j}), \tag{3.5}$$

where τ denotes the variance of the kth leaf node μ_{jk} conditioning on T_j after the linear transformation and could be computed via the following equation:

$$\tau := \frac{\max(y) - \min(y)}{2k\sqrt{m}}. \tag{3.6}$$

3.1.2 Error Variance Prior

The prior distribution for σ^2 is set to be inverse Gamma distribution with hyperparameter ν and λ:

$$\sigma^2 \sim InvGamma(\nu/2, \nu\lambda/2). \tag{3.7}$$

λ is chosen based on $\hat{\sigma}$, which is the residual standard deviation of simple linear regression $Y = X\beta + \epsilon$, such that:

$$P(\sigma < \hat{\sigma}) = q. \tag{3.8}$$

Here, the hyperparameter pair (ν, q) should be chosen before the experiment.

3.1.3 Tree Prior

The prior distribution of the trees consists of two parts:

1. A prior on the shape of tree T_j.
2. A prior for the splitting rules $\{x_b \leq h_b\}$ for each branch node of the tree, where x_b is a predictor variable(part of x) and h_b is chosen from available values at the branch node by the discrete uniform distribution [21].

For the first part, we should think about a function that limits the depth of trees so that each tree is only a weak learner. A node at depth d is a a branch(non-leaf node) with prior probability

$\frac{\alpha}{(1+d)^{\beta}}$, with $D_n = 1$ indicating that the nth node at depth $d(n)$ is branch. Here, n is the node index in the tree, and $d(n) = \lfloor log_2(n) \rfloor$. Thus, $D_n = 1$ has the following probability:

$$\mathbb{P}(D_n = 1) = \frac{\alpha}{(1+d)^{\beta}}, \quad 0 < \alpha < 1, \beta > 0. \tag{3.9}$$

The hyperparameter pair (α, β) is pre-selected to make the decision tree shallow. The hyperparameter setting will be further explained in Section 3.5.1. Assuming that there are p predictor variables and the probability to select each predictor is equal to $1/p$, the prior probability for the entire tree T_j could be expressed as:

$$\begin{aligned}\mathbb{P}(T_j) = &\prod_{\gamma} \mathbb{P}(D_n = 1)\mathbb{P}(\text{split on predictor i})\mathbb{P}(\text{select the kth value of predictor j to split}) \\ &\times \prod_{\mu}[1 - \mathbb{P}(D_n = 1)] \\ = &\prod_{\gamma} \frac{\alpha}{(1+d)^{\beta}} \times \frac{1}{p} \times \mathbb{P}(\text{select the kth value of predictor j to split}) \\ &\times \prod_{\mu}[1 - \frac{\alpha}{(1+d)^{\beta}}],\end{aligned} \tag{3.10}$$

where γ is the set of branch node indices and μ is the set of leaf node indices.

3.1.4 Summary of Prior Specification

Probability model:

$$y_i \mid \{(T_j, \mu_j)\}_{j=1}^{m}, \sigma \sim \mathcal{N}(\sum_{j=1}^{m} g_j(x_i; T_j, \mu_j), \sigma^2).$$

The priors for BART could be summarized as:

$$p((T_1,\mu_1),\ldots,(T_m,\mu_m),\sigma) = [\prod_{j=1}^{m} p(T_j)p(\mu_j \mid T_j)]p(\sigma^2),$$

$$\mu_j \mid T_j \sim \mathcal{N}(\vec{0}, \tau I_{L_j}),$$

$$\begin{aligned}\mathbb{P}(T_j) = \prod_{\gamma} &\mathbb{P}(D_n = 1)\mathbb{P}(\text{split on predictor i}) \\ &\times \mathbb{P}(\text{select the kth value of predictor j to split}) \\ &\times \prod_{\mu}[1 - \mathbb{P}(D_n = 1)], \\ &\sigma^2 \sim InvGamma(\nu/2, \nu\lambda/2).\end{aligned}$$

3.2 Posterior Draw

Following Bayes' Theorem, the posterior distribution is given below:

$$p(\{(T_j,\mu_j)\}_{j=1}^m, \sigma \mid y) \propto p(y \mid \{(T_j,\mu_j)\}_{j=1}^m, \sigma, X)p(\{(T_j,\mu_j)\}_{j=1}^m, \sigma). \tag{3.11}$$

It is tricky to derive the posterior draws of tree structures, leaf node means, and the error variance. Chipman et al.developed a strategy that used Metropolis-Hastings within a Gibbs sampler to obtain posterior draws [4]. It is called the MCMC Backfitting algorithm [4]. Gibbs sampling is the primary structure used to derive posterior distributions.

Definition 14 ⟨Gibbs Sampler [5]⟩

The process in a Gibbs Sampler is described as follows:

- *Initialize* $x^{(0)} = (x_1, \ldots, x_D) \sim q(x)$
- *For iteration* $i = 1, 2, \ldots$ *do*
  - ◇ $x_1^{(i)} \sim p(X_1 = x_1 \mid X_2 = x_2^{(i-1)}, X_3 = x_3^{(i-1)}, \ldots, X_D = x_D^{(i-1)})$
  - ◇ $x_2^{(i)} \sim p(X_1 = x_1 \mid X_1 = x_1^{(i-1)}, X_3 = x_3^{(i-1)}, \ldots, X_D = x_D^{(i-1)}) \vdots$
  - ◇ $x_D^{(i)} \sim p(X_D = x_D \mid X_1 = x_1^{(i-1)}, X_2 = x_2^{(i-1)}, \ldots, X_{D-1} = x_{D-1}^{(i-1)})$

Let pair (T_{-j}, μ_{-j}) denotes $\{(T_i, \mu_i)\}_{i=1}^{m} \setminus (T_j, \mu_j)$,then we could make one posterior draw of $\{(T_j, \mu_j)\}_{j=1}^{m}$ by the following procedure:

Procedure 3.1: Posterior draw of $\{(T_j, \mu_j)\}_{j=1}^{m}$ and σ

```
Sample $T_1, \mu_1 \mid (T_{-1}, \mu_{-1}), \sigma, y$
Sample $T_2, \mu_2 \mid (T_{-2}, \mu_{-2}), \sigma, y$
    $\vdots$
Sample $T_m, \mu_m \mid (T_{-m}, \mu_{-m}), \sigma, y$
Sample $\sigma \mid \{(T_j, \mu_j)\}_{j=1}^{m}, y$
```

Definition 15 ⟨Partial residual of observations [4]⟩

The jth partial residual of the ith observation in the BART model is defined as:

$$r_{ji} = y_i - \sum_{h \neq j} g_h(x_i; T_h, \mu_h). \tag{3.12}$$

The general form of the j-th partial residual in the BART model is defined as:

$$r_j = y - \sum_{h \neq j} g_h(x; T_h, \mu_h). \tag{3.13}$$

Since

$$r_{ji} = [y_i - \sum_{h=1}^{m} g_j(x_i; T_j, \mu_j)] + g_j(x_i; T_j, \mu_j), \tag{3.14}$$

then r_{ji} could also be written as:

$$r_{ji} = g_j(x_i; T_j, \mu_j) + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, \sigma^2). \tag{3.15}$$

Hence, the likelihood of r_{ji} conditional on $(T_j, \mu_j), \sigma$ is:

$$p(r_{ji} \mid (T_j, \mu_j), \sigma) \sim \mathcal{N}(g_j(x_i; T_j, \mu_j), \sigma^2). \tag{3.16}$$

We could replace $(T_{-i}, \mu_{-i}), \sigma, y$ with r_i in Procedure 3.1 to derive posterior draw of $\{T_j, \mu_j\}_{j=1}^m$. Therefore, the posterior draw of a single regression tree (T_j, μ_j) could be written as: $p(T_j, \mu_j \mid r_j, \sigma)$. It can also be further decomposed using Bayes' Theorem.

$$p(T_j, \mu_j \mid r_j, \sigma) \propto p(\mu_j \mid T_j, r_j, \sigma) p(T_j \mid r_j, \sigma). \tag{3.17}$$

Following the notation of r_i and the decomposition stated above, Procedure 3.1 could be modified to the following procedure. Compared with Procedure 3.1, the following procedure based on partial residuals r_i allows the Metropolis-Hastings algorithm to be used to draw posterior trees from conditional distributions.

Procedure 3.2: Posterior draw of $\{(T_j, \mu_j)\}_{j=1}^m$ and σ

```
Compute $r_1$
Sample $T_1 \mid r_1, \sigma$
Sample $\mu_1 \mid T_1, r_1, \sigma$
Compute $r_2$
Sample $T_2 \mid r_2, \sigma$
Sample $\mu_2 \mid T_2, r_2, \sigma$
        $\vdots$
Compute $r_m$
Sample $T_m \mid r_m, \sigma$
Sample $\mu_m \mid T_m, r_m, \sigma$
Sample $\sigma \mid \{(T_j, \mu_j)\}_{j=1}^m, y$
```

3.2.1 Posterior Draw of Tree

To propose a new tree from the old tree, an additional algorithm must be used. Chipman et al. used the Metropolis-Hastings algorithm to generate a candidate tree [4].

Definition 16 ⟨Metropolis-Hastings algorithm [5]⟩

The process in the Metropolis-Hastings algorithm is described as follows:

- *Start from an initial state $y^{(0)}$ and $t = 0$*
- *For iteration $t = 0, 1, \cdots$ do*
    - *Sample y^* from a proposal distribution $q(y^* \mid y^{(t)})$*

- ◇ *Compute the acceptance probability α, defined as:*

$$\alpha(y^*, y^{(t)}) = min\{1, \frac{p(y^*)q(y^{(t)} \mid y^*))}{p(y^{(t)})q(y^* \mid y^{(t)})}\} \tag{3.18}$$

- ◇ *Sample $U \sim U(0, 1)$*
- ◇ *If $U \leq \alpha$, then:*

$$y^{(t+1)} \leftarrow y^* \tag{3.19}$$

- ◇ *Else:*

$$y^{(t+1)} \leftarrow y^{(t)} \tag{3.20}$$

There are four possible ways to generate a candidate tree T^* from current tree T_j:

1. Grow: Randomly chooses a leaf node of current tree T_j and splits it further into left and right children;
2. Prune: Randomly chooses a branch node(non-leaf node) where both the children are leaf nodes and prunes the two leaf nodes to make the branch node a leaf node;
3. Change: Randomly chooses a branch node(non-leaf node) and changes its splitting rule;
4. Swap: Randomly chooses a parent-child pair which are both branch nodes and swap their splitting rules;

However, Pratola et al. demonstrated that only grow and prune proposals are necessary for generating tree candidates [17]. Typically, only these two proposals are implemented in BART models by software packages. The probabilities of modifying the current tree T_j with grow proposal or prune proposal are pre-specified.

$$\mathbb{P}_{\text{grow}}(T_j) = 0.5 \qquad \qquad \mathbb{P}_{\text{prune}}(T_j) = 0.5 \tag{3.21}$$

If we sample ξ from a Bernoulli distribution with $p = 0.5$, then $\xi = 1$ indicates that we will choose a grow proposal, otherwise we will choose a prune proposal. Let us now consider how to use the Metropolis-Hastings algorithm(Definition 16) to generate a new tree structure. Since the prune proposal is simply an inverse operation of grow proposal, the details for calculating acceptance probability α are nearly the same. We will go over the details of grow proposal. Since

$p(T_j \mid r_j, \sigma) \propto p(r_j \mid T_j, \sigma)p(T_j)$, and $p(T_j)$ represents the prior distribution of tree T_j, then the acceptance probability is:

$$\alpha(T^*, T_j) = min\{1, \frac{p(r^* \mid T^*, \sigma)p(T^*)q(T_j \mid T^*)}{p(r_j \mid T_j, \sigma)p(T_j)q(T^* \mid T_j)}\}. \tag{3.22}$$

Thus, the procedure for generating a single tree T_j in Procedure 3.2 is described below:

Procedure 3.3 Generate a single tree T_j

- Sample $w \sim Bernoulli(p = 0.5)$
- If $w = 1$, then: run grow proposal
- Else: run prune proposal
- Calculate acceptance probability α:

$$\alpha(T^*, T_j) = min\{1, \frac{p(r^* \mid T^*, \sigma)p(T^*)q(T_j \mid T^*)}{p(r_j \mid T_j, \sigma)p(T_j)q(T^* \mid T_j)}\} \tag{3.23}$$

- Sample $U \sim U(0, 1)$
- if $U \leq \alpha$, then: $\quad T_{j+1} \leftarrow T^*$
- Else: $\quad T_{j+1} \leftarrow T_j$

Here, we only consider one iteration in the Metropolis-Hastings algorithm since it is inside a Gibbs sampler. Thus, the only thing left to derive the posterior draw of T_j is to calculate α. Figure 3.3 and 3.4 visualize the current tree T_j and candidate tree T^* in a grow proposal respectively.

In Equation 3.23, $p(T_j)$ and $p(T^*)$ are prior distributions of trees which are pre-defined in Section 3.1. $q(T^* \mid T_j)$ is the transition probability from T_j to T^* with a grow proposal, while $q(T_j \mid T^*)$ represents the transition probability from T^* toT_j with a prune proposal. The following three steps determine the transition probability $q(T_j \mid T^*)$:

1. Randomly choose a leaf node and turn it into a branch node.
2. Randomly choose a predictor x_j for the splitting rule.
3. Randomly choose a cutoff point b_j from the observation values of x_j to split at.

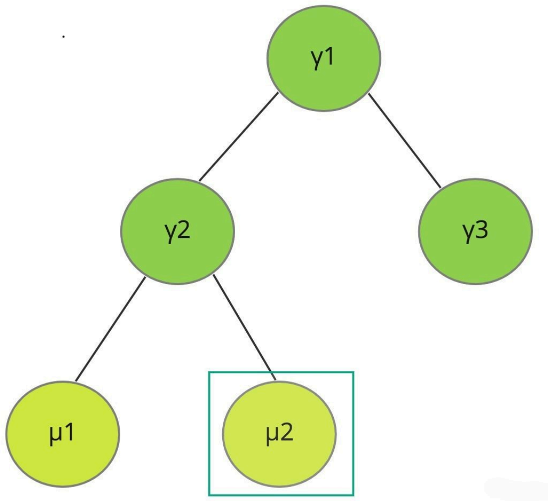

Figure 3.3 Current tree T_j in a grow proposal

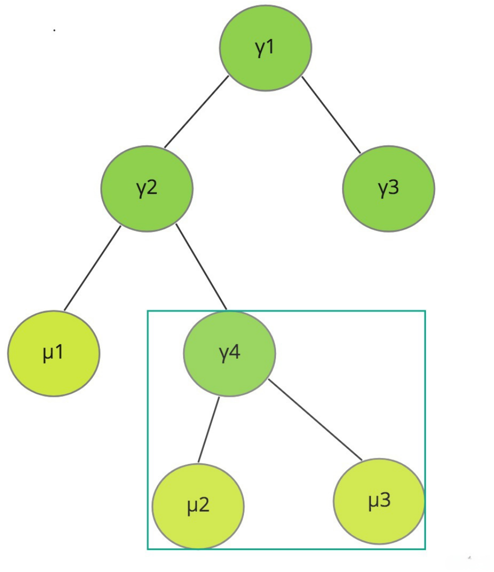

Figure 3.4 Candidate tree T^* in a grow proposal

Thus, $q(T^* \mid T_j)$ for a grow proposal is calculated as follows:

$$\begin{aligned}
q(T^* \mid T_j) = {} & P_{\text{grow}}(T_j) \times P(\text{Selecting a leaf node } h) \\
& \times P(\text{Selecting a predictor variable } x_j) \times P(\text{Selecting the kth observed value of } x_j) \\
= {} & 0.5 \times \frac{1}{L} \times \frac{1}{numberofavailablepredictorstospliton} \\
& \times \frac{1}{\text{number of available observed values of } x_j \text{ as cutoff point}},
\end{aligned} \tag{3.24}$$

where L is the number of leaf nodes in the tree T_j.

To make a prune proposal, we simply choose a branch node whose children are all leaf nodes at random and remove these two child nodes. Thus, $q(T^* \mid T_j)$ is calculated as follows:

$$\begin{aligned} q(T_j \mid T^*) &= P_{\text{prune}}(T^*) \times P(\text{Selecting a branch node whose children are all leaf nodes}) \\ &= 0.5 \times \frac{1}{\text{number of branch nodes whose children are all leaf nodes}}. \end{aligned} \tag{3.25}$$

Thus, the only way to derive $\alpha(T^*, T_j)$ is to get $p(r_j \mid T_j, \sigma)$ since $p(r^* \mid T^*, \sigma)$ could be calculated in the same way. It is simple to deduce from Definition 15 that for a single observation i, $p(r_{ji} \mid T_j, \mu_j, \sigma) \sim N(g_j(x_i; T_j, \mu_j), \sigma^2)$, We will show $p(r_j \mid T_j, \mu_j, \sigma)$ in a multivariate normal distribution form later. Then, using Bayes' Theorem and integration, we can derive $p(r_j \mid T_j, \sigma)$ as follows:

$$\begin{aligned} p(r_j \mid T_j, \sigma) &= \int p(r_j, \mu_j \mid T_j, \sigma) d\mu_j \\ &= \int p(r_j \mid \mu_j, T_j, \sigma) p(\mu_j \mid T_j) d\mu_j, \end{aligned} \tag{3.26}$$

where $p(\mu_j \mid T_j)$ is the prior distribution and has been specified in Section 3.1.

g_j is an indicator function that maps one observation x to the a single leaf node h. We assume that tree T_j has L_j leaf nodes and that there are n observations. Consider the predictor matrix $X = (x_1, x_2, \ldots, x_n)^t$ with all the observations included, thus the dimensions of X is $n \times p$. g_j maps matrix X to vector W_j with dimensions $n \times 1$, and W_j is defined with the help of basis matrix $\hat{X}_j$ with dimensions $n \times L_j$:

Definition 17 ⟨**helper matrices [5]**⟩

In order to clearly derive the posterior distribution formulas in the following parts, two helper matrices are defined as follows:

$$\hat{X}_j := \begin{bmatrix} \mathbf{1}(x_1 \in R_{j1}) & \mathbf{1}(x_1 \in R_{j2}) & \cdots & \mathbf{1}(x_1 \in R_{jL_j}) \\ \mathbf{1}(x_2 \in R_{j1}) & \mathbf{1}(x_2 \in R_{j2}) & \cdots & \mathbf{1}(x_2 \in R_{jL_j}) \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{1}(x_n \in R_{j1}) & \mathbf{1}(x_n \in R_{j2}) & \cdots & \mathbf{1}(x_n \in R_{jL_j}) \end{bmatrix}, \tag{3.27}$$

$$
\begin{aligned}
W_j \coloneqq & \begin{bmatrix} \mu_{j1}\mathbf{1}(x_1 \in R_{j1}) + \mu_{j2}\mathbf{1}(x_1 \in R_{j2}) + \cdots + \mu_{j1}\mathbf{1}(x_1 \in R_{jL_j}) \\ \mu_{j1}\mathbf{1}(x_2 \in R_{j1}) + \mu_{j2}\mathbf{1}(x_2 \in R_{j2}) + \cdots + \mu_{jL_j}\mathbf{1}(x_2 \in R_{jL_j}) \\ \vdots \\ \mu_{j1}\mathbf{1}(x_n \in R_{j1}) + \mu_{j1}\mathbf{1}(x_n \in R_{j2}) + \cdots + \mu_{j1}\mathbf{1}(x_n \in R_{jL_j}) \end{bmatrix} \\
=& \hat{X}_j \times \mu_j \qquad (3.28) \\
=& \begin{bmatrix} \mathbf{1}(x_1 \in R_{j1}) & \mathbf{1}(x_1 \in R_{j2}) & \cdots & \mathbf{1}(x_1 \in R_{jL_j}) \\ \mathbf{1}(x_2 \in R_{j1}) & \mathbf{1}(x_2 \in R_{j2}) & \cdots & \mathbf{1}(x_2 \in R_{jL_j}) \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{1}(x_n \in R_{j1}) & \mathbf{1}(x_n \in R_{j2}) & \cdots & \mathbf{1}(x_n \in R_{jL_j}) \end{bmatrix} \times \begin{bmatrix} \mu_{j1} \\ \mu_{j2} \\ \vdots \\ \mu_{jL_j} \end{bmatrix},
\end{aligned}
$$

where, R_{jh} represents a partition of the covariate space corresponding to leaf node h in tree T_j, μ_{jh} defines the leaf node mean corresponding to leaf node h in tree T_j, and L_j is the total number of leaf nodes in tree T_j.

Assuming the independence of distinct observations, with the notation that $r_j = (r_{j1}, r_{j2}, \ldots, r_{jn})^T$ is a $n \times 1$ vector and $\mu_j = (\mu_{j1}, \mu_{j2}, \ldots, \mu_{jL_j})^T$ is a $L_j \times 1$ vector, we could express $p(r_j \mid T_j, \mu_j, \sigma)$ in a multivariate normal distribution form:

$$r_j \mid T_j, \mu_j, \sigma \sim N(\hat{X}_j \mu_j, \sigma^2 I_n), \tag{3.29}$$

$$p(r_j \mid T_j, \mu_j, \sigma) = (2\pi)^{-n/2} \det(\sigma^2 I_n)^{-1/2} exp\{-\frac{1}{2\sigma^2}(r_j - \hat{X}_j \mu_j)^T (r_j - \hat{X}_j \mu_j)\}. \tag{3.30}$$

In the BART model discussed above, the concrete form of $p(r_j \mid T_j, \sigma)$ could be derived in the following theorem.

Theorem 1

The distribution of partial residual r_j conditional on (T_j, σ) is:

$$r_j \mid T_j, \sigma \sim \mathcal{N}(\vec{0}, \sigma^2 I_n + \tau \hat{X}_j \hat{X}_j^T). \tag{3.31}$$

Proof.

$$\begin{aligned}
p(r_j \mid T_j, \sigma) &= \int p(r_j \mid T_j, \mu_j, \sigma) p(\mu_j \mid T_j) d\mu_j \\
&= (2\pi)^{-(n+L_j)/2} \det(\sigma^2 I_n)^{-1/2} \det(\tau L_j)^{-1/2} \\
&\quad \times \int \exp\{-\frac{1}{2}[(r_j - \hat{X}_j \mu_j)^T (\sigma^2 I_n)^{-1} (r_j - \hat{X}_j \mu_j)]\} \exp\{-\frac{1}{2} \times \mu_j^T (\tau I_{L_j})^{-1} \mu_j\} d\mu_j \\
&= (2\pi)^{-(n+L_j)/2} \det(\sigma^2 I_n)^{-1/2} \det(\tau L_j)^{-1/2} \\
&\quad \times \int \exp\{-\frac{1}{2}[(r_j - \hat{X}_j \mu_j)^T (\sigma^2 I_n)^{-1} (r_j - \hat{X}_j \mu_j) + \mu_j^T (\tau I_{L_j})^{-1} \mu_j]\} d\mu_j \\
&= (2\pi)^{-(n+L_j)/2} \det(\sigma^2 I_n)^{-1/2} \det(\tau L_j)^{-1/2} \exp\{-\frac{r_j^T r_j}{2\sigma^2}\} \\
&\quad \times \int \exp\{-\frac{1}{2}[\mu_j^T (\sigma^{-2} \hat{X}_j^T \hat{X}_j + \tau^{-1} I_{L_j}) \mu_j - 2 r_j^T (\sigma^2 I_n)^{-1} \hat{X}_j \mu_j]\} d\mu_j.
\end{aligned}$$

With the following definitions of extra variables:

$$\begin{aligned}
B :=& \sigma^{-2} \hat{X}_j^T \hat{X}_j + \tau^{-1} I_{L_j}, \\
a :=& \sigma^{-2} B^{-1} \hat{X}_j^T r_j], \\
c :=& -\sigma^{-4} r_j T \hat{X}_j B^{-1} \hat{X}_j^T r_j,
\end{aligned}$$

yields

$$\begin{aligned}
p(r_j \mid T_j, \sigma) &= (2\pi)^{-(n+L_j)/2} \det(\sigma^2 I_n)^{-1/2} \det(\tau L_j)^{-1/2} \exp\{-\frac{r_j^T r_j}{2\sigma^2}\} \exp\{-\frac{c}{2}\} \\
&\quad \times \int \exp\{-\frac{1}{2}[(\mu_j - a)^T B (\mu_j - a)]\} d\mu_j.
\end{aligned} \tag{3.32}$$

According to the properties of multivariate Gaussian integral,

$$\int \exp\{-\frac{1}{2}[(\mu_j - a)^T B (\mu_j - a)]\} d\mu_j = (2\pi)^{L_j/2} \det(B^{-1})^{1/2}. \tag{3.33}$$

Thus,

$$
\begin{aligned}
p(r_j \mid T_j, \sigma) &= (2\pi)^{-(n+L_j)/2} \det(\sigma^2 I_n)^{-1/2} \det(\tau L_j)^{-1/2} \\
&\quad \times \exp\{-\frac{r_j^T r_j}{2\sigma^2} - \frac{c}{2}\} \times (2\pi)^{L_j/2} det(B^{-1})^{1/2} \\
&= (2\pi)^{-n/2} \det(\sigma^2 I_n)^{-1/2} det(\tau L_j)^{-1/2} \\
&\quad \times \exp\{-\frac{1}{2} r_j^T [\sigma^{-2} I_n - \sigma^{-4} \hat{X}_j B^{-1} \hat{X}_j^T] r_j\} \times \det(B^{-1})^{1/2}.
\end{aligned}
$$

Applying Woodbury matrix inverse formula, yields:

$$
\begin{aligned}
p(r_j \mid T_j, \sigma) &= (2\pi)^{-n/2} \det(\sigma^2 I_n)^{-1/2} det(\tau L_j)^{-1/2} \det([\sigma^{-2} \hat{X}_j^T \hat{X}_j + \tau^{-1} I_{L_j}]^{-1})^{1/2} \\
&\quad \times \exp\{-\frac{1}{2} r_j^T [\sigma^2 I_n + \tau \hat{X}_j \hat{X}_j^T]^{-1} r_j\}.
\end{aligned}
$$

Using properties of determinant, yields:

$$
p(r_j \mid T_j, \sigma) = (2\pi)^{-n/2} \det(\sigma^2 I_n + \tau \hat{X}_j \hat{X}_j^T)^{-1/2} \times \exp\{-\frac{1}{2} r_j^T [\sigma^2 I_n + \tau \hat{X}_j \hat{X}_j^T]^{-1} r_j\}. \tag{3.34}
$$

Thus, $p(r_j \mid T_j, \sigma)$ follows the form of a multivariate normal distribution and we could say:

$$
r_j \mid T_j, \sigma \sim \mathcal{N}(\vec{0}, \sigma^2 I_n + \tau \hat{X}_j \hat{X}_j^T). \tag{3.35}
$$

□

3.2.2 Posterior Draw of Leaf Node Means

Definition 18

In order to simplify the notations in the derivation of the following theorem, a helper matrix Θ and a helper variable $\tilde{r}_j$ are defined as follows:

$$
\Theta = (\tau^{-1} I_{L_j} + \sigma^{-2} \hat{X}_j^T \hat{X}_j)^{-1}, \tag{3.36}
$$

$$
\tilde{r}_j = \sigma^{-2} \hat{X}_j^T r_j. \tag{3.37}
$$

Theorem 2

In the BART probability model, the posterior distribution of μ_j is as follows:

$$\mu_j \mid r_j, T_j, \sigma \sim \mathcal{N}(\Theta\tilde{r}_j, \Theta). \tag{3.38}$$

Proof. According to Procedure 3.2, to derive the posterior draw of leaf node means is to calculate $p(\mu_j \mid r_j, T_j, \sigma)$ and sample from this posterior distribution. Thus, using Bayes' Theorem,

$$p(\mu_j \mid r_j, T_j, \sigma) \propto p(r_j \mid \mu_j, T_j, \sigma) \times p(\mu_j \mid T_j). \tag{3.39}$$

$p(\mu_j \mid T_j)$ is a prior distribution and has been pre-specified in Section 3.1. From Equation 3.30, $r_j \mid \mu_j, T_j, \sigma \sim \mathcal{N}(r_j - \hat{X}_j\mu_j, \sigma^2 I_n)$, thus,

$$\begin{aligned} p(\mu_j \mid r_j, T_j, \sigma) \propto & p(r_j \mid \mu_j, T_j, \sigma)p(\mu_j \mid T_j) \\ & \propto \exp\{-\frac{1}{2}[(r_j - \hat{X}_j\mu_j)^T(\sigma^2 I_n)^{-1}(r_j - \hat{X}_j\mu_j) + \mu_j^T(\tau^2 I_{L_j})^{-1}\mu_j]\}. \end{aligned}$$

By multiple matrix operations, yields:

$$p(\mu_j \mid r_j, T_j, \sigma) \propto \exp\{-\frac{1}{2}[(\mu_j - \Theta\tilde{r}_j)^T\Theta^{-1}(\mu_j - \Theta\tilde{r}_j)]\}. \tag{3.40}$$

Because a normal prior is conjugate to a normal likelihood with known variance, $\mu_j \mid r_j, T_j, \sigma$ also follows a multivariate normal distribution. As a result of the preceding derivations, we can obtain that $\mu_j \mid r_j, T_j, \sigma \sim \mathcal{N}(\Theta\tilde{r}_j, \Theta)$. □

3.2.3 Posterior Draw of Error Variance

According to Procedure 3.2, to derive the posterior draw of error variance, one must calculate $p(\sigma^2 \mid \{(T_j, \mu_j)\}_{j=1}^m, y)$ and sample from this posterior distribution. Using Bayes' Theorem,

$$p(\sigma^2 \mid \{(T_j, \mu_j)\}_{j=1}^m, y) \propto p(y \mid \{(T_j, \mu_j)\}_{j=1}^m, \sigma)p(\sigma). \tag{3.41}$$

$\sigma^2 \sim InvGamma(\nu/2, \nu\lambda/2)$ is pre-specified in Section 3.1.

Theorem 3

In the BART probability model, the posterior distribution of σ^2 is as follows:

$$\sigma^2 \mid \{(T_j, \mu_j)\}_{j=1}^m, y \sim InvGamma(\frac{\nu+n}{2}, \frac{1}{2}[y_i - \sum_{j=1}^m g_j(x_i; T_j, \mu_j) + \nu\lambda]). \tag{3.42}$$

Proof. According to the definition of the BART model, for a single observation k, we have:

$$y_k \mid \{(T_j, \mu_j)\}_{j=1}^m, \sigma \sim \mathcal{N}(\sum_{j=1}^m g_j(x_k; T_j, \mu_j), \sigma^2). \tag{3.43}$$

Thus, assuming the independence of distinct observations, $y = (y_1, y_2, \ldots, y_n)^T$ conditioning on $(\{(T_j, \mu_j)\}_{j=1}^m, \sigma)$ yields:

$$y \mid \{(T_j, \mu_j)\}_{j=1}^m, \sigma \sim \mathcal{N}(\vec{g}, \sigma^2 I_n), \tag{3.44}$$

where $\vec{g} := (\sum_{j=1}^m g_j(x_1; T_j, \mu_j), \sum_{j=1}^m g_j(x_2; T_j, \mu_j), \ldots, \sum_{j=1}^m g_j(x_n; T_j, \mu_j))^t$ Thus,

$$\begin{aligned} p(\sigma^2 \mid \{(T_j, \mu_j)\}_{j=1}^m, y) \propto & p(y \mid \{(T_j, \mu_j)\}_{j=1}^m, \sigma) p(\sigma) \\ \propto & (\sigma^2)^{-n/2} \exp\{-\frac{1}{2\sigma^2} \sum_{i=1}^n [y_i - \sum_{j=1}^m g_j(x_i; T_j, \mu_j)]\} \times (\sigma^2)^{-\nu/2-1} \exp\{-\frac{\nu\lambda/2}{\sigma^2}\} \\ \propto & (\sigma^2)^{-(\nu+n)/2-1} \exp\{-\frac{1}{2\sigma^2} \sum_{i=1}^n [y_i - \sum_{j=1}^m g_j(x_i; T_j, \mu_j)] - \frac{\nu\lambda/2}{\sigma^2}\} \\ \propto & (\sigma^2)^{-(\nu+n)/2-1} \exp\{-\frac{1}{2\sigma^2} \sum_{i=1}^n [y_i - \sum_{j=1}^m g_j(x_i; T_j, \mu_j) + \nu\lambda]\}. \end{aligned} \tag{3.45}$$

Hence,

$$\sigma^2 \mid \{T_i, \mu_i\}_{i=1}^m, y \sim InvGamma(\frac{\nu+n}{2}, \frac{1}{2}[y_i - \sum_{j=1}^m g_j(x_i; T_j, \mu_j) + \nu\lambda]). \tag{3.46}$$

□

3.3 Summary of BART Model

The details for calculating a posterior draw of $\{(T_j, \mu_j)\}_{j=1}^m, \sigma$ are now completed. The MCMC Backfitting algorithm is summarized below to sample iter.max posterior draws:

Algorithm 1: Metropolis-Hasting within Gibbs sampler for BART model

```
Input: $n$ observations $\{x_i, y_i\}_{i=1}^n$, hyperparameter set $(\nu, q, k, \alpha, \beta)$, number of MCMC
        iterations iter.max, number of trees $m$.
Output: Posterior draws of $\sum_{j=1}^m g_j(x_i; T_j^{(t)}, \mu_j^{(t)})$ and $\sigma^{(t)}$, for $t = 1, 2, \ldots,$ iter.max.
/* Step 1: Initialization at $t = 0$ */
for $j = 1$ to $m$ do
    Initialize $T_j^{(0)}$ with a single leaf node
    Sample $\mu_j^{(0)} \mid T_j^{(0)}$from prior distribution
end for
Sample $(\sigma^2)^{(0)} \mid \{(T_j^{(0)}, \mu_j^{(0)})\}_{j=1}^m, y$
/* Step 2: Posterior draws */
for $t = 1$ to iter.max do
    for $j = 1$ to $m$ do
        Set $r_{ji} \leftarrow y_i - \sum_{h \neq j} g_h(x_i; T_h, \mu_h)$
        Sample $\xi \sim Bernoulli(p = 0.5), \xi = \begin{cases} 1 \rightarrow \text{a grow proposal} \\ 0 \rightarrow \text{a prune proposal} \end{cases}$
        Sample $T_j^{(t)} \mid r_j, \sigma^{(t-1)}, T_j^{(t-1)}$ from Metropolis-Hasting algorithm according to a
         grow/prune proposal
        Sample $\mu_j^{(t)} \mid r_j, T_j^{(t)}, \sigma^{(t-1)} \sim N(\Theta\tilde{r}_j, \Theta)$
    end for
    Sample $(\sigma^2)^{(t)} \mid \{(T_j^{(t)}, \mu_j^{(t)})\}_{j=1}^m, y \sim IG(\frac{\nu+n}{2}, \frac{1}{2}[y_i - \sum_{j=1}^m g_j(x_i; T_j^{(t)}, \mu_j^{(t)}) + \nu\lambda])$
end for
```

3.4 BART Model for Classification

What has been discussed until now is the BART model for regression with continuous outcomes. According to Chipman et al., BART could also be used for classification with binary outcomes [4]. It could be extended to include classification using the Logit model or Prohit models. The Logit and Prohit link function could map probability $p \in (0, 1)$ to the real axis$(-\infty, +\infty)$.

Definition 19 ⟨BART for classification [4]⟩

The concrete expression of the BART model used for classification is given as follows:

$$\mathbb{P}(y_i = 1 \mid x_i, \{T_j, \mu\_j\}_{j=1}^{m}) = u(\sum_{j=1}^{m} g_j(x_i; T_j, \mu_j)), \tag{3.47}$$

where $u(\alpha) = \frac{1}{1+exp(-\alpha)}$ *or* $u = \Phi$*, the cumulative distribution function of a standard normal distribution.*

Then, let us consider the case when $u = \Phi$.

$$\mathbb{P}(y_i = 1 \mid x_i, \{T_j, \mu_j\}_{j=1}^{m}) = \Phi(\sum_{j=1}^{m} g_j(x_i; T_j, \mu_j)). \tag{3.48}$$

Considering a latent variable Z, which satisfies:

$$\begin{cases} y_i = \mathbf{1}\{Z_i > 0\}, \\ Z_i \sim \mathcal{N}(\lambda, 1), \implies Z_i = \lambda + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, 1). \end{cases} \tag{3.49}$$

Thus, by symmetry of standard normal cdf,

$$\begin{aligned} \mathbb{P}(y_i = 1 \mid x_i, \{T_j, \mu_j\}_{j=1}^{m}) &= \mathbb{P}(Z_i > 0 \mid x_i, \{T_j, \mu_j\}_{j=1}^{m}) \\ &= \mathbb{P}(\lambda + \epsilon_i > 0 \mid x_i, \{T_j, \mu_j\}_{j=1}^{m}) \\ &= \mathbb{P}(\epsilon_i > -\lambda \mid x_i, \{T_j, \mu_j\}_{j=1}^{m}) \\ &= \mathbb{P}(\epsilon_i \leq \lambda \mid x_i, \{T_j, \mu_j\}_{j=1}^{m}) \\ &= \Phi(\lambda). \end{aligned} \tag{3.50}$$

According to Definition 19, we could choose the value λ as:

$$\lambda = \sum_{j=1}^{m} g_j(x_i; T_j, \mu_j). \tag{3.51}$$

Thus,

$$Z_i \sim \mathcal{N}(\sum_{j=1}^{m} g_j(x_i; T_j, \mu_j), 1). \tag{3.52}$$

Meanwhile,

$$\begin{cases} Z_i > 0 & \text{if } y_i = 1, \\ Z_i < 0 & \text{if } y_i = 0. \end{cases} \tag{3.53}$$

The preceding discussions provide us with the intuition to define Z_i conditioning on y_i.

Definition 20 ⟨latent variable Z [3]⟩

We introduce latent variable Z with the following form in the procedure of the BART model for classification.

$$\begin{aligned} Z_i \mid y_i = 1 &\sim max\{\mathcal{N}(\sum_{j=1}^{m} g_j(x_i; T_j, \mu_j), 1), 0\}, \\ Z_i \mid y_i = 0 &\sim min\{\mathcal{N}(\sum_{j=1}^{m} g_j(x_i; T_j, \mu_j), 1), 0\}. \end{aligned} \tag{3.54}$$

As a result, in the procedure of the BART model for binary outcomes, we could use latent variable Z instead of y as continuous outcomes. We draw a new Z_i based on the current $\sum_{j=1}^{m} g_j(x_i; T_j, \mu_j)$ in each MCMC iteration, and then the new Z_i is used to update $\sum_{j=1}^{m} g_j(x_i; T_j, \mu_j)$.

In terms of the BART model's prior distribution specification, the independence assumptions and hyperparameters are the same as in the BART model for continuous outcomes, except for error variance σ^2. The parameter σ is not included in the BART model for classification due to Definition 19. The procedure in Gibbs sampler is nearly identical to Algorithm 1. As a result, the Metropolis-Hastings within Gibbs Sampler for BART with Probit link is summarized below, along with definitions of Θ and $\tilde{r}_j$ in Definition 18.

Algorithm 2: Metropolis-Hasting within Gibbs sampler for BART model with binary outcomes

```
Input: $n$ observations $\{x_i, y_i\}_{i=1}^{n}$, , hyperparameter set $(k, \alpha, \beta)$, number of MCMC
        iterations iter.max, number of trees $m$.
Output: Posterior draws of $\sum_{j=1}^{m} g_j(x_i; T_j^{(t)}, \mu_j^{(t)})$ and $\sigma^{(t)}$, for $t = 1, 2, \ldots,$ iter.max.
/* Step 1: Initialization at $t = 0$                                              */
for $j = 1$ to $m$ do
    Initialize $T_j^{(0)}$ with a single leaf node
    Sample $\mu_j^{(0)} \mid T_j^{(0)}$from prior distribution
end for
/* Step 2: Posterior draws                                                        */
for $t = 1$ to iter.max do
    Sample $Z_i \mid y_i, \{T_j^{(t-1)}, \mu_j^{(t-1)}\}_{j=1}^{m} \sim$
    $\begin{cases} max\{\mathcal{N}(\sum_{j=1}^{m} g_j(x_i; T_j^{(t-1)}, \mu_j^{(t-1)}), 1),\ 0\} & \text{if } \ y_k = 1 \\ min\{\mathcal{N}(\sum_{j=1}^{m} g_j(x_i; T_j^{(t-1)}, \mu_j^{(t-1)}), 1),\ 0\} & \text{if } \ y_k = 0 \end{cases}$ for $j = 1$ to $m$ do
        Set $r_{ji} \leftarrow z_i - \sum_{h \neq j} g_h(x_i; T_h, \mu_h)$
        Sample $\xi \sim Bernoulli(p = 0.5), \xi = \begin{cases} 1 \rightarrow \text{a grow proposal} \\ 0 \rightarrow \text{a prune proposal} \end{cases}$
        Sample $T_j^{(t)} \mid r_j, \sigma^{(t-1)}, T_j^{(t-1)}$ from Metropolis-Hasting algorithm according to a
          grow/prune proposal
        Sample $\mu_j^{(t)} \mid r_j, T_j^{(t)}, \sigma^{(t-1)} \sim \mathcal{N}(\Theta\tilde{r}_j, \Theta)$
    end for
end for
```

3.5 BART implementation

3.5.1 Hyperparameter Setting

3.5.1.1 Hyperparameters for Tree Size

We know from Section 3.1 that the probability of a node at depth d being a branch node is $\alpha(1 + d)^{-\beta}$; thus, with the default setting $\alpha = 0.95,\ \ \beta = 2$ we can get the table of tree size and its corresponding probability.

tree size	1	2	3	4	5
probability	0.05	0.55	0.28	0.09	0.03

Table 3.1 Table to show the distribution of tree size.

Thus, it is likely to have trees of size 2 or 3. This means that the default setting for α and β prefers small trees.

3.5.1.2 Hyperparameters for Error Variance

As a rule of thumb, we have three realizations of a hyperparameter pair (v, q) to choose from. Each of the choices leads to a different probability distribution of σ and thus results in different fitting results. $(10, 0.75)$ leads to conservative fitting, $(3, 0.9)$ leads to default fitting and $(3, 0.99)$ results in aggressive fitting. We use the default setting $(v, q) = (3, 0.9)$ in our implementations.

3.5.1.3 Summary of Hyperparameter Setting

Cross Validation is a common method for fine-tuning hyperparameters. We try several times with different settings of $(\nu, q, k, \alpha, \beta)$, and their differences are slight. A number of articles have mentioned that the default settings for the BART prior are effective and that BART models are easy to use. As a result, they rarely require hyperparameter tuning. For hyperparameter m, increasing the number of trees usually makes a difference for $m < 200$. However, for large m, the difference is marginal, and the computing power and time consumption are excessive.

3.5.2 Inference Statistics

In posterior inference for the BART model, we discard the first 500 iterations and used the next 1000 iterations for posterior inference as the default setting. Let us assume that the posterior draws of $f(x_i)$ are $\{\hat{f}_1(x_i), \hat{f}_2(x_i) \ldots, \hat{f}_{1000}(x_i)\}$, where $\hat{f}_k$ is the sum of trees in one iteration and x_i denotes the ith observation, i.e. as described in Algorithm 1,

$$\hat{f}_k(x_i) = \sum_{j=1}^{m} g_j(x_i; T_j^{(500+k)}, \mu_j^{(500+k)}). \tag{3.55}$$

Thus, the posterior mean of $f(x_i)$ is given by $\tilde{f}(x_i)$:

$$\tilde{f}(x_i) = \frac{1}{1000}\sum_{k=1}^{1000}\hat{f}_k. \tag{3.56}$$

Furthermore, we are able to derive credible intervals from posterior draws. We consider the following two approaches to creating credible intervals. **Method.1** was first proposed by Chipman et al. while we also design **Method.2** ourselves [4].

Method.1 **Directly use quantiles:**

To get a $(1-\alpha)\%$ credible interval is to calculate the upper $\alpha/2$ and lower $\alpha/2$ quantiles of $\{\hat{f}_1(x_i), \hat{f}_2(x_i), \ldots, \hat{f}_{1000}(x_i)\}$.

Method.2 **Add stochastic error:**

From each posterior draw, we could also get $\sigma^{(500+k)}$ and then draw posterior $\hat{y}_k$ by adding a stochastic error which is sampled from a normal distribution, i.e.

$$\begin{aligned}\hat{y}_k &= \hat{f} * k(x_i) + \hat{\epsilon}\\ &= \sum_{j=1}^{m} g_j(x_i; T_j^{(500+k)}, \mu_j^{(500+k)}) + \hat{\epsilon}, \quad \hat{\epsilon} \sim \mathcal{N}(0, \sigma^{(500+k)}),\end{aligned} \tag{3.57}$$

where x_i denotes the ith observation. Then we can construct a $(1-\alpha)\%$ credible interval by calculate the upper $\alpha/2$ and lower $\alpha/2$ quantiles of $\{\hat{y}_1, \hat{y}_2 \ldots, \hat{y}_{1000}\}$

3.5.3 Posterior Inference for Propensity Score

Although the propensity score could be computed by any classification algorithm in principle, we estimate the true propensity score e_i using Prohit BART for more robust results with the following model.

$$e_i = \Phi(\sum_{h=1}^{m} g_h(\tilde{x}_i; T_h, \mu_h)) + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, 1), \tag{3.58}$$

where Φ denotes the cdf of standard normal distribution and the covariates $\tilde{x}_i$ to estimate propensity score e_i may be different from covariates x_i to estimate outcome variable Y_i.

According to the official website of the 2022 ACIC Data Challenge, treatment status can be a function of practice-level covariates X and pre-treatment outcomes. Furthermore, because we choose to work on a practice-level track, we discard patient-level covariates V and added the number of patients in each practice in the pre-treatment stage. Because the organization committee, Mathematica, emphasizes the difference in outcomes Y between years 1 and 2, it is included as an additional covariate.

There are several R packages available to implement the BART model, and we use the "BART" package(Sparapani et al. [6]) for our implementations. The "BART" package includes functions for the BART model that produce both continuous and binary results. It is convenient to select data for training and testing. We use BART::pbart to model propensity scores with the previously mentioned covariates. It is worth noting that the function's output values are not predicted propensity scores $\hat{e}_i$. We need an additional logistic function to convert the posterior draws of a sum of trees $\sum_{j=1}^{m} g_j(x_i; T_j, \mu_j)$ to range $(0, 1)$. The procedure is outlined below.

Algorithm 3: BARТ to estimate propensity score using R package

```
Input: 3400 datasets and number of trees $m$.
Output: Estimated propensity scores for each dataset.
for $i = 1$ to 3400 do
    Read $i$th dataset
    Construct dataframe $df_{pscore}$ with practice-level covariates $X$, pre-treatment outcomes
      $Y_1$ and $Y_2$, number of patients in pre-treatment stage $size_1$ and $size_2$, difference of
      year1 outcome and year2 outcome $diff$
    Convert categorical variables to factors
    Use BayesTree::makeind to dummy-encode factors
    Run BART::pbart with treatment status $Z$ as response variable for training data,
      $df_{pscore}$ as training data as predictors for training data
    Convert BART::pbart outputs to probabilities with a logistic function
    Average the probabilities across iterations to get estimated propensity scores
    Save the $i$th dataset with estimated propensity scores
end for
```

3.5.4 Posterior Inference for SATTs

In this subsection, we will demonstrate how to estimate $SATT_{overall}$. We use the following model to fit outcomes Y in year 3 and year 4.

$$Y_k = f_k(x, Z, \hat{e}(\tilde{x})) + \epsilon_k \quad k \in year3, year4 \tag{3.59}$$

Where x denotes selected predictor covariates except treatment status, Z denotes treatment status, $\hat{e}(x)$ denotes the estimated propensity score which is obtained in Algorithm 5.

With regard to variable selection, for $k = year3$, we select the practice-level covariates X, estimated propensity score $\hat{e}$, treatment status variable Z, pre-treatment outcomes Y_1 and Y_2, number of patients in pre-treatment stage $size_1$ and $size_2$, the difference of year1 outcome and year2 outcome $diff$. While for $k = year4$, we select the same covariates as for $year3$ together with the year 3 outcome $Yn\_patients_3$. It is important to note that we add estimated propensity score as an additional covariate for both year 3 and year 4 to increase the robustness of the model.

Moreover, to reduce the number of covariates, we performed the variable selection algorithms proposed by Bleich et al. [1]. Unfortunately, no significant results were obtained. The selected variables varied too much across different datasets and the prediction results were bad with the selected variables. As a result, they were not used in my implementations.

Since

$$SATT_{overall} = \frac{1}{\sum_{t=3}^{4} N_t} \sum_{t=3}^{4} \sum_{i:Z_i=1} n\_patients_{t,i}(Y_{t,i}(1) - Y_{t,i}(0)), \tag{3.60}$$

with i as the practice id, I built two BART models respectively with R package "BART". I build one BART model with the selected predictors and $Y_{3,i}$ as the response variable with, and then predicted $Y_{3,i}(1)$ and $Y_{3,i}(0)$. I used $Y_{4,i}$ as the response variable with the selected predictors to build another BART model and then predicted $Y_{4,i}(1)$ and $Y_{4,i}(0)$ in the same way. The predicted values of these two BART models are also used to compute annual SATTs and subgroup SATTs. I tried both **Method.1** and **Method.2** in Section 3.5.2 to construct credible intervals. The processes are summarized below.

Algorithm 4: BART to calculate SATTs using R package with **Method.1** to construct credible intervals

```
Input: 3400 datasets and number of trees $m$.
Output: Estimated $SATT_{overall}$, $SATT_{yearly}$ and $SATT_{subgroup}$ and their corresponding
        credible intervals for each dataset.
Initialize record dataset which is used to save estimated SATTs and the upper and lower
  bounds of credible intervals
Initialize $count = 0$ which remembers the index to write the estimations in the record
  dataset
for $i = 1$ to $3400$
    Read $i$th dataset
    Read the record dataset
    Read the estimated propensity score $\hat{e}$ saved in Algorithm 5
    Construct dataframe $df_{year3}$ which consists of the practice-level covariates $X$,
      treatment status variable $Z$, estimated propensity score $\hat{e}$, pre-treatment outcomes $Y_1$
      and $Y_2$, number of patients in pre-treatment stage $size_1$ and $size_2$, difference of year1
      outcome and year2 outcome $diff$ and estimated propensity score $\hat{e}$
    Construct dataframe $df_{year4}$ which consists of the practice-level covariates $X$,
      treatment status variable $Z$, estimated propensity score $\hat{e}$, pre-treatment outcomes $Y_1$
      and $Y_2$, number of patients in pre-treatment stage $size_1$ and $size_2$, difference of year1
      outcome and year2 outcome $diff$, year3 outcomes $Y_3$ and estimated propensity score
      $\hat{e}$
    Convert categorical variables in dataframes $df_{year3}$ and $df_{year4}$ to factors
    Use BayesTree::makeind to dummy-encode factors in dataframes $df_{year3}$ and $df_{year4}$
    Select rows in dataframe $df_{year3}$ with $Z = 1$ and save it as $df_{year3}^{target}$
    Change the column of $Z$ in $df_{year3}^{target}$ to $(0, 0, \ldots, 0)^t$ and save it as $\check{df}_{year3}^{target}$
    Select rows in dataframe $df_{year4}$ with $Z = 1$ and save it as $df_{year4}^{target}$
    Change the column of $Z$ in $df_{year4}^{target}$ to $(0, 0, \ldots, 0)^t$ and save it as $\check{df}_{year4}^{target}$
    Run BART::wbart with $Y_3$ as response variable for training data, $df_{year3}$ as predictors
      for training data, $rbind(df_{year3}^{target}, \check{df}_{year3}^{target})$ as predictors for test data
    Run BART::wbart with $Y_4$ as response variable for training data, $df_{year4}$ as predictors
      for training data, $rbind(df_{year4}^{target}, \check{df}_{year4}^{target})$ as predictors for test data
```

Algorithm 4: BART to calculate SATTs using R package with **Method.1** to construct credible intervals

```
    Initialize 15 numeric vectors each with length 1000 for $satt_{overall}$, $satt_{year3}$, $satt_{year4}$
      and 12 subgroup SATTs
    Use predictions from the two BART models and the number of patients of
      corresponding practice in year 3 $n\_patients_{year3}$ and year 4 $n\_patients_{year4}$ to
      calculate SATTs in 1000 posterior draws and save them in the 15 numeric vectors
    Average each of 15 numeric vectors to get the final estimation for
      $satt_{overall}$, $satt_{year3}$, $satt_{year4}$ and 12 subgroup SATTs and save it to the record dataset
    Construct credible intervals for 15 SATTs with 0.05 and 0.95 quantiles of their
      corresponding numeric vectors as lower and upper bounds and save them to the
      record dataset
    Update $count \leftarrow count + 15$
end for
```

Algorithm 5: BART to calculate SATTs using R package with **Method.2** to construct credible intervals

```
Input: 3400 datasets and number of trees $m$.
Output: Estimated $SATT_{overall}$, $SATT_{yearly}$ and $SATT_{subgroup}$ and their corresponding
        credible intervals for each dataset.
Initialize record dataset which is used to save estimated SATTs and the upper and lower
  bounds of credible intervals
Initialize $count = 0$ which remembers the index to write the estimations in the record
  dataset
Set $num\_samples = 10$ which reflects number of posterior $Y_i$ sampled from each
  posterior draw $N(\sum_{k=1}^{m} g_k(x_i; T_k^{(t)}, \mu_k^{(t)}), \sigma^{(t)})$, $t \in \{1, 2, \ldots, 1000\}$
for $i = 1$ to $3400$
    Read $i$th dataset
    Read the record dataset
    Read the estimated propensity score $\hat{e}$ saved in Algorithm 5
    Construct dataframe $df_{year3}$ which consists of the practice-level covariates $X$,
      treatment status variable $Z$, estimated propensity score $\hat{e}$, pre-treatment outcomes $Y_1$
      and $Y_2$, number of patients in pre-treatment stage $size_1$ and $size_2$, difference of year1
      outcome and year2 outcome $diff$ and estimated propensity score $\hat{e}$
    Construct dataframe $df_{year4}$ which consists of the practice-level covariates $X$,
      treatment status variable $Z$, estimated propensity score $\hat{e}$, pre-treatment outcomes $Y_1$
      and $Y_2$, number of patients in pre-treatment stage $size_1$ and $size_2$, difference of year1
      outcome and year2 outcome $diff$, year3 outcomes $Y_3$ and estimated propensity score
      $\hat{e}$
    Convert categorical variables in dataframes $df_{year3}$ and $df_{year4}$ to factors
    Use BayesTree::makeind to dummy-encode factors in dataframes $df_{year3}$ and $df_{year4}$
    Select rows in dataframe $df_{year3}$ with $Z = 1$ and save it as $df_{year3}^{target}$
    Change the column of $Z$ in $df_{year3}^{target}$ to $(0, 0, \ldots, 0)^t$ and save it as $\check{df}_{year3}^{target}$
    Select rows in dataframe $df_{year4}$ with $Z = 1$ and save it as $df_{year4}^{target}$
    Change the column of $Z$ in $df_{year4}^{target}$ to $(0, 0, \ldots, 0)^t$ and save it as $\check{df}_{year4}^{target}$
    Run BART::wbart with $Y_3$ as response variable for training data, $df_{year3}$ as predictors
      for training data, $rbind(df_{year3}^{target}, \check{df}_{year3}^{target})$ as predictors for test data
```

Algorithm 5: BART to calculate SATTs using R package with **Method.2** to construct credible intervals

```
    for $j = 1$ to $1000$
        Run BART::wbart with $Y_4$ as response variable for training data, $df_{year4}$ as
          predictors for training data, $rbind(df^{target}_{year4}, \check{df}^{target}_{year4})$ as predictors for test data
        Initialize 15 numeric vectors each with length $(1000 \times num\_samples)$ for
          $satt_{overall}, satt_{year3}, satt_{year4}$ and 12 subgroup SATTs
        Extract the posterior $\sigma^{(j)}_{year3},\ \sigma^{(j)}_{year4}$ and
          $\sum_{k=1}^{m} g_k(x_i; T^{(j)}_{k,year3}, \mu^{(j)}_{k,year3}),\ \sum_{k=1}^{m} g_k(x_i; T^{(j)}_{k,year4}, \mu^{(j)}_{k,year4})$ in the $j$th posterior
          draw for the two BART models respectively
        Sample $num\_samples$ $Y_{i3}$ from $N(\sum_{k=1}^{m} g_k(x_i; T^{(j)}_{k,year3}, \mu^{(j)}_{k,year3}), \sigma^{(j)}_{year3})$
        Sample $num\_samples$ $Y_{i4}$ from $N(\sum_{k=1}^{m} g_k(x_i; T^{(j)}_{k,year4}, \mu^{(j)}_{k,year4}), \sigma^{(j)}_{year4})$
        Use sampled $num\_samples$ $(Y_{i3},\ Y_{i4})$ and the number of patients in
          corresponding practice $n\_patients_{year3}$, $n\_patients_{year4}$ to calculate
          $satt_{overall}, satt_{year3}, satt_{year4}$ and 12 subgroup SATTs, then save them to 15
          numeric vectors
    end for
    Use predictions from the two BART models and the number of patients of
      corresponding practice in year 3 $n\_patients_{year3}$ and year 4 $n\_patients_{year4}$ to
      calculate SATTs in 1000 posterior draws and save them in the 15 numeric vectors
    Average each of 15 numeric vectors to get the final estimation for
      $satt_{overall}, satt_{year3}, satt_{year4}$ and 12 subgroup SATTs and save it to the record dataset
    Construct credible intervals for 15 SATTs with 0.05 and 0.95 quantiles of their
      corresponding numeric vectors as lower and upper bounds and save them to the
      record dataset
    Update $count \leftarrow count + 15$
end for
```

4 Bayesian Causal Forest(BCF)

4.1 Regularization-induced confounding(RIC)

Consider a true model with $X = (x_1, x_2, \ldots, , x_n)^t, y = (y_1, y_2, \ldots, y_n)^t$,

$$\begin{aligned}
y_i &= h(x_i) - Z_i + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, 0.6^2) \;\; i.i.d. \\
x_{i1}, x_{i2} &\sim \mathcal{N}(0,1) \;\; i.i.d. \\
h(x_i) &= \begin{cases} 1 & \text{if } \; x_{i1} < x_{i2}, \\ -1 & \text{if } \; x_{i1} \geq x_{i2}. \end{cases} \\
\mathbb{P}(Z_i = 1 \mid x_{i1}, x_{i2}) &= \Phi(h(x_i)).
\end{aligned} \tag{4.1}$$

This example demonstrates the so-called "targeted selection phenomenon". In real life, doctors treat patients who they think need it. If the expected outcome without treatment is large, the probability of treatment increases or decreases. In model 4.1, $\mathbb{E}[y_i \mid Z_i = 0, x_i] = h(x_i)$, thus,

$$\begin{aligned}
\mathbb{P}(Z_i = 1 \mid x_{i1}, x_{i2}) &= \Phi(\mathbb{E}[y_i \mid Z_i = 0, x_i]) \\
&= \begin{cases} \Phi(1) = 0.84, & \text{if } x_{i1} < x_{i2}, \\ \Phi(-1) = 0.16, & \text{if } x_{i1} \geq x_{i2}. \end{cases}
\end{aligned} \tag{4.2}$$

In this case, patients with $x_{i1} < x_{i2}$ are five times more likely to receive treatment owing to their better outcomes when not treated. Hahn et al. first introduced the concept of a phenomenon called "regularization-induced confounding" (RIC) [8]. The regularization-induced confounding phenomenon is consistently produced by targeted selection.

Definition 21 ⟨Regularization induced confounding (RIC) [8]⟩
The regularization-induced confounding occurs when the following conditions are fulfilled:

1. *$h(x_i) \coloneqq \mathbb{E}[y_i \mid Z_i = 0, x_i]$ is complex.*

2. *$e(x_i) \coloneqq \mathbb{P}(Z_i = 1 \mid x_i)$ looks like $h(x_i)$, then misattributing $h(x_i)$ to treatment effect can result in a similar overall fit with a much simpler response surface, which may be favored by a regularization prior.*

In Definition 21, since $e(x_i) \coloneqq \mathbb{P}(Z_i = 1 \mid x_i)$ looks like $h(x_i)$, $h(x_i)$ could be approximated by a tree that splits at Z_i. Given the prior specification in the BART model, which prefers small trees and penalizes the total number of splits, the BART model would rather split on Z_i than x_i. This is referred to as "confusing confounding" for treatment effects. As a result, when only the BART model is used, it is likely to have a higher bias.

A simple solution to solve regularization-induced confounding (RIC) is to add propensity score as an additional covariate. Adding propensity score as an additional covariate makes it simple to deconfound. Section 2.5 discussed using a classification model to estimate propensity score $e(x_i)$. Assuming that the estimated propensity score is $\hat{e}(x_i)$, the new BART model with propensity score is:

$$y_i = f(x_i, \hat{e}(x_i), z_i) + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, \sigma^2). \tag{4.3}$$

4.2 Bayesian Causal Forest(BCF)

Hahn et al. first proposed the Bayesian Causal Forest (BCF), which is a variant of the BART model [8]. BCF is specifically designed for estimating treatment effects. The BART model lacks a direct mechanism for regularizing the treatment effect function. Hahn et al. also proposed the following model to reparameterize the BART model [8]:

$$y_i = h(x_i, \hat{e}(x_i)) + \tau(x_i)z_i + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, \sigma^2). \tag{4.4}$$

Thus,

$$\tau(x_i) = \mathbb{E}[y_i \mid x_i, z_i = 1] - \mathbb{E}[y_i \mid x_i, z_i = 0]. \tag{4.5}$$

Let $\tau(x_i)$ denote the treatment effect function. BCF model consists of two sequential BART models, one for modelling $h(\cdot)$ and the other for modelling $\tau(\cdot)$. Hence, we could apply stricter BART prior on $\tau(\cdot)$ by changing the hyperparameters set (m, α, β) to have fewer trees and smaller sizes of trees. As a result, the BCF model is more robust than the BART model.

Definition 22 ⟨**Bayesian Causal Forest(BCF) model [8]**⟩

In more detail, the BCF could be expressed as

$$y_i = \sum_{l=1}^{L} u_l(x_i, \hat{e}(x_i); T_l, \mu_l) + \sum_{k=1}^{K} v_k(x_i; S_k, \omega_k) + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, \sigma^2), \tag{4.6}$$

where u_l is the function that assigns $\mu_{lj} \in \mu_j$ to $(x_i, \hat{e}(x_i))$,$\{T_1, T_2, \ldots, T_L\}$ denotes L trees for modelling $h(x_i, \hat{e}(x_i))$. $\{\mu_1, \mu_2, \ldots, \mu_L\}$ denotes the mean vectors of the corresponding leaf nodes. v_k is the function that assigns $\omega_{kj} \in \omega_k$ to x_i, $\{S_1, S_2, \ldots, S_K\}$ denotes K trees for modelling $\tau(x_i)$, and their corresponding leaf node mean vectors are $\{\omega_1, \omega_2, \ldots, \omega_K\}$.

In the BCF model, $\mathbb{E}[y_i \mid x_i, \hat{e}(x_i)]$ is approximated by sum of $L + K$ trees.

Definition 23 ⟨**Total residuals [10]**⟩

The total residual in the BCF model is defined as follows:

$$r_i := y_i - \sum_{l=1}^{L} u_l(x_i, \hat{e}(x_i); T_l, \mu_l) - \sum_{j=1}^{K} v_j(x_i; S_k, \omega_k). \tag{4.7}$$

Definition 24 ⟨**Prognostic residual [10]**⟩

The prognostic residual in the BCF model is defined as follows:

$$p := y_i - \sum_{l=1}^{L} u_l(x_i, \hat{e}(x_i); T_l, \mu_l). \tag{4.8}$$

Definition 25 ⟨**Treatment residual [10]**⟩

The treatment residual in the BCF model is defined as follows:

$$tr := y_i - \sum_{j=1}^{K} v_j(x_i; S_j, \omega_j). \tag{4.9}$$

In accordance with Algorithm 6, the update steps for tree T_l, S_j also depend on partial residuals.

Definition 26 ⟨Partial residual [10]⟩

The partial residuals corresponding to the prognostic term or the treatment term in the BCF model are defined as follows:

$$\begin{aligned} r_{li}^{[T]} &:= r_i + u_l(x_i, \hat{e}(x_i); T_l, \mu_l), \\ r_{ji}^{[S]} &:= r_i + v_j(x_i; S_j, \omega_j). \end{aligned} \tag{4.10}$$

The partial residuals defined in Definition 26 have the same form as the partial residuals defined in Definition 15 due to Equation 3.14.

The BCF model could be decomposed into two sequential BART models. It is obvious that $r_i \sim \mathcal{N}(0, \sigma^2)$, and thus

$$\begin{aligned} r_{li}^{[T]} &\sim \mathcal{N}(u_l(x_i, \hat{e}(x_i); T_l, \mu_l), \sigma^2), \\ r_{ji}^{[S]} &\sim \mathcal{N}(v_j(x_i; S_j, \omega_j), \sigma^2). \end{aligned} \tag{4.11}$$

Recalling the partial residual in Section 3.2,

$$r_{ji} \sim \mathcal{N}(g_j(x_i; T_j, \mu_j), \sigma^2). \tag{4.12}$$

They share a similar form and could thus be used to generate posterior draws in the same way. In addition, the mathematical details to derive $p(r_{li}^{[T]} \mid T_l, \sigma)$, $p(\mu_l \mid r_{li}^{[T]}, T_l, \sigma)$, $p(r_{ji}^{[S]} \mid S_j, \sigma)$, $p(\mu_j \mid r_{ji}^{[S]}, S_j, \sigma)$ is the same as described in Section 3.2. We refer $h(\cdot)$ as the prognostic term, $\tau(\cdot)$ as the treatment term. No published article clearly shows the procedure and algorithm for the vanilla BCF model. We summarize the procedure and algorithm for the vanilla BCF model based on contents from Krantsevich et al. and Caron et al. [10, 11]. The procedure for the BCF is summarized as follows:

Step.1 **Update prognostic term:**

We first initialize L trees for modelling $h(\cdot)$ with a single leaf node. The covariate space included the estimated propensity score $\hat{e}(x_i)$. The prior specification follows the same as what has been discussed in Section 3.1, including regularization in tree size, independence assumption of distinct observations, and leaf nodes. The partial residuals updating, on the other hand, is quite different. In this case, the partial residual depends on the total residual.

For each tree $l \in \{1, 2, \ldots, L\}$, $r_l^{[T]}$ is computed. After one loop through trees, the total residual must be updated. For each tree, we sequentially update the following terms:

◇ $T_l \mid r_{li}^{[T]}, \sigma, T_l^{\text{prev}}$

◇ $\mu_l \mid r_{li}^{[T]}, T_l, \sigma$

After one iteration of L trees, we update the total residual r_i and sample new σ.

Step.2 **Update treatment term:**

Then, for modelling $\tau(\cdot)$, we then initialize K trees. For each tree $j \in \{1, 2, \ldots, K\}$, $r_j^{[S]}$ is computed. The main difference is that we have stricter prior regularization. The residuals updating are updated in the same order as in **Step.1**. K tends to be much smaller than L.

◇ $S_k \mid r_{ji}^{[S]}, \sigma, S_k^{\text{prev}}$

◇ $\omega \mid r_{ji}^{[S]}, S_k, \sigma$

After one iteration of K trees, we update the total residual r_i and sample new σ.

Following the notations in Definition 18, we define Θ_1, Θ_2, $\tilde{r}_l^T$ and $\tilde{r}_j^{[S]}$ in posterior draws of the BCF for two sequential BART models.

Definition 27 **⟨matrix Θ and vector $\tilde{r}_j$ in two sequential BART models of BCF separately⟩** *With basis matrix definition in Definition 17, we could easily define basis matrics $\hat{X}_l$ and $\bar{X}_j$, which correspond to the prognostic term and the treatment term respectively.*

Following the definition of Θ and $\tilde{r}_j$ in Definition 18, we could get Θ_1 and $\tilde{r}_l^{[T]}$, Θ_2 and $\tilde{r}_j^{[S]}$ for the two sequential BART models within the BCF model. We denote the covariates used to predict prognostic term as x.

■ *For the lth tree in the posterior draws of prognostic term,*

$$\Theta_1 = (\tau_1^{-1} I_{W_l} + \sigma^{-2} \hat{X}_l^t \hat{X}_l)^{-1}, \tag{4.13}$$

$$\tilde{r}_l^{[T]} = \sigma^{-2} \hat{X}_l^t r_l^{[T]}, \tag{4.14}$$

where the lth tree has W_l leaf nodes, $\{R_{l1}, R_{l2}, \dots, R_{lW_l}\}$ is the partition of covariate space by the lth decision tree, $\mu_l \mid T_l \sim \mathcal{N}(\vec{0}, \tau_1 I_{W_l})$ in the prior specification, and:

$$\hat{X}_l = \begin{bmatrix} \mathbf{1}(x_1 \in R_{l1}) & \mathbf{1}(x_1 \in R_{l2}) & \cdots & \mathbf{1}(x_1 \in R_{lW_l}) \\ \mathbf{1}(x_2 \in R_{l1}) & \mathbf{1}(x_2 \in R_{l2}) & \cdots & \mathbf{1}(x_2 \in R_{lW_l}) \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{1}(x_n \in R_{l1}) & \mathbf{1}(x_n \in R_{l2}) & \cdots & \mathbf{1}(x_n \in R_{lW_l}) \end{bmatrix}. \tag{4.15}$$

Although $\bar{X}_j$ is in a similar form as $\hat{X}_l$, its covariates to predict treatment term may be different from x and we denote them as $\tilde{x}$.

- *For the jth tree in the posterior draws of treatment term,*

$$\Theta_2 = (\tau_2^{-1} I_{D_j} + \sigma^{-2} \bar{X}_j^t \bar{X}_j)^{-1}, \tag{4.16}$$

$$\tilde{r}_j^{[S]} = \sigma^{-2} \bar{X}_j^t r_j^{[S]}, \tag{4.17}$$

where the jth tree has D_j leaf nodes, $\{G_{j1}, G_{j2}, \dots, G_{jD_j}\}$ is the partition of covariate space by the jth decision tree, $\omega_j \mid S_j \sim \mathcal{N}(\vec{0}, \tau_2 I_{D_j})$ in the prior specification, and:

$$\bar{X}_j = \begin{bmatrix} \mathbf{1}(\tilde{x}_1 \in G_{j1}) & \mathbf{1}(\tilde{x}_1 \in G_{j2}) & \cdots & \mathbf{1}(\tilde{x}_1 \in G_{jD_j}) \\ \mathbf{1}(\tilde{x}_2 \in G_{j1}) & \mathbf{1}(\tilde{x}_2 \in G_{j2}) & \cdots & \mathbf{1}(\tilde{x}_2 \in G_{jD_j}) \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{1}(\tilde{x}_n \in G_{j1}) & \mathbf{1}(\tilde{x}_n \in G_{j2}) & \cdots & \mathbf{1}(\tilde{x}_n \in G_{jD_j}) \end{bmatrix}. \tag{4.18}$$

4.3 Summary of BCF Model

Algorithm 6: Bayesian Causal Forest(BCF)

```
Input: $n$ observations $\{x_i, \hat{e}(x_i), y_i\}_{i=1}^{n}$, and hyperparameter set $(\nu, q, k_1, \alpha_1, \beta_1)$ for
       modelling prognostic term, $(\nu, q, k_2, \alpha_2, \beta_2)$ for modelling treatment term, number
       of MCMC iterations iter.max, number of trees $K$ and $L$.
Output: Posterior draws of $[\sum_{l=1}^{L} u_l(x_i, \hat{e}(x_i); T_l^{(t)}, \mu_l^{(t)}) + \sum_{j=1}^{K} v_j(x_i; S_j^{(t)}, \omega_j^{(t)})]$, for
        $t = 1, 2, \dots,$ iter.max.
/* Step 1: Initialization at $t = 0$ */
for $l = 1$ to $L$ do
    Initialize $T_l^{(0)}$ with a single leaf node
    Sample $\mu_l^{(0)} \mid T_l^{(0)}$from prior distribution
end for
for $j = 1$ to $K$ do
    Initialize $S_j^{(0)}$ with a single leaf node
    Sample $\omega_j^{(0)} \mid S_j^{(0)}$from prior distribution
end for
Sample $(\sigma^2)^{(0)} \mid \{(T_l^{(0)}, \mu_l^{(0)})\}_{l=1}^{L}, \{(S_j^{(0)}, \omega_j^{(0)})\}_{j=1}^{K}, y$
Initialize $r_i \leftarrow y_i - \sum_{l=1}^{L} u_l(x_i, \hat{e}(x_i); T_l^{(0)}, \mu_l^{(0)}) - \sum_{j=1}^{K} v_j(x_i; S_j^{(0)}, \omega_j^{(0)})$
/* Step 2: Posterior draws of prognostic term */
for $t = 1$ to $iter.max$ do
    for $l = 1$ to $L$ do
        Set $r_{li}^{[T]} \leftarrow r_i + u_l(x_i, \hat{e}(x_i); T_l, \mu_l)$
        Sample $T_l^{(t)} \mid r_l^{[T]}, \sigma^{(t-1)}, T_l^{(t-1)}$ from Metropolis-Hasting algorithm
        Sample $\mu_l^{(t)} \mid r_l^{[T]}, T_l^{(t)}, \sigma^{(t-1)} \sim N(\Theta_1 \tilde{r}_l^{[T]}, \Theta_1)$
    end for
    Update $r_i \leftarrow y_i - \sum_{l=1}^{L} u_l(x_i, \hat{e}(x_i); T_l^{(t)}, \mu_l^{(t)}) - \sum_{j=1}^{K} v_j(x_i; S_j^{(t-1)}, \omega_j^{(t-1)})$
    Sample $(\sigma^2)^{(t)} \mid \{(T_l^{(t)}, \mu_l^{(t)})\}_{l=1}^{L}, \{(S_j^{(t-1)}, \omega_j^{(t-1)})\}_{j=1}^{K}, y \sim$
      $InvGamma(\frac{\nu+n}{2}, \frac{1}{2}[y_i - \sum_{l=1}^{L} u_l(x_i, \hat{e}(x_i); T_l^{(t)}, \mu_l^{(t)}) - \sum_{j=1}^{K} v_j(x_i; S_j^{(t-1)}, \omega_j^{(t-1)})])$
end for
```

Algorithm 6: Bayesian Causal Forest(BCF)

```
/* Step 3: Posterior draws of treatment term */
for $t = 1$ to $iter.max$ do
    for $j = 1$ to $K$ do
        Set $r_{ji}^{[S]} \leftarrow r_i + v_j(x_i; S_j, \omega_j)$
        Sample $S_j^{(t)} \mid r_j^{[S]}, \sigma^{(t)}, S_j^{(t-1)}$ from Metropolis-Hasting algorithm
        Sample $\omega_j^{(t)} \mid r_j^{[S]}, S_j^{(t)}, \sigma^{(t)} \sim N(\Theta_2 \tilde{r}_j^{[S]}, \Theta_2)$
    end for
    Update $r_i \leftarrow y_i - \sum_{l=1}^{L} u_l(x_i, \hat{e}(x_i); T_l^{(t)}, \mu_l^{(t)}) - \sum_{j=1}^{K} v_j(x_i; S_j^{(t)}, \omega_j^{(t)})$
    Sample $(\sigma^2)^{(t)} \mid \{(T_l^{(t)}, \mu_l^{(t)})\}_{l=1}^{L}, \{(S_j^{(t)}, \omega_j^{(t)})\}_{j=1}^{K}, y \sim$
      $IG(\frac{\nu+n}{2}, \frac{1}{2}[y_i - \sum_{l=1}^{L} u_l(x_i, \hat{e}(x_i); T_l^{(t)}, \mu_l^{(t)}) - \sum_{j=1}^{K} v_j(x_i; S_j^{(t)}, \omega_j^{(t)})])$
end for
```

Due to the modifications discussed above, BCF outperforms BART and other tree-based models in CATE estimations when compared to BART. SATTs are the target estimands of the 2022 ACIC data challenge. Because SATTs are calculated using estimated $\tau(\cdot)$, thus BCF model is more likely to have better predictions for the targeted estimands.

4.4 BCF Implementation

Bayesian causal forest(BCF) consists of two sequential BART models. For our BCF implementations, we use the R package "bcf," which was written by Hahn et al. [7]. Let us recall the expression of the BCF model:

$$
\begin{aligned}
y_i &= h([x_i\ \hat{e}(\tilde{x}_i)]) + \tau(w_i) z_i + \epsilon_i \\
&= \sum_{l=1}^{L} u_l(x_i, \hat{e}(\tilde{x}_i); T_l, \mu_l) + \sum_{k=1}^{K} v_k(x_i; S_k, \omega_k) + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, \sigma^2).
\end{aligned}
\tag{4.19}
$$

Equation 4.19 differs slightly from Definition 22 of the BCF model proposed by Hahn et al., since w_i may differ from x_i in implementations [8]. Moreover, according to Section 3.5.3, the predictor covariates $\tilde{x}_i$ used to estimate propensity score $\hat{e}$ are also different from x_i.

4.4.1 Hyperparameters Tuning

The tuning situation for hyperparameters is similar to that of the BART model. For the BART used to estimate $h(\cdot)$, we simply use the default settings of hyperparameters for the BART model. However, for the BART model used to estimate $\tau(\cdot)$, Caron et al. proposed that it uses a smaller number of trees in the ensemble by setting $K = 50$ [11]. It also prefers smaller tree sizes for each tree by changing the hyperparameters to control tree depth (ν, β) from $(0.95, 2)$ to $(0.25, 3)$. The reasoning for changing the hyperparameter settings of BART to estimate $\tau(\cdot)$ is to improve its ability to detect weak heterogeneous patterns. Similarly to the BART implementations, the first 500 iterations were discarded and the next 1000 iterations were used for posterior inference.

4.4.2 Inference Statistics

Let us assume that the posterior draws of $\tau(w_i)$ to be $\{\hat{\tau}_1(w_i), \hat{\tau}_2(w_i), \dots, \hat{\tau}_{1000}(w_i)\}$, where $\hat{\tau}_k$ is the sum of trees of the latter BART model in one iteration, i.e. as described in Algorithm 6,

$$\hat{\tau}_k(w_i) = \sum_{j=1}^{K} v_j(w_i; S_j^{(0)}, \omega_j^{(0)}), \tag{4.20}$$

where x_i denotes the ith practice in year3 or year4. Hence, the posterior mean $\tilde{\tau}(w_i)$ of $\tau(w_i)$ is given by $\frac{1}{1000}\sum_{t=1}^{1000}\hat{\tau}_t$. Let us denote the posterior mean of $\tau(w_i)$ as $\tilde{\tau}(w_i)$. Like **Method.2** in Section 3.5.2, to get a $(1-\alpha)\%$ credible interval for $\tilde{\tau}(w_i)$ is to calculate the upper $\alpha/2$ and lower $\alpha/2$ quantiles of $\{\hat{\tau}_1(w_i), \hat{\tau}_2(w_i), \dots, \hat{\tau}_{1000}(w_i)\}$.

4.4.3 Posterior Inference for SATTs

In the BCF model, the estimated propensity score is part of the input data. We choose the estimated propensity score modelled by BART using Algorithm 5. Unlike processes in Algorithm 6, we do not need to take treated practices and construct test data to predict $Y_{t,1}(1), Y_{t,i}(0),\ t \in \{3, 4\}$ before running BART. After obtaining the posterior draws of $\tau(\cdot)$, we could directly select the practice id for which $Z_i = 1$. Overall SATT could thus be re-expressed as:

$$SATT_{overall} = \frac{1}{\sum_{t=3}^{4} N_t} \sum_{t=3}^{4} \sum_{i:Z_i=1} n\_patients_{t,i}\tilde{\tau}(w_i). \tag{4.21}$$

Like the implementations in the BART model, we implement the BCF for year 3 and year 4 respectively. The variable selections are nearly identical to those found in BART implementations. The design matrix in R package "bcf" consists of selected covariates except for estimated propensin_patientsty score $\hat{e}(\tilde{x})$ and treatment status Z. In a BCF model, the design matrix for $h(\cdot)$ and $\tau(\cdot)$ must be specified separately. With the same notations of covariates in Section 3.5.4, the design matrix for $h(\cdot)$ denoted as $x\_moderate$ is $(X, Y_1, Y_2, size_1, size_2, diff)$ for year3 and $(X, Y_1, Y_2, Y_3, size_1, size_2, diff)$ for year4. The design matrix for $\tau(\cdot)$ is denoted as $x\_control$ and we set $x\_control = X$, where X denotes the practice-level covariates. The regularization prior for $\tau(\cdot)$ selects a much smaller number of trees and a much smaller tree size, resulting in a much smaller number of splitting rules across the trees. That is why, to estimate $\tau(\cdot)$, we only use a subset of covariates X. Section 4.4.2 describes how to construct confidence intervals for $\tilde{\tau}(w_i),\ i\colon Z_i = 1$.

4.4.4 Summary of BCF Implementation

The process to implement BCF in the R package to estimate SATTs is summarized as follows.

Algorithm 7: BCF to calculate SATTs using R package

```
Input: 3400 datasets and number of trees $K$ and $L$
Output: Estimated $SATT_{overall}$, $SATT_{yearly}$ and $SATT_{subgroup}$ and their corresponding
        credible intervals for each dataset.
Initialize record dataset which is used to save estimated SATTs and the upper and lower bounds of credible intervals Initialize $count = 0$ which remembers the index to write the estimations in the record dataset  for $i = 1$ to $3400$
    Read $i$th dataset Read the record dataset Read the estimated propensity score $\hat{e}$ saved in Construct dataframe $df_{year3}$ which consists of the practice-level covariates $X$, treatment status variable $Z$, estimated propensity score $\hat{e}$, pre-treatment outcomes $Y_1$ and $Y_2$, number of patients in pre-treatment stage $size_1$ and $size_2$, difference of year1 outcome and year2 outcome $diff$ and estimated propensity score $\hat{e}$
    Construct dataframe $df_{year4}$ which consists of the practice-level covariates $X$, treatment status variable $Z$, estimated propensity score $\hat{e}$, pre-treatment outcomes $Y_1$ and $Y_2$, number of patients in pre-treatment stage $size_1$ and $size_2$, difference of year1 outcome and year2 outcome $diff$, year3 outcomes $Y_3$ and estimated propensity score $\hat{e}$
    Convert categorical variables in dataframes $df_{year3}$ and $df_{year4}$ to factors
    Use BayesTree::makeind to dummy-encode factors in dataframes $df_{year3}$ and $df_{year4}$
    Convert dataframes $df_{year3}$, $df_{year4}$ to matrices $X\_moderate_{year3}$, $X\_moderate_{year4}$
    Select the $id.pratice$ of rows in dataframe $df_{year3}$ with $Z = 1$ and save it as $treated\_index$
    Run bcf::bcf ($Y_3$, $Z$, $x\_control_{year3}$, $x\_moderate_{year3} = X$, $\hat{e}$, nburn=500, nsim=1000, ntree$_c ontrol$ =K, $ntree_m oderate$ =L)
    Run bcf::bcf ($Y_4$, $Z$, $x\_control_{year4}$, $x\_moderate_{year4} = X$, $\hat{e}$, nburn=500, nsim=1000, ntree$_c ontrol$ =K, $ntree_m oderate$ =L)
    Initialize 15 numeric vectors each with length 1000 for $satt_{overall}$, $satt_{year3}$, $satt_{year4}$ and 12 subgroup SATTs
```

Algorithm 7: BCF to calculate SATTs using R package

```
    Select posterior draws of $\tau(\cdot)$ from the two BCF models with row index $treated\_index$
      and save them as matrices $\hat{\tau}^{target}_{year3}$, $\hat{\tau}^{target}_{year4}$
    Use matrices $\hat{\tau}^{target}_{year3}$, $\hat{\tau}^{target}_{year4}$ and the number of patients in corresponding practice at
      year 3 $n\_patients_{year3}$ and year 4 $n\_patients_{year4}$ to calculate SATTs in 1000
      posterior draws and save them in the 15 numeric vectors
    Average each of 15 numeric vectors to get the final estimation for $satt_{overall}$, $satt_{year3}$,
      $satt_{year4}$ and 12 subgroup SATTs and save it to the record dataset
    Construct credible intervals for 15 SATTs with 0.05 and 0.95 quantiles of their
      corresponding numeric vectors as lower and upper bounds and save them to the
      record dataset
    Update $count \leftarrow count + 15$
end for
```

5 Double Machine Learning

Machine learning methods were designed for prediction, but standard machine learning methods are biased estimators for treatment effects. Since mean squared error(MSE) equals bias squared plus variance, to minimize MSE, we trade off variance for bias. Another problem is that consistent machine learning methods converge more slowly than $1/\sqrt{n}$ [18]. Furthermore, standard machine learning methods do not provide confidence intervals for the treatment estimates. As a result, using machine learning methods for causal inference is tricky and this is why we need double machine learning for causal inference. The following are the primary goals of the double machine learning model:

- Eliminate the bias,
- Achieve $\sqrt{n}$-consistency,
- Construct valid confidence intervals for estimators [18].

There are two sources of bias in naive estimators from machine learning models. One source of bias is regularization bias. Machine learning algorithms employ regularization to avoid overfitting data with complex functional forms. Regularization reduces estimator variance and prevents overfitting. However, it also introduces bias and causes a slower convergence rate [18]. Another source of bias is overfitting. The model overfits when it models the idiosyncrasies of the particular sample too closely [18]. This results in poor out-of-sample performance and means that the model is unable to generalize well to new data. Sometimes the efforts to regularize fail to prevent overfitting.

Chernozhukov et al. first proposed the double or debiased ML (DML) methods that make use of Neyman orthogonality and sample-splitting [14]. Double machine learning overcomes these two sources of bias in naive estimators from machine learning models. It corrects bias caused by regularization using Neyman orthogonality and bias caused by overfitting using sample-splitting.

5.1 Neyman Orthogonality

Definition 28 ⟨Nuisance parameter⟩

A nuisance parameter is any unspecified parameter that must be accounted for when hypothesis testing for the parameters of interest.

Definition 29 ⟨Identification condition equation [13]⟩

Let us assume that W is the observation data, θ is the targeted parameter, θ_0 is the true value of θ, η is the nuisance parameter, η_0 is the true value of η [15]. The Identification condition equation for score function ψ is:

$$\mathbb{E}[\psi(W;\theta_0,\eta_0)] = 0. \tag{5.1}$$

Definition 30 ⟨Neyman orthogonality [13]⟩

Let us assume that W is the observation data, θ is the targeted parameter, θ_0 is the true value of θ, η is the nuisance parameter, η_0 is the true value of η, and $\psi(\cdot)$ is the score function [15]. Then the score function $\psi(\cdot)$ obeys Neyman orthogonality if the Gateaux derivative of score function $\psi(\cdot)$ regarding the nuisance parameter η is 0 *[13, 19]. The mathematical expression is as follows:*

$$\begin{aligned}\partial_\eta \, \mathbb{E}[\psi(W;\theta_0,\eta_0)][\eta-\eta_0] &\coloneqq \frac{d}{dr}\,\mathbb{E}[\psi(W;\theta_0,\eta_0+r(\eta-\eta_0))] \mid_{r=0} \\ &= 0.\end{aligned} \tag{5.2}$$

5.2 Sample-Splitting

Assuming that in total we have N observations, we divide the sample into K folds, each of size $n = N/K$. For simplicity, let us assume N/K is an integer. Let us denote the kth fold by I_k and its complement by I_k^c, for $k \in \{1, 2, \cdots, K\}$.Then, for each fold, we build a machine learning estimator $\hat{\beta}_k$ and use the observations in I_k^c to get the fit and estimate the parameters. Cross-fitting is based on sample splitting and was emphasized by Chernozhukov et al. to allow for broader use of machine learning models to estimate the nuisance parameters [15, 13, 14].

Definition 31 ⟨Cross-fitting [13]⟩

Assume that after sample splitting, we have K partitions of N observations, i.e. $\{(I_1, I_1^c), (I_2, I_2^c), \cdots, (I_K, I_K^c)\}$, and η_0 is true value of nuisance parameter, θ_0 is the true value for the target param-

eter [15]. For $k \in \{1, 2, \cdots, K\}$, $\hat{\eta}_0^k$ is the machine learning estimator for η_0 based on observations in I_k^c is defined as below:

$$\hat{\eta}_0^k(I_k^c) = \hat{\eta}_0^k((W_i)_{i \notin I_c}). \tag{5.3}$$

Then, we use $\hat{\eta}_0^k$ for the complementary set I_k and get K estimators of θ_0, which is defined by $\check{\theta}_0^k(W; \hat{\eta}_0^k(I_k^c))$, and finally we average the K estimators to get $\tilde{\theta}_0$:

$$\tilde{\theta}_0 := \frac{1}{K} \sum_{k=1}^{K} \check{\theta}_0^k(W; \hat{\eta}_0^k(I_k^c)). \tag{5.4}$$

During the cross-fitting process, the most important thing to notice is that the estimator we use for fold k was fit in I_k^c. Let us now consider the identification condition $\mathbb{E}[\psi(W; \theta_0, \eta_0)] = 0$ and the partition(I_k, I_k^c) [15, 13, 14]. To get $\check{\theta}_0^k(W; \hat{\eta}_0^k(I_k^c))$ is to solve the following equation which represents the identification condition:

$$\begin{aligned} \mathbb{E}[\psi(W; \check{\theta}_0^k, \hat{\eta}_0^k(I_k^c))] &= \frac{1}{n} \sum_{i \in I_k} \psi(W_i; \check{\theta}_0^k, \hat{\eta}_0^k(I_k^c)). \\ &= 0 \end{aligned} \tag{5.5}$$

Definition 32 ⟨A variant of Cross-fitting [13]⟩

Using the same notations as in Definition 31, we could construct another estimator $\bar{\theta}_0$, which is the solution of the following equation [15, 13, 14]:

$$\begin{aligned} \frac{1}{K} \sum_{k=1}^{K} \mathbb{E}[\psi(W; \bar{\theta}_0, \hat{\eta}_0^k(I_k^c))] &= \frac{1}{N} \sum_{k=1}^{K} \sum_{i \in I_k} \psi(W_i; \bar{\theta}_0, \hat{\eta}_0^k(I_k^c)) \\ &= 0. \end{aligned} \tag{5.6}$$

5.3 Model

In terms of double machine learning, there are two classical models for estimating causal effects.

5.3.1 Partially Linear Regression Model

Partially linear regression model is defined as follows [15, 13, 14, 21]:

$$Y = Z\theta_0 + g_0(X) + \zeta, \qquad \mathbb{E}[\zeta \mid Z, X] = 0, \tag{5.7}$$

$$Z = m_0(X) + \xi, \qquad \mathbb{E}[\xi \mid X] = 0, \tag{5.8}$$

where Y is the outcome variable, Z is the binary treatment variable, X consists of other predictor covariates, and ζ and ξ are stochastic errors [21].

The following is the most common procedure for performing double machine learning to estimate θ_0:

Procedure 5.1 Estimate θ_0 via double machine learning

1. Use a machine learning model $\hat{g}_0$ with covariates X to estimate y and set estimated residuals $\hat{W} = Y - \hat{g}_0(X)$,
2. Use a machine learning model $\hat{m}_0$ with covariates X to estimate y and set estimated residuals $\hat{V} = Z - \hat{m}_0(X)$,
3. Estimate θ_0 by regressing the residual $\hat{W}$ on $\hat{V}$ and get the estimation $\hat{\theta}_0$ [21].

Definition 33

The following is the score function for estimating ATE using the Partially Linear Regression model [15, 13, 14]:

$$\psi(W; \theta, \eta) := (Y - g(X) - \theta(Z - m(X)))(Z - m(X)), \tag{5.9}$$

where $W := (X, Y, Z)$, the nuisance parameter $\eta := (g, m)$, and the true value for the nuisance parameter is $\eta_0 := (g_0, m_0)$ [15]. Moreover,

$$\mathbb{E}[Y \mid X] = g_0(X), \tag{5.10}$$

$$\mathbb{E}[Z \mid X] = m_0(X). \tag{5.11}$$

Theorem 4

Performing Procedure 5.2 to estimate θ_0 in the Partially Linear Regression model is equivalent to using the score function in Definition 33 to solve identification condition equation without sample-splitting to estimate θ_0.

Proof. Using the estimation of coefficients in a simple linear regression model, for Procedure 5.2, we get:

$$\hat{\theta}_0 = (\frac{1}{n}\sum_{i=1}^{n}\hat{V}_i^2)^{-1}\frac{1}{n}\sum_{i=1}^{n}\hat{V}_i\hat{W}_i, \tag{5.12}$$

where n is the number of observations. Next, let us consider the identification condition $\mathbb{E}[\psi(W;\theta_0,\hat{\eta}_0)] = 0$ with score function in Definition 33,

$$\begin{aligned}\mathbb{E}[\psi(W;\tilde{\theta}_0,\hat{\eta}_0)] =& \frac{1}{n}\sum_{i=1}^{n}[(y_i - \hat{g}_0(x_i) - \tilde{\theta}_0(Z_i - \hat{m}_0(x_i)))(Z_i - \hat{m}_0(x_i))] \\ =& \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{g}_0(x_i))(Z_i - \hat{m}_0(x_i)) - \tilde{\theta}_0\frac{1}{n}\sum_{i=1}^{n}(Z_i - \hat{m}_0(x_i))^2,\end{aligned} \tag{5.13}$$

where, $\hat{\eta}$ is the estimation of true nuisance parameter via machine learning methods, $\tilde{\theta}_0$ is an estimator of θ_0. Thus,

$$\mathbb{E}[\psi(W;\tilde{\theta}_0,\hat{\eta}_0)] = 0, \tag{5.14}$$

$$\Longleftrightarrow \tilde{\theta}_0 = [\frac{1}{n}\sum_{i=1}^{n}(Z_i - \hat{m}_0(x_i))^2]^{-1}\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{g}_0(x_i))(Z_i - \hat{m}_0(x_i)). \tag{5.15}$$

Assuming that we use the same machine learning methods to obtain $\hat{g}_0$ and $\hat{m}_0$,

$$\hat{\theta}_0 = \tilde{\theta}_0. \tag{5.16}$$

□

Corollary 2

The score function $\psi(W;\theta,\eta) = (Y - g(X) - \theta(Z - m(X)))(Z - m(X))$ [15, 13, 14, 21] used to estimate ATT satisfies Neyman orthogonality.

Proof. According to Definition 30, we need to verify the following equation:

$$\frac{d}{dr}\mathbb{E}[\psi(W;\theta_0,\eta_0 + r(\eta - \eta_0))] \mid_{r=0} = 0. \tag{5.17}$$

In the score function in Definition 33, $\eta = (g, m)$, let δ_g, δ_m be test functions respectively perturbing g, m, u. This means that $\eta - \eta_0 = (\delta_g, \delta_m)$

Thus,

$$\begin{aligned}
\text{LHS of Equation 5.17} &= \frac{d}{dr}\,\mathbb{E}[\psi(W;\theta_0,\eta_0+r(\delta_g,\delta_m))]\mid_{r=0}\\
&= \frac{d}{dr}\,\mathbb{E}[(Y-g_0(X)-r\delta_g-\theta_0(Z-m_0(X)-r\delta_m))(Z-m_0(X)-r\delta_m)]\mid_{r=0}\\
&= \frac{d}{dr}\,\mathbb{E}[(Y-g_0(X)-r\delta_g)(Z-m_0(X)-r\delta_m)]\mid_{r=0} -\theta_0\\
&\quad \times \frac{d}{dr}\,\mathbb{E}[(Z-m_0(X)-r\delta_m)^2]\mid_{r=0}.
\end{aligned}$$

Differentiating under the expectation sign due to dominated converge theorem, yields:

$$\begin{aligned}
&= \mathbb{E}[\frac{d}{dr}(Y-g_0(X)-r\delta_g)(Z-m_0(X)-r\delta_m)]\mid_{r=0} -\theta_0\\
&\quad \times \mathbb{E}[\frac{d}{dr}(Z-m_0(X)-r\delta_m)^2]\mid_{r=0}\\
&= \mathbb{E}[-(Y-g_0(X))\delta_m-(Z-m_0(X))\delta_g]\mid_{r=0} + \mathbb{E}[2r\delta_m\delta_g]\mid_{r=0}\\
&\quad -\theta_0\,\mathbb{E}[-2(Z-m_0(X)-r\delta_m)\delta_m]\mid_{r=0}\\
&= -\,\mathbb{E}[(Y-g_0(X))\delta_m+(Z-m_0(X))\delta_g]+2\theta_0\,\mathbb{E}[(Z-m_0(X))\delta_m].
\end{aligned}$$

Applying Law of Total Expectation and Pulling out known factors of conditional expectation, yields:

$$\begin{aligned}
&= \mathbb{E}[\mathbb{E}[(Y-g_0(X))\delta_m(X)\mid X]]+\mathbb{E}[\mathbb{E}[(Z-m_0(X))\delta_g(X)\mid X]]\\
&\quad -\theta_0\,\mathbb{E}[\mathbb{E}[(Z-m_0(X))\delta_m(X)\mid X]]\\
&= \mathbb{E}[\mathbb{E}[(Y-g_0(X))\mid X]\delta_m(X)]+\mathbb{E}[\mathbb{E}[(Z-m_0(X))\mid X]\delta_g(X)]\\
&\quad -\theta_0\,\mathbb{E}[\mathbb{E}[(Z-m_0(X))\mid X]\delta_m(X)]\\
&= 0.
\end{aligned} \tag{5.18}$$

The last equation holds due to the following facts:

$$\mathbb{E}[Y\mid X]=g_0(X), \tag{5.19}$$

$$\mathbb{E}[Z\mid X]=m_0(X). \tag{5.20}$$

□

We do not use the Partially Linear Regression model to estimate ATT because it does not allow it. Chernozhukov et al. demonstrated that the Interactive Regression Model is the best choice for estimating treatment effects through double machine learning [14].

5.3.2 Interactive Regression Model

Interactive regression models have the following form [15, 13, 14]:

$$Y = g_0(Z, X) + \zeta \quad \mathbb{E}[\zeta \mid Z, X] = 0, \tag{5.21}$$

$$Z = m_0(X) + \xi \quad \mathbb{E}[\xi \mid X] = 0., \tag{5.22}$$

where Y is the outcome variable, Z is the binary treatment variable, X consists of other predictor covariates, and ζ and ξ are stochastic errors [13]. The visualization of causal relationships is shown in Figure 5.1.

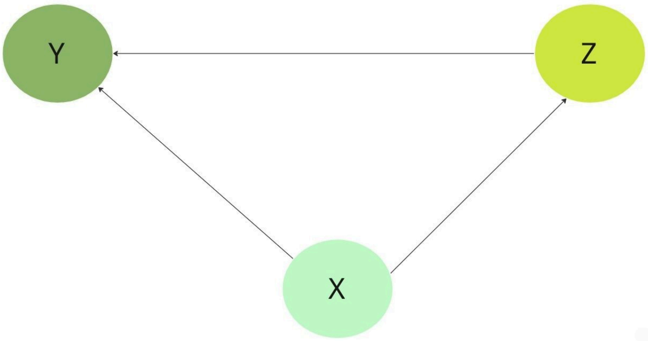

Figure 5.1 Visualization of causal relationships in Interactive Regression Model

The Interactive Regression Model is the model that we use in implementations of double machine learning. Our target estimand is the average treatment effect for the treated (ATT). Chernozhukov et al. proposed the score function in an interactive regression model to estimate ATT for the first time [14].

Definition 34

The score function to estimate ATT using the Interactive Regression model is given below [15, 13, 14]:

$$\psi(W; \theta, \eta) := \frac{Z(Y - g(0, X))}{u} - \frac{m(W)(1 - Z)(y - g(0, X))}{u(1 - m(X))} - \theta \frac{Z}{u}, \tag{5.23}$$

where $W := (X, Y, Z)$, the nuisance parameter $\eta := (g, m, u)$, and the true value for nuisance parameter is $\eta_0 := (g_0, m_0, u_0)$ [15]. Moreover,

$$u_0 := \mathbb{E}[Z]. \tag{5.24}$$

Due to the definition of the Interactive Regression Model, we have:

$$\mathbb{E}[Y \mid Z, X] = g_0(Z, X), \tag{5.25}$$

$$\mathbb{E}[Y \mid Z = 0, X] = g_0(0, X), \tag{5.26}$$

$$\mathbb{E}[Z \mid X] = m_0(X). \tag{5.27}$$

Definition 35 ⟨Linear score function⟩

The score function $\psi(W; \theta, \eta)$ is called linear if it has the form [13]:

$$\psi(W; \theta, \eta) = \psi_a(W; \eta)\theta + \psi_b(W; \eta). \tag{5.28}$$

It is obvious that the score functions in Definition 33 and Definition 34 are linear score functions. Furthermore, in the following corollary, we will demonstrate that the score function used to estimate ATT satisfies Neyman orthogonality.

Corollary 3

The score function $\psi(W; \theta, \eta) = \frac{Z(Y-g(0,X))}{u} - \frac{m(X)(1-Z)(Y-g(0,X))}{u(1-m(X))} - \theta\frac{Z}{u}$ [15, 13, 14] used to estimate ATT satisfies Neyman orthogonality.

Proof. According to Definition 30, we need to verify the following equation:

$$\frac{d}{dr}\,\mathbb{E}[\psi(W; \theta_0, \eta_0 + r(\eta - \eta_0))] \mid_{r=0} = 0. \tag{5.29}$$

In the score function in Definition 34, $\eta = (g, m, u)$, let $\delta_g, \delta_m, \delta_u$ be test functions respectively perturbing g, m, u. This means that $\eta - \eta_0 = (\delta_g, \delta_m, \delta_u)$ Thus,

$$\begin{aligned}
\text{LHS of Equation 5.29} &= \frac{d}{dr}\mathbb{E}[\psi(W;\theta_0,\eta_0 + r(\delta_g,\delta_m,\delta_u))]\mid_{r=0} \\
&= \frac{d}{dr}\mathbb{E}[\frac{Z(Y-g_0(0,X)-r\delta_g(0,X))}{u_0+r\delta_u} \\
&\quad - \frac{(m_0(X)+r\delta_m(X))(1-Z)(Y-g_0(0,X)-r\delta_g(0,X))}{(u_0+r\delta_u)(1-m_0(X)-r\delta_m(X))} - \theta_0\frac{Z}{u_0+r\delta_u}]\mid_{r=0} \\
&= \frac{d}{dr}\mathbb{E}[\frac{Y-g_0(0,X)-r\delta_g(0,X)}{u_0+r\delta_u}\times\frac{Z-m_0(X)-r\delta_m}{1-m_0(X)-r\delta_m}]\mid_{r=0} -\theta_0\times\frac{d}{dr}\mathbb{E}[\frac{Z}{u_0+r\delta_u}]
\end{aligned}$$

Differentiating under the expectation sign due to dominated converge theorem, yields:

$$\begin{aligned}
&= \mathbb{E}[\frac{d}{dr}(\frac{Y-g_0(0,X)-r\delta_g(0,X)}{u_0+r\delta_u}\times\frac{Z-m_0(X)-r\delta_m}{1-m_0(X)-r\delta_m})]\mid_{r=0} \\
&\quad -\theta_0\times\mathbb{E}[\frac{d}{dr}(\frac{Z}{u_0+r\delta_u})]\mid_{r=0} \\
&= \mathbb{E}[\frac{Y-g_0(0,X)-r\delta_g(0,X)}{u_0+r\delta_u} \\
&\quad\times\frac{-\delta_m(1-m_0(X)-r\delta_m)+(Z-m_0(X)-r\delta_m)\delta_m}{(1-m_0(X)-r\delta_m)^2}]\mid_{r=0} + \mathbb{E}[\frac{Z-m_0(X)-r\delta_m}{1-m_0(X)-r\delta_m} \\
&\quad\times\frac{-\delta_g(0,X)(u_0+r\delta_u)-(Y-g_0(0,X)-r\delta_g)\delta_u}{(u_0+r\delta_u)^2}]\mid_{r=0} -\theta_0\times\mathbb{E}[\frac{-Z\delta_u}{(u_0+r\delta_u)^2}]\mid_{r=0} \\
&= \mathbb{E}[\frac{(Y-g_0(0,X))\delta_m(Z-1)}{u_0(1-m_0(X))^2}] \\
&\quad+\mathbb{E}[\frac{(Z-m_0(X))[-u_0\delta_g(0,X)-(Y-g_0(0,X))\delta_u]}{(1-m_0(X))u_0^2}] - \theta_0\times\mathbb{E}[-\frac{Z\delta_u}{u_0^2}].
\end{aligned}$$

Taking the equation $\theta_0 = \frac{u_0}{Z}\times[\frac{Z(Y-g(0,X))}{u} - \frac{m(X)(1-Z)(Y-g(0,X))}{u(1-m(X))}]$ into account, yields:

$$\begin{aligned}
&= \mathbb{E}[\frac{(Y-g_0(0,X))\delta_m(Z-1)}{u_0(1-m_0(X))^2}] \\
&\quad+\mathbb{E}[\frac{(Z-m_0(X))[-u_0\delta_g(0,X)-(Y-g_0(0,X))\delta_u]}{(1-m_0(X))u_0^2}] - \frac{u_0}{Z} \\
&\quad\times[\frac{Z(Y-g(0,X))}{u_0} - \frac{m(X)(1-Z)(Y-g(0,X))}{u_0(1-m(X))}]\times\mathbb{E}[-\frac{Z\delta_u}{u_0^2}] \\
&= \mathbb{E}[\frac{(Y-g_0(0,X))\delta_m(Z-1)}{u_0(1-m_0(X))^2}] \\
&\quad+\mathbb{E}[\frac{(Z-m_0(X))[-u_0\delta_g(0,X)-(Y-g_0(0,X))\delta_u]}{(1-m_0(X))u_0^2}] \\
&\quad+\mathbb{E}[\frac{(Y-g_0(0,X))(Z-m_0(X))\delta_u}{u_0^2(1-m_0(X))}].
\end{aligned}$$

Applying Law of total expectation, yields:

$$\begin{aligned}
\text{LHS of Equation 5.29} &= \mathbb{E}[\frac{(Y-g_0(0,X))\delta_m(Z-1)}{u_0(1-m_0(X))^2} \mid Z=0] \times \mathbb{P}(Z=0) \\
&+ \mathbb{E}[\frac{(Y-g_0(0,X))\delta_m(Z-1)}{u_0(1-m_0(X))^2} \mid Z=1] \times \mathbb{P}(Z=1) \\
&+ \mathbb{E}[\frac{(Z-m_0(X))[-u_0\delta_g(0,X)-(Y-g_0(0,X))\delta_u]}{(1-m_0(X))u_0^2}] \\
&+ \mathbb{E}[\frac{(Y-g_0(0,X))(Z-m_0(X))\delta_u}{u_0^2(1-m_0(X))}] \\
&= \mathbb{E}[-\frac{(Y-g_0(0,X))\delta_m}{u_0(1-m_0(X))^2} \mid Z=0] \times \mathbb{P}(Z=0) \\
&+ \mathbb{E}[\frac{(Z-m_0(X))[-u_0\delta_g(0,X)-(Y-g_0(0,X))\delta_u]}{(1-m_0(X))u_0^2}]] \\
&+ \mathbb{E}[\frac{(Y-g_0(0,X))(Z-m_0(X))\delta_u}{u_0^2(1-m_0(X))}].
\end{aligned}$$

Applying tower property of conditional expectation, yields:

$$\begin{aligned}
&= \mathbb{E}[\mathbb{E}[-\frac{(Y-g_0(0,X))\delta_m}{u_0(1-m_0(X))^2} \mid X, Z] \mid Z=0] \times \mathbb{P}(Z=0) \\
&+ \mathbb{E}[\mathbb{E}[\frac{(Z-m_0(X))[-u_0\delta_g(0,X)-(Y-g_0(0,X))\delta_u]}{(1-m_0(X))u_0^2} \mid X]] \\
&+ \mathbb{E}[\mathbb{E}[\frac{(Y-g_0(0,X))(Z-m_0(X))\delta_u}{u_0^2(1-m_0(X))} \mid X]].
\end{aligned}$$

Considering Equations 5.21 and 5.22, yields:

$$\begin{aligned}
&= \mathbb{E}[\mathbb{E}[-\frac{(g_0(Z,X)+\zeta-g_0(0,X))\delta_m}{u_0(1-m_0(X))^2} \mid X, Z] \mid Z=0] \times \mathbb{P}(Z=0) \\
&+ \mathbb{E}[\mathbb{E}[\frac{\xi[-u_0\delta_g(0,X)-(Y-g_0(0,X))\delta_u]}{(1-m_0(X))u_0^2} \mid X]] + \mathbb{E}[\mathbb{E}[\frac{(Y-g_0(0,X))\xi\delta_u}{u_0^2(1-m_0(X))} \mid X]]
\end{aligned}$$

Combing the last two terms and cancelling out two conditional expectations regarding $Y - g_0(0,X)$, yields:

$$\begin{aligned}
&= \mathbb{E}[\mathbb{E}[-\frac{(g_0(Z,X)+\zeta-g_0(0,X))\delta_m}{u_0(1-m_0(X))^2} \mid X, Z] \mid Z=0] \times \mathbb{P}(Z=0) \\
&+ \mathbb{E}[\mathbb{E}[\frac{-\xi u_0\delta_g(0,X)}{(1-m_0(X))u_0^2} \mid X]] \\
&= \mathbb{E}[\mathbb{E}[-\zeta \mid X, Z]\frac{\delta_m(X)}{u_0(1-m_0(X))^2} \mid Z=0] \times \mathbb{P}(Z=0) \\
&+ \mathbb{E}[\mathbb{E}[\xi \mid X] \times \mathbb{E}[\frac{-u_0\delta_g(0,X)}{(1-m_0(X))u_0^2} \mid X]].
\end{aligned}$$

Thus, due to the definition of the Interactive Regression Model, we have:

$$\text{LHS of Equation 5.29} = 0.$$

□

5.4 Double Machine Learning Algorithm

Definition 31 and Definition 32 refer to cross-fitting and a variant of cross-fitting respectively. These two methods, along with the score functions, provide the intuitions for the algorithms described below, which were first proposed by Chernozhukov et al. [14]. I_k denotes the kth fold while (I_k, I_k^c) forms a partition of the dataset. To estimate ATT, we use the Interactive Regression model and the score function ψ in Definition 34. To implement double machine learning in the Interactive Regression model, the following two procedures explain two ways that correspond to Definition 31 and Definition 32 respectively.

Procedure 5.2 Double machine learning estimation and inference on ATT via Interactive Regression Model and cross-fitting [14]

Step.1 Create K partitions $\{(I_1, I_1^c), (I_2, I_2^c), \cdots, (I_K, I_K^c)\}$ from data, each of size $n := N/K$.

Step.2 Construct K estimators for ATT based on K partitions respectively:

- for $k = 1$ to K do

  ◇ Estimate the true value of nuisance parameter $\eta_0 = (g_0, m_0, u_0)$:

  A. Select two machine learning models to fit $g_0(Z, X), m_0(X)$ based on I_k^c respectively and get machine learning estimators $\hat{g}_0^k(I_k^c), \hat{m}_0^k(I_k^c)$.

  B. Calculate $\hat{u}_0^k$ as:

$$\hat{u}_0^k = \mathbb{E}[Z_i = 1 \mid i \in I_k^c] = \frac{1}{N-n} \sum_{i \in I_k^c} Z_i. \tag{5.30}$$

  ◇ Taking machine learning estimators $\hat{\eta}_0^k = (\hat{g}_0^k(I_k^c), \hat{m}_0^k(I_k^c), \hat{u}_0^k)$ into the Equation 5.5 with $W = I_k^c$. $\check{\theta}_0^k$ is the root of the following equation:

$$\frac{1}{n} \sum_{i \in I_k} \psi(W; \check{\theta}_0^k, \hat{\eta}_0^k(I_k^c)) = 0. \tag{5.31}$$

■ end for

Step.3 Aggregate K estimators to get final estimator $\tilde{\theta}_0$:

$$\tilde{\theta}_0 := \frac{1}{K}\sum_{k=1}^{K}\check{\theta}_0^k. \tag{5.32}$$

Procedure 5.3 Another algorithm for Double machine learning estimation and inference on ATT in the interactive regression model and a variant of cross-fitting [14]

Step.1 Create K partitions $\{(I_1, I_1^c), (I_2, I_2^c), \cdots, (I_K, I_K^c)\}$ from data, each of size $n := N/K$.

Step.2 Construct K estimators for ATT based on K partitions respectively:

■ for $k = 1$ to K do

◇ Estimate the true value of nuisance parameter $\eta_0 = (g_0, m_0, u_0)$:

A. Select two machine learning models to fit $g_0(Z, X), m_0(X)$ based on I_k^c respectively and get machine learning estimators $\hat{g}_0^k(I_k^c), \hat{m}_0^k(I_k^c)$.

B. Calculate $\hat{u}_0^k$ as:

$$\hat{u}_0^k = \mathbb{E}[Z_i = 1 \mid i \in I_k^c] = \frac{1}{N-n}\sum_{i\in I_k^c} Z_i. \tag{5.33}$$

◇ Take machine learning estimators $\hat{\eta}_0^k = (\hat{g}_0^k(I_k^c), \hat{m}_0^k(I_k^c), \hat{u}_0^k)$ into LHS of Equation 5.6 with $W = I_k$, get the linear form $\bar{\psi}_a^k(W; \hat{\eta}_0^k)\theta + \bar{\psi}_b^k(W; \hat{\eta}_0^k)$ and record the coefficient $\bar{\psi}_a^k(W; \hat{\eta}_0^k), \bar{\psi}_b^k(W; \hat{\eta}_0^k)$ for future use with the following formulas:

$$\bar{\psi}_a^k(W_i; \hat{\eta}_0^k) := \frac{1}{n}\sum_{i\in I_k}\psi_a(W; \hat{\eta}_0^k), \tag{5.34}$$

$$\bar{\psi}_b^k(W_i; \hat{\eta}_0^k) := \frac{1}{n}\sum_{i\in I_k}\psi_b(W; \hat{\eta}_0^k). \tag{5.35}$$

■ end for

Step.3 The final estimator $\tilde{\theta}_0$ is the root of the following equation:

$$\frac{1}{K}\sum_{k=1}^{K}[\bar{\psi}_a^k(W;\hat{\eta}_0^k)\times\tilde{\theta}_0+\bar{\psi}_b^k(W;\hat{\eta}_0^k)]=0. \tag{5.36}$$

Chernozhukov et al. introduced the theorem below, which provides the theoretical foundation for double machine learning algorithms to estimate ATT [12]. Meanwhile, the theorem demonstrates how to construct confidence intervals.

Theorem 5

Assume that the ATT, $\theta_0=\mathbb{E}[g_0(1,Z)-g_0(0,Z)\mid Z=1]$, is the target parameter and we use the estimator $\tilde{\theta}_0$ of θ_0, which is true value of nuisance parameter $\eta_0=(g_0,m_0,u_0)$ [15]. Moreover, we define σ^2, $\hat{J}_0$, $\hat{\sigma}^2$ as below [13]:

$$\sigma^2 := \mathbb{E}[\psi^2(W;\theta_0,\eta_0)], \tag{5.37}$$

$$\hat{J}_0 := \frac{1}{N}\sum_{k=1}^{K}\sum_{i\in I_k}\psi_a(W_i;\hat{\eta}_0^k), \tag{5.38}$$

$$\hat{\sigma}^2 := \hat{J}_0^{-2}\frac{1}{N}\sum_{k=1}^{K}\sum_{i\in I_k}[\psi(W_i;\tilde{\theta}_0,\hat{\eta}_0^k)]^2. \tag{5.39}$$

Then the estimator $\tilde{\theta}_0$ concentrates around θ_0 at a rate of $1/\sqrt{N}$.It is approximately unbiased and normally distributed under regularity conditions [14]:

$$\sigma^{-1}\sqrt{N}(\tilde{\theta}_0-\theta_0)\rightsquigarrow N(0,1), \tag{5.40}$$

and the result continues to hold if σ^2 is replaced by $\hat{\sigma}^2$. Furthermore, the confidence interval CI_0 based on $\tilde{\theta}_0$ has the following asymptotic property [14]:

$$\mathbb{P}(\theta_0\in CI_0)\rightarrow(1-\alpha), \tag{5.41}$$

where $CI_0 := [\tilde{\theta}_0\pm\Phi^{-1}(1-\alpha/2)\hat{\sigma}/\sqrt{N}]$ [15]. Thus, CI_0 forms an approximate $(1-\alpha)$ confidence interval.

Theorem 5 gives us the theoretical foundation for performing variance estimation and constructing uncertainty intervals. The approach below demonstrates how to apply Theorem 5 for variance estimation and confidence interval construction.

Procedure 5.4 Estimate variance and construct confidence intervals using the Inputs and outputs from Procedure 5.2 or Procedure 5.3

Step.1 Variance estimation:

◇ Compute $\hat{J}_0 = \frac{1}{N}\sum_{k=1}^{K}\sum_{i\in I_k}\psi_a(W_i;\hat{\eta}_0^k)$

◇ Compute the asymptotic variance of $\tilde{\theta}_0$ by:

$$\hat{\sigma}^2 = \hat{J}_0^{-2}\frac{1}{N}\sum_{k=1}^{K}\sum_{i\in I_k}[\psi(W_i;\tilde{\theta}_0,\hat{\eta}_0^k)]^2. \tag{5.42}$$

Step.2 Construction of approximate confidence intervals:

◇ Set significance level = $1-\alpha$

◇ Compute $\tilde{\theta}_0 - \Phi^{-1}(1-\alpha/2)\hat{\sigma}/\sqrt{N}$ as lower bound of the approximate confidence interval

◇ Compute $\tilde{\theta}_0 + \Phi^{-1}(1-\alpha/2)\hat{\sigma}/\sqrt{N}$ as upper bound of the approximate confidence interval

5.5 Modifications of Model in Section 5.4

In the previous section, we used the following model to estimate SATTs:

$$\begin{aligned} &Y_t = g_{0,t}(Z,X_t)+\zeta_t, && \mathbb{E}[\zeta_t \mid Z,X_t]=0, \quad t\in\{year3, year4\},\\ &Z = m_0(M)+\xi, && \mathbb{E}[\xi \mid M]=0, \end{aligned} \tag{5.43}$$

where Y is the outcome variable, Z is the binary treatment variable, X_t is a collection of covariates used to estimate Y_t, M is a collection of covariates used to estimate Z, and ζ and ξ are stochastic errors.

In the process of estimating $SATT_{overall}$ in Section 5.4, we used variable $id.practice$ to create folds and treated year3 data and year4 data with the same $id.practice$ in the same fold. However, the process for building machine learning estimators $\hat{\eta}_0^k(I_k^c) = (\hat{g}_0^k(I_k^c), \hat{m}_0^k(I_k^c), \hat{u}_0^k(I_k^c))$ could be modified. $\hat{m}_0^k(I_k^c)$ is modelled from the pre-treatment stage data. Given a $id.practice$ value, it outputs the same value for both year3 input and year4 input. $\hat{u}_0^k(I_k^c)$ is a constant that is computed across year3 data and year4 data.

$$\begin{aligned}\hat{u}_0^k(I_k^c) &= \mathbb{E}[Z_i = 1 \mid i \in I_k^c] \\ &= \frac{1}{\sum_{i \in I_k^c}(n\_patients_{year3}[i] + n\_patients_{year4}[i])} \\ &\quad \times \sum_{i \in I_k^c} Z_i \times (n\_patients_{year3}[i] + n\_patients_{year4}[i]).\end{aligned} \tag{5.44}$$

The modification in the process for estimating $SATT_{overall}$ keeps machine learning estimators $\hat{m}_0^k(I_k^c), \hat{u}_0^k(I_k^c)$ the same as described in Section 5.4. However, in Section 5.4, we used $\hat{g}_0^{k,3}(I_k^c)$, $\hat{g}_0^{k,4}(I_k^c)$ to predict the year3 outcome Y_3, year4 outcome Y_4 separately for each partition (I_k, I_k^c). Since we treat the year3 data and the year4 data with the same $id.practice$ value as integrals to create folds, it is natural to think about using year as an additional covariate in the baseline model. Following dummy-encoding, $year = 0$ denotes year3 and $year = 1$ denotes year4. After modification, the model to estimate $SATT_{overall}$ changes to the following form :

$$\begin{aligned} &Y_t = g_0(Z, X_t, t) + \zeta_t \quad \mathbb{E}[\zeta_t \mid Z, X_t, t] = 0, \quad t \in \{0, 1\}, \\ &Z = m_0(M) + \xi \qquad\qquad\quad \mathbb{E}[\xi \mid M] = 0, \end{aligned} \tag{5.45}$$

where Y is the outcome variable, Z is the binary treatment variable, X_t consists of covariates to estimate Y_t, M consists of covariates to estimate Z, and ζ and ξ are stochastic errors.

Different from notations in previous models, Y_0 denotes the year3 outcome and Y_1 denotes the year4 outcome in Model 5.45. We also implement Model 5.45 to estimate $SATT_{overall}$ and the results will be shown in the following chapter.

5.6 DML Implementation

5.6.1 The Processes to Estimate $SATT_{overall}$, $SATT_{yearly}$ and $SATT_{subgroup}$

We use Interactive Regression Model and score function in Definition 34 to estimate SATTs. Like the implementations of BART and BCF models in Section 3.5 and Section 4.4, we deal with ATT in year 3 and year 4 respectively. The model is stated as follows:

$$\begin{aligned} Y_t &= g_0^t(Z, X_t) + \zeta_t, \quad \mathbb{E}[\zeta_t \mid Z, X_t] = 0, \quad t \in \{year3, year4\}, \\ Z &= m_0(M) + \xi \qquad\qquad \mathbb{E}[\xi \mid M] = 0 \end{aligned} \tag{5.46}$$

where Y is the outcome variable, Z is the binary treatment variable, X_t consists of covariates to estimate Y_t, M consists of covariates to estimate Z, and ζ and ξ are stochastic errors.

The Model 5.46 is slightly different from the Interactive Regression model in Section 5.3.2. The covariates used to estimate Z may be different from covariates used to estimate Y_t. This is the primary reason that we build double machine learning from scratch rather than using existing R packages. R includes a well-developed package called "dml" that incorporates many double machine learning models. However, the same covariates are required to estimate $g_0(\cdot)$ and $m_0(\cdot)$.

Let us assume that the K-fold sample-splitting is identified as $\{(I_{1,t}, I_{1,t}^c), (I_{2,t}, I_{2,t}^c), \cdots, (I_{K,t}, I_{K,t}^c)\}$ for $t \in \{year3, year4\}$ and consider the estimation of $SATT_{yearly}$. The baseline machine learning models are flexible and we just choose BART models. Since Z is a binary variable, an estimator for $m_0(\cdot)$ is the estimated propensity score. We obtained the estimation $\hat{e}$ in Section 3.4.3 3.5.3 by treating the entire dataset as training data in the BART model. However, due to cross-fitting during the estimation of $SATT_{yearly}$, we implement BART K times to estimate propensity score with different training data, where K is the number of folds in sample-splitting. $u_0 = \mathbb{E}[Z]$ is a constant and the estimation of u_0 in each partition is also a constant via computation of discrete conditional expectation conditioning on the complementary set in each partition. Estimating $g_{0,t}$ is similar to estimating m_0. For each partition $(I_{k,t}, I_{k,t}^c)$, we use a BART model with $I_{k,t}^c$ as training data and fitted the model with $I_{k,t}$. After that, we get one estimator for $SATT_{yearly}$ by solving the identification condition equation $\mathbb{E}[\psi(W_t; \theta_0^t, \hat{\eta}_0^{k,t})] = 0$. For data from a single year, variable $id.practice$ could be used to uniquely identify one observation. To create K folds, the yearly sample-splitting is based on the variable $id.practice$.

Considering the estimation of $SATT_{overall}$, the most difficult task is to create K folds. In principle, we could create one partition to estimate $SATT_{overall}$ by combining $(I_{k_1,year3}, I^c_{k_1,year3})$ with $(I_{k_2,year4}, I^c_{k_2,year4})$ for $1 \le k_1, k_2 \le K$. If k_1 and k_2 are chosen at random, it needs additional effort. For the sake of simplicity, we use $k_1 = k_2 = k$ to create the kth partition $(I_{k,year3} \cup I_{k,year4}, I^c_{k,year3} \cup I^c_{k,year4})$. In this case, we also use the variable $id.practice$ as a unique identifier to create K folds and the kth fold $I_{k,t}$ consists of a set of distinct $id.practice$ values. As a result, year3 data and year4 data with the same $id.practice$ value will be assigned to the same fold. Hence,

$$I_{k,year3} = I_{k,year4}, \tag{5.47}$$

$$I^c_{k,year3} = I^c_{k,year4} \tag{5.48}$$

The process to estimate u_0 and g_0 over the kth partition is the same as that for estimating $SATT_{yearly}$. Estimating m_0 is another story. Since the covariates W used to estimate m_0 only depend on data from the pre-treatment stage (year1 and year2), whether to estimate $SATT_{yearly}$ or $SATT_{overall}$ has no effect on the estimation result of m_0 as long as the partition of $id.practice$ is the same.

Figure 5.2 depicts the partition process used to estimate SATTs across year 3 and year 4.

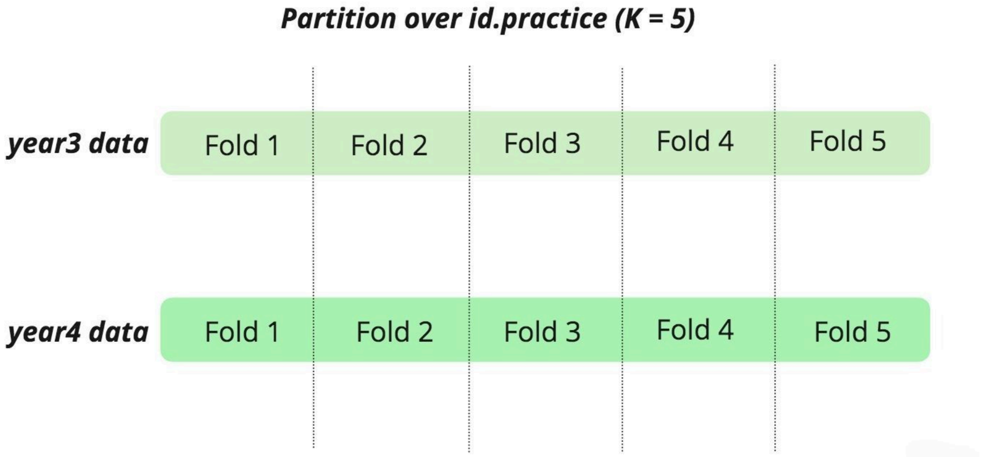

Figure 5.2 Illustration of the partition process to estimate $SATT_{overall}$

Estimating $SATT_{subgroup}$ differs greatly from estimating $SATT_{overall}$ and $SATT_{yearly}$. To estimate $SATT_{subgroup}$ using double machine learning, we make the following two modifications:

1. **Train m_0 and g_0 without sample-splitting:**

   In one specified subgroup, the sample size is considerably small. As a result, there may be not enough samples in the training set and overfitting is likely to occur. Like the procedure to estimate $SATT_{overall}$, we use variable $id.practice$ as the identifier to create K subgroup folds and combined the kth subgroup fold in year3 with the kth subgroup fold in year4. For partition $(I_{k,S}, I_{k,S}^c)$ within the subgroup S, we simply utilize the estimation $\hat{e}$ in Section 3.5.3 for $\hat{m}_0^{k,t}$ without any additional efforts because they are predicted without sample-splitting. We should notice that $\hat{m}_0^{k,year3} = \hat{m}_0^{k,year4} = \hat{m}_0^k$ if variable $id.practice$ is used to create subgroup folds. We use (X_t, Z) with outcome Y_t to train a BART model over the entire dataset and predict over the kth subgroup fold $I_{k,t,S}$. Hence, $\hat{g}_0^{k,t}$ is created. $\hat{u}_0^{k,t}$ is a constant via computation of discrete conditional expectation conditioning on the complementary set $I_{k,S}^c$ within subgroup S.

2. **Reduce number of folds K:**

   Given that the score function to estimate ATT on the kth fold I_k with partition (I_k, I_k^c):

   $$\psi(W; \check{\theta}_0^k, \hat{\eta}_0^k(I_k^c)) = \frac{Z(Y - \hat{g}_0^k(0, X))}{\hat{u}_0^k} - \frac{\hat{m}_0^k(M)(1-Z)(Y - \hat{g}_0^k(0, X))}{\hat{u}_0^k(1 - \hat{m}_0^k(M))} - \check{\theta}_0^k \frac{Z}{\hat{u}_0^k}, \quad (5.49)$$

   where $u_0 = \mathbb{E}[Z]$, $\hat{\eta}_0^k(I_k^c)$ consists of the machine learning estimators which are modelled from the complementary set I_k^c,

   $$\begin{aligned} \hat{u}_0^k &= \mathbb{E}[Z_i = 1 \mid i \in I_k^c] \\ &= \frac{1}{N-n} \sum_{i \in I_k^c} Z_i. \end{aligned} \quad (5.50)$$

   It is necessary to require $\hat{u}_0^k > 0$ to get a meaningful computation. Since the sample size for one subgroup is small, if we split them into smaller folds, it is more likely to encounter a fold with all Z_is equal to zero. To avoid meaningless computation, we should reduce the number of folds K in $SATT_{subgroup}$ estimation.

For all SATTs estimations, suppose that we use K folds sample-splitting and cross-fitting, with regard to the kth partition, we obtain an estimator $\check{\theta}_0^k$ for θ_0 by solving identification condition equation. We adapt Procedure 5.2 in Section 5.4 to estimate SATTs by averaging K estimators $\{\check{\theta}_0^1, \check{\theta}_0^2, \ldots, \check{\theta}_0^K\}$, and then got final estimator $\tilde{\theta}_0$.

5.6.2 Details of the Processes in Section 5.6.1

The following subsections explain the processes for estimating $SATT_{yearly}$, $SATT_{overall}$ and $SATT_{subgroup}$. Since we are handling practice-level data, it is necessary to include the number of patients in each practice as an additional coefficient in the equation $\mathbb{E}[\psi(W;\theta_0,\hat{\eta}_0)] = 0$.

The number of patients in practice i at year t is $n\_patients_{t,i}$, and each dataset has 500 practices. In this case, we use new notations for t. $t = 3$ represents $year3$ and $t = 4$ represents $year4$.

5.6.2.1 Estimation of $SATT_{yearly}$ at Year t

Step.1 Estimate $SATT_{yearly}$ over the kth partition $(I_{k,t}, I^c_{k,t})$ where $k \in \{1, 2, \ldots, K\}$:

We create the partition over $id.practice$ and $I_{k,t}$ denotes the set of distinct $id.practice$ values in the kth fold at year t. Then:

$$\begin{aligned}
\mathbb{E}[\psi(W;\theta_0,\hat{\eta}_0^{k,t})] &= \frac{1}{\sum_{i\in I_{k,t}} n\_patients_{t,i}} \sum_{i\in I_{k,t}} n\_patients_{t,i} \times \\
&\quad [\frac{Z_i(Y_{t,i} - \hat{g}_0^{k,t}(0,x_i))}{\hat{u}_0^{k,t}} - \frac{\hat{m}_0^k(M_i)(1-Z_i)(Y_{t,i} - \hat{g}_0^{k,t}(0,x_i))}{\hat{u}_0^{k,t}(1-\hat{m}_0^k(M_i))} - \theta_0 \frac{Z_i}{\hat{u}_0^{k,t}}] \\
&= \frac{1}{\sum_{i\in I_{k,t}} n\_patients_{t,i}} (- \sum_{i\in I_{k,t}} n\_patients_{t,i} \times \frac{Z_i}{\hat{u}_0^{k,t}} \times \theta_0 \\
&\quad + \sum_{i\in I_{k,t}} n\_patients_{t,i} \times [\frac{Z_j(Y_{t,i} - \hat{g}_0^{k,t}(0,x_i))}{\hat{u}_0^{k,t}} \\
&\quad - \frac{\hat{m}_0^k(M_i)(1-Z_i)(Y_{t,i} - \hat{g}_0^{k,t}(0,x_i))}{\hat{u}_0^{k,t}(1-\hat{m}_0^k(M_i))}]) \\
&:= \frac{1}{\sum_{i\in I_{k,t}} n\_patients_{t,i}} [\psi_a^{sum,t}(W;\hat{\eta}_0^{k,t})\theta_0 + \psi_b^{sum,t}(W;\hat{\eta}_0^{k,t})],
\end{aligned} \tag{5.51}$$

where:

$$\psi_a^{sum,t}(W;\hat{\eta}_0^{k,t}) := \sum_{i\in I_{k,t}} \psi_a(W_i;\hat{\eta}_0^{k,t}), \tag{5.52}$$

$$\psi_b^{sum,t}(W;\hat{\eta}_0^{k,t}) := \sum_{i\in I_{k,t}} \psi_b(W_i;\hat{\eta}_0^{k,t}). \tag{5.53}$$

Solve identification condition equation $\mathbb{E}[\psi(W;\theta_0,\hat{\eta}_0^{k,t})]=0$, yields:

$$
\begin{aligned}
\check{\theta}_0^{k,t} &= -\frac{\psi_b^{sum,t}(W;\hat{\eta}_0^{k,t})}{\psi_a^{sum,t}(W;\hat{\eta}_0^{k,t})} \\
&= \frac{\psi_b^{sum,t}(W;\hat{\eta}_0^{k,t})}{-\psi_a^{sum,t}(W;\hat{\eta}_0^{k,t})} \\
&= \frac{\sum_{i\in I_{k,t}} n\_patients_{t,i} \times [\frac{Z_i(Y_{t,i}-\hat{g}_0^{k,t}(0,x_i))}{\hat{u}_0^{k,t}} - \frac{\hat{m}_0^k(M_i)(1-Z_i)(Y_{t,i}-\hat{g}_0^{k,t}(0,x_i))}{\hat{u}_0^{k,t}(1-\hat{m}_0^k(M_i))}]}{\sum_{i\in I_{k,t}} n\_patients_{t,i} \times \frac{Z_i}{\hat{u}_0^{k,t}}}.
\end{aligned}
\tag{5.54}
$$

Given that variable $id.practice$ is used to create folds, $\hat{m}_0^k$ is not affected by the choice of year t and distinct t shares the same partition over $id.practice$.

Step.2 **Estimate $SATT_{yearly}$ at year t via cross-fitting:**

The final estimator for $SATT_{yearly}$ at year t is calculated by averaging K estimators $\{\check{\theta}_0^{1,t},\check{\theta}_0^{2,t},\ldots,\check{\theta}_0^{K,t}\}$.

$$
\tilde{\theta}_0^t = \sum_{k=1}^{K} \check{\theta}_0^{k,t}. \tag{5.55}
$$

Step.3 **Construct $(1-\alpha)\%$ approximate confidence interval:**

Applying Theorem 5, yields:

$$
\begin{aligned}
\hat{J}_0^t &= \frac{1}{\sum_{k=1}^{K}\sum_{i\in I_{k,t}} n\_patients_{t,i}} \sum_{k=1}^{K}\sum_{i\in I_{k,t}} n\_patients_{t,i} \times \psi_a(W_i;\hat{\eta}_0^{k,t}) \\
&= \frac{1}{\sum_{i=1}^{500} n\_patients_{t,i}} \sum_{i=1}^{500} n\_patients_{t,i} \times \frac{Z_i}{\hat{u}_0^k},
\end{aligned}
\tag{5.56}
$$

$$
\begin{aligned}
(\hat{\sigma}^t)^2 &= (\hat{J}_0^t)^{-2} \frac{1}{\sum_{k=1}^{K}\sum_{i\in I_{k,t}} n\_patients_{t,i}} \sum_{k=1}^{K}\sum_{i\in I_{k,t}} n\_patients_{t,i} \times [\psi(W_i;\tilde{\theta}_0^t,\hat{\eta}_0^{k,t})]^2 \\
&= (\hat{J}_0^t)^{-2} \frac{1}{\sum_{i=1}^{500} n\_patients_{t,i}} \sum_{i=1}^{500} n\_patients_{t,i} \times [\psi_a(W_i;\hat{\eta}_0^{k,t})\tilde{\theta}_0^t + \psi_b(W_i;\hat{\eta}_0^{k,t})]^2.
\end{aligned}
\tag{5.57}
$$

Thus, the $(1-\alpha)\%$ approximate confidence interval is give by:

$$
[\ \tilde{\theta}_0^t - \Phi^{-1}(1-\alpha/2)\hat{\sigma}^t / \sqrt{\sum_{i=1}^{500} n\_patients_{t,i}}\ ,\ \tilde{\theta}_0^t + \Phi^{-1}(1-\alpha/2)\hat{\sigma}^t / \sqrt{\sum_{i=1}^{500} n\_patients_{t,i}}\ ]. \tag{5.58}
$$

5.6.2.2 Estimation of $SATT_{overall}$

Step.1 **Estimate $SATT_{overall}$ over the kth partition (I_k, I_k^c) where $k \in \{1, 2, \dots, K\}$**

We create the partition over $id.practice$ and $I_{k,t}$ denotes the set of distinct $id.practice$ values in the kth fold at year t. Since we divide 500 $id.practice$ values to make folds and each $id.practice$ value corresponds to four-years data. Thus, the partition over $id.practice$ satisfies $(I_{k,3}, I_{k,3}^c) = (I_{k,4}, I_{k,4}^c) = (I_k, I_k^c)$. Then:

$$\begin{aligned}
\mathbb{E}[\psi(W;\theta_0,\hat{\eta}_0^{k,overall})] &= \frac{1}{\sum_{t=3}^{4}\sum_{i\in I_{k,t}} n\_patients_{t,i}} \sum_{t=3}^{4}\sum_{i\in I_{k,t}} n\_patients_{t,i} \times \\
&\quad [\frac{Z_i(Y_{t,i}-\hat{g}_0^{k,t}(0,x_i))}{\hat{u}_0^{k,overall}} - \frac{\hat{m}_0^k(M_i)(1-Z_i)(Y_{t,i}-\hat{g}_0^{k,t}(0,x_i))}{\hat{u}_0^{k,overall}(1-\hat{m}_0^k(M_i))} - \theta_0 \frac{Z_i}{\hat{u}_0^{k,overall}}] \\
&= \frac{1}{\sum_{t=3}^{4}\sum_{i\in I_{k,t}} n\_patients_{t,i}} \times (-n\_patients_{t,i} \times \sum_{t=3}^{4}\sum_{i\in I_{k,t}} \frac{Z_i}{\hat{u}_0^{k,overall}} \\
&\quad \times \theta_0 + \sum_{t=3}^{4}\sum_{i\in I_{k,t}} n\_patients_{t,i} \times [\frac{Z_j(Y_{t,i}-\hat{g}_0^{k,t}(0,x_i))}{\hat{u}_0^{k,overall}} \\
&\quad - \frac{\hat{m}_0^k(M_i)(1-Z_i)(Y_{t,i}-\hat{g}_0^{k,t}(0,x_i))}{\hat{u}_0^{k,overall}(1-\hat{m}_0^k(M_i))}]) \\
&:= \frac{1}{\sum_{t=3}^{4}\sum_{i\in I_{k,t}} n\_patients_{t,i}} [\psi_a^{sum,overall}(W;\hat{\eta}_0^{k,overall})\theta_0 \\
&\quad + \psi_b^{sum,overall}(W;\hat{\eta}_0^{k,overall})],
\end{aligned} \tag{5.59}$$

where:

$$\psi_a^{sum,overall}(W;\hat{\eta}_0^{k,overall}) := \sum_{t=3}^{4}\sum_{i\in I_{k,t}} \psi_a(W_i;\hat{\eta}_0^{k,overall,t}), \tag{5.60}$$

$$\psi_b^{sum,overall}(W;\hat{\eta}_0^{k,overall}) := \sum_{t=3}^{4}\sum_{i\in I_{k,t}} \psi_b(W_i;\hat{\eta}_0^{k,overall,t}). \tag{5.61}$$

Solve identification condition equation $\mathbb{E}[\psi(W;\theta_0,\hat{\eta}_0^{k,overall})]=0$, yields:

$$\begin{aligned}
\check{\theta}_0^{k,overall} &= -\frac{\psi_b^{sum,overall}(W;\hat{\eta}_0^{k,overall})}{\psi_a^{sum,overall}(W;\hat{\eta}_0^{k,overall})} \\
&= \frac{\psi_b^{sum,overall}(W;\hat{\eta}_0^{k,overall})}{-\psi_a^{sum,overall}(W;\hat{\eta}_0^{k,overall})} \\
&= \frac{\sum_{t=3}^{4}\sum_{i\in I_{k,t}} n\_patients_{t,i} \times [\frac{Z_i(Y_{t,i}-\hat{g}_0^{k,t}(0,x_i))}{\hat{u}_0^{k,overall}} - \frac{\hat{m}_0^k(M_i)(1-Z_i)(Y_{t,i}-\hat{g}_0^{k,t}(0,x_i))}{\hat{u}_0^{k,overall}(1-\hat{m}_0^k(M_i))}]}{\sum_{t=3}^{4}\sum_{i\in I_{k,t}} n\_patients_{t,i} \times \frac{Z_i}{\hat{u}_0^{k,overall}}}.
\end{aligned} \tag{5.62}$$

Given that variable $id.practice$ is used to create folds, $\hat{m}_0^k$ is not affected by the choice of year t and distinct t shares the same partition over $id.practice$.

Step.2 Estimate $SATT_{overall}$ via cross-fitting:

The final estimator for $SATT_{overall}$ is calculated by averaging K estimators $\{\check{\theta}_0^{1,overall},\check{\theta}_0^{2,overall}$ $,\dots,\check{\theta}_0^{K,overall}\}$.

$$\tilde{\theta}_0^{overall} = \sum_{k=1}^{K}\check{\theta}_0^{k,overall}. \tag{5.63}$$

Step.3 Construct $(1-\alpha)\%$ approximate confidence interval:

Applying Theorem 5, yields:

$$\begin{aligned}
\hat{J}_0^{overall} &= \frac{1}{\sum_{t=3}^{4}\sum_{i=1}^{500} n\_patients_{t,i}}\sum_{t=3}^{4}\sum_{k=1}^{K}\sum_{i\in I_{k,t}} n\_patients_{t,i}\times\psi_a(W_i;\hat{\eta}_0^{k,overall,t}) \\
&= \frac{1}{\sum_{t=3}^{4}\sum_{i=1}^{500} n\_patients_{t,i}}\sum_{t=3}^{4}\sum_{i=1}^{500} n\_patients_{t,i}\times\frac{Z_i}{\hat{u}_0^k},
\end{aligned} \tag{5.64}$$

$$\begin{aligned}
(\hat{\sigma}^{overall})^2 &= (\hat{J}_0^{overall})^{-2}\frac{1}{\sum_{t=3}^{4}\sum_{i=1}^{500} n\_patients_{t,i}}\sum_{t=3}^{4}\sum_{k=1}^{K}\sum_{i\in I_{k,t}}[\psi(W_i;\tilde{\theta}_0^{overall},\hat{\eta}_0^{k,overall,t})]^2 \\
&= (\hat{J}_0^{overall})^{-2}\frac{1}{\sum_{t=3}^{4}\sum_{i=1}^{500} n\_patients_{t,i}}\sum_{t=3}^{4}\sum_{i=1}^{500}[\psi_a(W_i;\hat{\eta}_0^{k,t,overall})\tilde{\theta}_0^{overall} \\
&\quad +\psi_b(W_i;\hat{\eta}_0^{k,overall,t})]^2.
\end{aligned} \tag{5.65}$$

Thus, the $(1-\alpha)\%$ approximate confidence interval is given by:

$$
\begin{aligned}
[\ \tilde{\theta}_0^{overall} - \Phi^{-1}(1-\alpha/2)\hat{\sigma}^{overall}/\sqrt{\sum_{t=3}^{4}\sum_{i=1}^{500} n\_patients_{t,i}}\ ,\ \tilde{\theta}_0^{overall} \\
+\Phi^{-1}(1-\alpha/2)\hat{\sigma}^{overall}/\sqrt{\sum_{t=3}^{4}\sum_{i=1}^{500} n\_patients_{t,i}}\ ].
\end{aligned}
\tag{5.66}
$$

5.6.2.3 Estimation of $SATT_{subgroup}$ with Subgroup S

Let us denote $SATT_{subgroup}$ with subgroup S as $SATT(S)$

Step.1 Estimate $SATT(S)$ over the kth partition $(I_{k,S}, I_{k,S}^c)$ where $k \in \{1,2,\ldots,K\}$:

We create the partition over $id.practice$ within subgroup S and $I_{k,t,S}$ denotes the set of distinct $id.practice$ values in the kth fold at year t within Subgroup S. Since we split $id.practice$ values within Subgroup S to create folds and each $id.practice$ value corresponds to four-years data. Thus, the partition over $id.practice$ satisfies $(I_{k,3,S}, I_{k,3,S}^c) = (I_{k,4,S}, I_{k,4,S}^c) = (I_{k,S}, I_{k,S}^c)$. Then:

$$
\begin{aligned}
\mathbb{E}[\psi(W;\theta_0,\hat{\eta}_0^{k,S})] &= \frac{1}{\sum_{t=3}^{4}\sum_{i\in I_{k,t,S}} n\_patients_{t,i}} \sum_{t=3}^{4}\sum_{i\in I_{k,t,S}} n\_patients_{t,i} \times \\
&\quad [\frac{Z_i(Y_{t,i}-\hat{g}_0^{k,t}(0,x_i))}{\hat{u}_0^{k,S}} - \frac{\hat{m}_0^k(M_i)(1-Z_i)(Y_{t,i}-\hat{g}_0^{k,t}(0,x_i))}{\hat{u}_0^{k,S}(1-\hat{m}_0^k(M_i))} - \theta_0\frac{Z_i}{\hat{u}_0^{k,S}}] \\
&= \frac{1}{\sum_{t=3}^{4}\sum_{i\in I_{k,t,S}} n\_patients_{t,i}} \times (-\sum_{t=3}^{4}\sum_{i\in I_{k,t,S}} n\_patients_{t,i} \times \frac{Z_i}{\hat{u}_0^{k,S}} \times \theta_0 \\
&\quad + \sum_{t=3}^{4}\sum_{i\in I_{k,t,S}} n\_patients_{t,i} \times [\frac{Z_j(Y_{t,i}-\hat{g}_0^{k,t}(0,x_i))}{\hat{u}_0^{k,S}} \\
&\quad - \frac{\hat{m}_0^k(M_i)(1-Z_i)(Y_{t,i}-\hat{g}_0^{k,t}(0,x_i))}{\hat{u}_0^{k,S}(1-\hat{m}_0^k(M_i))}]) \\
&:= \frac{1}{\sum_{t=3}^{4}\sum_{i\in I_{k,t,S}} n\_patients_{t,i}} [\psi_a^{sum,S}(W;\hat{\eta}_0^{k,S})\theta_0 + \psi_b^{sum}(W;\hat{\eta}_0^{k,S})],
\end{aligned}
\tag{5.67}
$$

where:

$$\psi_a^{sum,S}(W;\hat{\eta}_0^{k,S}) \coloneqq \sum_{t=3}^{4}\sum_{i\in I_{k,t,S}} \psi_a(W_i;\hat{\eta}_0^{k,S,t}), \tag{5.68}$$

$$\psi_b^{sum,S}(W;\hat{\eta}_0^{k,S}) \coloneqq \sum_{t=3}^{4}\sum_{i\in I_{k,t,S}} \psi_b(W_i;\hat{\eta}_0^{k,S,t}). \tag{5.69}$$

Solve identification condition equation $\mathbb{E}[\psi(W;\theta_0,\hat{\eta}_0^{k,S})]=0$, yields:

$$\begin{aligned}
\check{\theta}_0^k &= -\frac{\psi_b^{sum,S}(W;\hat{\eta}_0^{k,S})}{\psi_a^{sum,S}(W;\hat{\eta}_0^{k,S})}\\
&= \frac{\psi_b^{sum,S}(W;\hat{\eta}_0^{k,S})}{-\psi_a^{sum,S}(W;\hat{\eta}_0^{k,S})}\\
&= \frac{\sum_{t=3}^{4}\sum_{i\in I_{k,t,S}} n\_patients_{t,i}\times[\frac{Z_i(Y_{t,i}-\hat{g}_0^{k,t}(0,x_i))}{\hat{u}_0^{k,S}}-\frac{\hat{m}_0^k(M_i)(1-Z_i)(Y_{t,i}-\hat{g}_0^{k,t}(0,x_i))}{\hat{u}_0^{k,S}(1-\hat{m}_0^k(M_i))}]}{\sum_{t=3}^{4}\sum_{i\in I_{k,t,S}} n\_patients_{t,i}\times\frac{Z_i}{\hat{u}_0^{k,S}}}.
\end{aligned} \tag{5.70}$$

Given that variable $id.practice$ is used to create folds, $\hat{m}_0^k$ is not affected by the choice of year t and distinct t shares the same partition over $id.practice$.

Step.2 Estimate $SATT(S)$ via cross-fitting:

The final estimator for $SATT(S)$ is calculated by averaging K estimators $\{\check{\theta}_0^{1,S},\check{\theta}_0^{2,S},\dots,\check{\theta}_0^{K,S}\}$.

$$\tilde{\theta}_0^S=\sum_{k=1}^{K}\check{\theta}_0^{k,S}. \tag{5.71}$$

Step.3 Construct $(1-\alpha)\%$ approximate confidence interval:

Applying Theorem 5, yields:

$$\begin{aligned}
(\hat{J}_0)^S &= \frac{1}{\sum_{t=3}^{4}\sum_{k=1}^{K}\sum_{i\in I_{k,t,S}} n\_patients_{t,i}}\sum_{t=3}^{4}\sum_{k=1}^{K}\sum_{i\in I_{k,t,S}} n\_patients_{t,i}\times\psi_a(W_i;\hat{\eta}_0^{k,S,t})\\
&= \frac{1}{\sum_{t=3}^{4}\sum_{k=1}^{K}\sum_{i\in I_{k,t,S}} n\_patients_{t,i}}\sum_{t=3}^{4}\sum_{k=1}^{K}\sum_{i\in I_{k,t,S}} n\_patients_{t,i}\times\frac{Z_i}{\hat{u}_0^{k,S}},
\end{aligned} \tag{5.72}$$

$$
\begin{aligned}
(\hat{\sigma}^S)^2 &= (\hat{J}_0^S)^{-2} \frac{1}{\sum_{t=3}^{4}\sum_{k=1}^{K}\sum_{i\in I_{k,t,S}} n\_patients_{t,i}} \sum_{t=3}^{4}\sum_{k=1}^{K}\sum_{i\in I_{k,t,S}} n\_patients_{t,i} \\
&\quad \times [\psi(W_i;\tilde{\theta}_0,\hat{\eta}_0^{k,S,t})]^2 \\
&= (\hat{J}_0^S)^{-2} \frac{1}{\sum_{t=3}^{4}\sum_{k=1}^{K}\sum_{i\in I_{k,t,S}} n\_patients_{t,i}} \sum_{t=3}^{4}\sum_{k=1}^{K}\sum_{i\in I_{k,t,S}} n\_patients_{t,i} \\
&\quad \times [\psi_a(W_i;\hat{\eta}_0^{k,S,t})\tilde{\theta}_0 + \psi_b(W_i;\hat{\eta}_0^{k,S,t})]^2.
\end{aligned}
\tag{5.73}
$$

Thus, the $(1-\alpha)\%$ approximate confidence interval is give by:

$$
\begin{aligned}
[\ \tilde{\theta}_0^S - \Phi^{-1}(1-\alpha/2)\hat{\sigma}^S/\sqrt{\sum_{t=3}^{4}\sum_{k=1}^{K}\sum_{i\in I_{k,t,S}} n\_patients_{t,i}}\ ,\ \tilde{\theta}_0^S \\
+\Phi^{-1}(1-\alpha/2)\hat{\sigma}^S/\sqrt{\sum_{t=3}^{4}\sum_{k=1}^{K}\sum_{i\in I_{k,t,S}} n\_patients_{t,i}}\ ].
\end{aligned}
\tag{5.74}
$$

The biggest challenge in above the estimations is computing $\hat{\sigma}^2$. We design the following two solutions.

1. Use two numeric vectors to save $(\psi_a(W_i;\hat{\eta}_0^{k,overall,t}),\psi_b(W_i;\hat{\eta}_0^{k,overall,t}))$, $(\psi_a(W_i;\hat{\eta}_0^{k,3}),\psi_b(W_i;\hat{\eta}_0^{k,3}))$, $(\psi_a(W_i;\hat{\eta}_0^{k,4}),\psi_b(W_i;\hat{\eta}_0^{k,4}))$, $(\psi_a(W_i;\hat{\eta}_0^{k,S,t}),\psi_b(W_i;\hat{\eta}_0^{k,S,t}))$ respectively for each $i\in I_{k,t}$ or $i\in I_{k,t,S}$ over the kth fold at year t;

2. Write custom functions with $\tilde{\theta}_0$ as the function input:

   ■ For $SATT_{yearly}$ at year t, define $f_k^t(x)$ over each fold:

   $$
   f_k^t(x) = \sum_{i\in I_{k,t}} [\psi_a(W_i;\hat{\eta}_0^{k,t}) \times x + \psi_b(W_i;\hat{\eta}_0^{k,t})]^2 \times n\_patients_{t,i}. \tag{5.75}
   $$

   ■ For $SATT_{overall}$, define $f_k^{overall}(x)$ over each fold:

   $$
   f_k^{overall}(x) = \sum_{t=3}^{4}\sum_{i\in I_{k,t}} [\psi_a(W_i;\hat{\eta}_0^{k,overall,t}) \times x + \psi_b(W_i;\hat{\eta}_0^{k,overall,t})]^2 \times n\_patients_{t,i}. \tag{5.76}
   $$

   ■ For $SATT(S)$, define $f_k^S(x)$ over each fold:

   $$
   f_k^S(x) = \sum_{t=3}^{4}\sum_{i\in I_{k,t,S}} [\psi_a(W_i;\hat{\eta}_0^{k,S,t}) \times x + \psi_b(W_i;\hat{\eta}_0^{k,S,t})]^2 \times n\_patients_{t,i}. \tag{5.77}
   $$

In our implementations, we use the second method to save storage space and make the computations clear.

5.6.3 Summary of DML Implementations in R

We implement double machine learning in R with the Interactive Regression model to estimate $SATTs$ and wrote R codes from scratch. In the procedures to estimate $SATT_{overall}$ and $SATT_{yearly}$, we select $K = 5$. Bach et al. found that $K = 4$ or $K = 5$ performs better than $K = 2$ in a variety of simulations and they made $K = 5$ the default setting [13]. However, we use $K = 3$ or $K = 2$ for estimations of $SATT_{subgroup}$ to avoid the case that $\hat{u}_0^k = 0$.

The score function defined in Definition 34 only involves $g(0, X)$ to estimate ATT. Consequently, in the implementation of the BART model to estimate $\hat{g}_0^{k,t}$ over the kth partition, we set $Z = 0$ in the predictors for test data.

Algorithm 8: Use Double machine learning to estimate $SATT_{overall}$ and $SATT_{yearly}$

```
Input: 3400 datasets and number of folds $K = 5$, significance level $\alpha = 0.05$.
Output: Estimated $SATT_{overall}$, $SATT_{yearly}$ and their corresponding approximate
        confidence intervals for each dataset.
Initialize the record dataset which is used to save estimated SATTs and the upper and
  lower bounds of confidence intervals
Initialize $count = 0$ which remembers the index to write the estimations in the record
  dataset  for $i = 1$ to 3400
    Read $i$th dataset
    Read the record dataset
    Construct dataframe $df_{m_0}$ with practice-level covariates $X$, pre-treatment outcomes $Y_1$
      and $Y_2$, number of patients in pre-treatment stage $size_1$ and $size_2$, difference of year1
      outcome and year2 outcome $diff$
    Construct dataframe $df_{year3}$ which consists of the practice-level covariates $X$,
      treatment status variable $Z$, pre-treatment outcomes $Y_1$ and $Y_2$, number of patients in
      pre-treatment stage $size_1$ and $size_2$ and difference of year1 outcome and year2
      outcome $diff$
    Construct dataframe $df_{year4}$ which consists of the practice-level covariates $X$,
      treatment status variable $Z$, pre-treatment outcomes $Y_1$ and $Y_2$, number of patients in
      pre-treatment stage $size_1$ and $size_2$, difference of year1 outcome and year2 outcome
      $diff$ and year3 outcomes $Y_3$
    Convert categorical variables in dataframes $df_{m_0}$, $df_{year3}$ and $df_{year4}$ to factors
    Use BayesTree::makeind to dummy-encode factors in dataframes $df_{m_0}$, $df_{year3}$ and
      $df_{year4}$
    /* Create 5 folds with R function caret::createFolds:                          */
    $cv\_folds\_train \leftarrow caret :: createFolds(1 : N_0, k = 5, list =$
      $TRUE, returnTrain = TRUE$
    Initialize numeric vectors $theta\_overall$, $theta\_year3$ and $theta_year4$, each with
      length $K$, to save $\check{\theta}_0^{k,overall}$, $\check{\theta}_0^{k,3}$, $\check{\theta}_0^{k,4}$ with $k \in \{1, 2, \dots, K\}$
    Initialize numeric vectors $psi\_a\_sum\_overall$, $psi\_a\_sum\_year3$ and
      $psi\_a\_sum\_year4$, each with length $K$, to save $\psi_a^{sum,overall}(W; \hat{\eta}_0^{k,overall})$,
      $\psi_a^{sum,3}(W; \hat{\eta}_0^{k,3})$, $\psi_a^{sum,4}(W; \hat{\eta}_0^{k,4})$ with $k \in \{1, 2, \dots, K\}$
```

Algorithm 8: Use Double machine learning to estimate $SATT_{overall}$ and $SATT_{yearly}$

```
    Initialize numeric vectors $psi\_b\_sum\_overall$, $psi\_b\_sum\_year3$ and $psi\_b\_sum\_year4$, each with length $K$, to save $\psi_b^{sum,overall}(W; \hat{\eta}_0^{k,overall})$, $\psi_a^{sum,3}(W; \hat{\eta}_0^{k,t})$, $\psi_a^{sum,4}(W; \hat{\eta}_0^{k,t})$ with $k \in \{1, 2, \ldots, K\}$
    Initialize numeric vectors $N\_overall$, $N\_year3$ and $N\_year4$, each with length $K$
    Initialize list vectors $psi\_squared\_overall$, $psi\_squared\_year3$ and $psi\_squared\_year4$, each with length $K$, to save custom functions $f_k^{overall}(x)$, $f_k^3(x)$, $f_k^4(x)$ with $k \in \{1, 2, \ldots, K\}$
    for $k = 1$ to $K$
        $train\_index \leftarrow cv\_folds\_train[[j]]$
        $test\_index \leftarrow setdiff(1 : 500, train\_index)$
        Change the column of $Z$ in $df_{year3}[test\_index, ]$ to $(0, 0, \ldots, 0)^t$ and save it as $df_{year3}^{target}$
        Change the column of $Z$ in $df_{year4}[test\_index, ]$ to $(0, 0, \ldots, 0)^t$ and save it as $df_{year4}^{target}$
        Run BART::pbart with $Z[train\_index]$ as the response variable for training data, $df_{m_0}[train\_index, ]$ as predictors for training data, predict $\hat{m}_0^k((df_{m_0})_i)$ for each $i \in test\_index$ via a logistic link function and mean of posterior draws and save it as vector $p\_score$
        Run BART::wbart with $Y_3[train\_index]$ as the response variable for training data, $df_{year3}[train\_index, ]$ as predictors for training data, $df_{year3}^{target}$ as predictors for test data, predict $\hat{g}_0^{k,3}(0, (df_{year3} \setminus Z)_i)$ for each $i \in test\_index$ via mean of posterior draws and save it as vector $g\_0\_year3$
        Run BART::wbart with $Y_4[train\_index]$ as the response variable for training data, $df_{year4}[train\_index, ]$ as predictors for training data, $df_{year4}^{target}$ as predictors for test data, predict $\hat{g}_0^{k,4}(0, (df_{year4} \setminus Z)_i)$ for each $i \in test\_index$ via mean of posterior draws and save it as vector $g\_0\_year4$
        /* Compute $N_k^{overall} = \sum_{t=3}^{4} \sum_{i \in I_{k,t}} n\_patients_{t,i}$ for $SATT_{overall}$:    */
        $N\_overall[k] \leftarrow sum(n\_patients_{year3}[test\_index] + n\_patients_{year4}[test\_index])$
```

Algorithm 8: Use Double machine learning to estimate $SATT_{overall}$ and $SATT_{yearly}$

```
    |
        /* Compute $N_k^t = \sum_{i \in I_{k,t}} n\_patients_{t,i},\ t \in \{3,4\}$ for $SATT_{yearly}$: */
        $N\_year3[k] \leftarrow sum(n\_patients_{year3}[test\_index])$
        $N\_year4[k] \leftarrow sum(n\_patients_{year4}[test\_index])$
        /* Compute $\hat{u}_0^{k,overall} = \frac{1}{\sum_{t=3}^{4} \sum_{i \in I_{k,t}^c} n\_patients_{t,i}} \sum_{t=3}^{4} \sum_{i \in I_{k,t}^c} Z_i \times n\_patients_{t,i}$: */
        $u\_0\_overall \leftarrow sum(Z[train\_index] \times n\_patients_{year3}[train\_index] + Z[train\_index] \times n\_patients_{year4}[train\_index])/sum(n\_patients_{year3}[train\_index] + n\_patients_{year4}[train\_index])$
        /* Compute $\hat{u}_0^{k,t} = \frac{1}{\sum_{i \in I_{k,t}^c} n\_patients_{t,i}} \sum_{i \in I_{k,t}^c} Z_i \times n\_patients_{t,i},\ t \in \{3,4\}$: */
        $u\_0\_year3 \leftarrow sum(Z[train\_index] \times n\_patients_{year3}[train\_index])/sum(n\_patients_{year3}[train\_index])$
        $u\_0\_year4 \leftarrow sum(Z[train\_index] \times n\_patients_{year4}[train\_index])/sum(n\_patients_{year4}[train\_index])$
        /* Compute $\psi_a^{sum,overall}(W; \hat{\eta}_0^{k,overall}) = \sum_{t=3}^{4} \sum_{i \in I_{k,t}} n\_patients_{t,i} \times \frac{Z_i}{\hat{u}_0^{k,overall}}$: */
        $psi\_a\_sum\_overall[k] \leftarrow sum(Z[test\_index] \times n\_patients_{year3}[test\_index] + Z[test\_index] \times n\_patients_{year4}[test\_index])/u\_0\_overall$
        /* Compute $\psi_a^{sum,t}(W; \hat{\eta}_0^{k,t}) = \sum_{i \in I_{k,t}} n\_patients_{t,i} \times \frac{Z_i}{\hat{u}_0^{k,t}},\ t \in \{3,4\}$: */
        $psi\_a\_sum\_year3[k] \leftarrow sum(Z[test\_index] \times n\_patients_{year3}[test\_index])/u\_0\_year3$
        $psi\_a\_sum\_year4[k] \leftarrow sum(Z[test\_index] \times n\_patients_{year4}[test\_index])/u\_0\_year4$
        /* Compute $\psi_b^{sum,overall}(W; \hat{\eta}_0^{k,overall}) = \sum_{t=3}^{4} \sum_{i \in I_{k,t}} n\_patients_{t,i} \times [\frac{Z_i(Y_{t,i} - \hat{g}_0^{k,t}(0,x_i))}{\hat{u}_0^{k,overall}} - \frac{\hat{m}_0^k(M_i)(1-Z_i)(Y_{t,i} - \hat{g}_0^{k,t}(0,x_i))}{\hat{u}_0^{k,overall}(1-\hat{m}_0^k(M_i))}]$: */
        $psi\_b\_sum\_overall[k] \leftarrow sum((Y_3[test\_index] - g\_0\_year3) \times Z[test\_index] \times n\_patients_{year3}[test\_index])/u\_0\_overall - sum(p\_score \times (1 - Z[test\_index]) \times (Y_3[test\_index] - g\_0\_year3) \times n\_patients_{year3}[test\_index]/(u\_0\_overall \times (1 - p\_score))) + sum((Y_4[test\_index] - g\_0\_year4) \times Z[test\_index] \times n\_patients_{year4}[test\_index])/u\_0\_overall - sum(p\_score \times (1 - Z[test\_index]) \times (Y_4[test\_index] - g\_0\_year4) \times n\_patients_{year4}[test\_index]/(u\_0\_overall \times (1 - p\_score)))$
```

Algorithm 8: Use Double machine learning to estimate $SATT_{overall}$ and $SATT_{yearly}$

```
        /* Compute $\psi_b^{sum,t}(W;\hat{\eta}_0^{k,t}) = \sum_{i \in I_{k,t}} n\_patients_{t,i} \times [\frac{Z_i(Y_{t,i}-\hat{g}_0^{k,t}(0,x_i))}{\hat{u}_0^{k,t}} - \frac{\hat{m}_0^k(M_i)(1-Z_i)(Y_{t,i}-\hat{g}_0^{k,t}(0,x_i))}{\hat{u}_0^{k,t}(1-\hat{m}_0^k(M_i))}]$, $t \in \{3,4\}$: */
        $psi\_b\_sum\_year3[k] \leftarrow sum((Y_3[test\_index] - g\_0\_year3) \times Z[test\_index] \times n\_patients_{year3}[test\_index])/u\_0\_overall - sum(p\_score \times (1 - Z[test\_index]) \times (Y_3[test\_index] - g\_0\_year3) \times n\_patients_{year3}[test\_index]/(u\_0\_overall \times (1 - p\_score)))$
        $psi\_b\_sum\_year4[k] \leftarrow sum((Y_4[test\_index] - g\_0\_year4) \times Z[test\_index] \times n\_patients_{year4}[test\_index])/u\_0\_overall - sum(p\_score \times (1 - Z[test\_index]) \times (Y_4[test\_index] - g\_0\_year4) \times n\_patients_{year4}[test\_index]/(u\_0\_overall \times (1 - p\_score)))$
        /* Compute $\psi_a(W_i;\hat{\eta}_0^{k,overall,t})$ and $\psi_b(W_i;\hat{\eta}_0^{k,overall,t})$, $t \in \{3,4\}$ to prepare for function $f_k^{overall}(x)$ definition: */
        $psi\_a\_overall \leftarrow Z[test\_index]/u\_0\_overall$
        $psi\_b\_overall_t3 \leftarrow (Y_3[test\_index] - g\_0\_year3) \times Z[test\_index]/u\_0\_overall - p\_score \times (1 - Z[test\_index]) \times (Y_3[test\_index] - g\_0\_year3)/(u\_0\_overall \times (1 - p\_score))$
        $psi\_b\_overall_t4 \leftarrow Y_4[test\_index] - g\_0\_year4) \times Z[test\_index]/u\_0\_overall - p\_score \times (1 - Z[test\_index]) \times (Y_4[test\_index] - g\_0\_year4)/(u\_0\_overall \times (1 - p\_score))$
        /* Compute $\psi_a(W_i;\hat{\eta}_0^{k,t})$ and $\psi_b(W_i;\hat{\eta}_0^{k,t})$ to prepare for function $f_k^t(x)$, $t \in \{3,4\}$ definitions: */
        $psi\_a\_year3 \leftarrow Z[test\_index]/u\_0\_year3$
        $psi\_a\_year4 \leftarrow Z[test\_index]/u\_0\_year4$
        $psi\_b\_year3 \leftarrow (Y_3[test\_index] - g\_0\_year3) \times Z[test\_index]/u\_0\_year3 - p\_score \times (1 - Z[test\_index]) \times (Y_3[test\_index] - g\_0\_year3)/(u\_0\_year3 \times (1 - p\_score))$
        $psi\_b\_year4 \leftarrow Y_4[test\_index] - g\_0\_year4) \times Z[test\_index]/u\_0\_year4 - p\_score \times (1 - Z[test\_index]) \times (Y_4[test\_index] - g\_0\_year4)/(u\_0\_year4 \times (1 - p\_score))$
```

Algorithm 8: Use Double machine learning to estimate $SATT_{overall}$ and $SATT_{yearly}$

```
        /* Define
          $f_k^{overall}(x) = \sum_{t=3}^{4} \sum_{i \in I_{k,t}} [\psi_a(W_i; \hat{\eta}_0^{k,overall,t}) \times x + \psi_b(W_i; \hat{\eta}_0^{k,overall,t})]^2 \times n\_patients_{t,i}$:    */
        $psi\_squared\_overall[[k]] \leftarrow function(x)sum((psi\_b\_overall\_t3 -$
          $psi\_a_overall \times x)^2 \times n\_patients_{year3}[test\_index]) + sum((psi\_b\_overall\_t4 -$
          $psi\_a\_overall \times x)^2 \times n\_patients_{year4}[test\_index])$
        /* Define $f_k^t(x) = \sum_{i \in I_{k,t}} [\psi_a(W_i; \hat{\eta}_0^{k,t}) \times x + \psi_b(W_i; \hat{\eta}_0^{k,t})]^2 \times n\_patients_{t,i}$, $t \in \{3, 4\}$:    */
        $psi\_squared\_year3[[k]] \leftarrow function(x)sum((psi\_b\_year3 - psi\_a\_year3 \times$
          $x)^2 \times n\_patients_{year3}[test\_index])$
        $psi\_squared\_year4[[k]] \leftarrow function(x)sum((psi\_b\_year4 - psi\_a\_year4 \times$
          $x)^2 \times n\_patients_{year4}[test\_index])$
        /* Compute $\check{\theta}_0^{k,overall}$:    */
        theta$_overall[k]¸solve(psi_asum_overall[j], psi_bsum_overall[k])theta\_overall[k] \leftarrow$
          $solve(psi\_a\_sum\_overall[j], psi\_b\_sum\_overall[k])$
        /* Compute $\check{\theta}_0^{k,t}$, $t \in \{3, 4\}$:    */
        $theta\_year3[k] \leftarrow solve(psi\_a\_sum\_year3[j], psi\_b\_sum\_year3[j])$
        $theta\_year4[k] \leftarrow solve(psi\_a\_sum\_year4[j], psi\_b\_sum\_year4[j])$
    end for
    /* Save the final estimators $\tilde{\theta}_0^{overall}$, $\tilde{\theta}_0^3$ and $\tilde{\theta}_0^4$ via the weighted mean of $K$ estimators
       respectively:    */
    $satt\_overall \leftarrow sum(theta\_overall \times N\_overall)/sum(N\_overall)$
    $satt\_year3 \leftarrow sum(theta\_year3 \times N\_year3)/sum(N\_year3)$
    $satt\_year4 \leftarrow sum(theta\_year4 \times N\_year4)/sum(N\_year4)$
    Initialize $psi\_squared\_sum\_overall \leftarrow 0$
    $psi\_squared\_sum\_year3 \leftarrow 0$
    $psi\_squared\_sum\_year4 \leftarrow 0$
```

Algorithm 8: Use Double machine learning to estimate $SATT_{overall}$ and $SATT_{yearly}$

```
for $k = 1$ to $K$
    /* Take $satt\_overall$ into function $psi\_squared\_overall[[k]]$ and update the sum across
       the folds:                                                                          */
    $psi\_squared\_sum\_overall \leftarrow$
     $psi\_squared\_sum\_overall + psi\_squared\_overall[[k]](satt\_overall)$
    /* Take $satt\_overall$ into function $psi\_squared\_overall[[k]]$ and update the sum across
       the folds:                                                                          */
    $psi\_squared\_sum\_overall \leftarrow$
     $psi\_squared\_sum\_overall + psi\_squared\_overall[[k]](satt\_overall)$
    /* Take $satt\_year3$ into function $psi\_squared\_year3[[k]]$ and update the sum across the
       folds:                                                                              */
    $psi\_squared\_sum\_year3 \leftarrow$
     $psi\_squared\_sum\_year3 + psi\_squared\_year3[[k]](satt\_year3)$
    /* Take $satt\_year4$ into function $psi\_squared\_year4[[k]]$ and update the sum across the
       folds:                                                                              */
    $psi\_squared\_sum\_year4 \leftarrow$
     $psi\_squared\_sum\_year4 + psi\_squared\_year3[[k]](satt\_year4)$
end for
/* Compute estimated $\hat{\sigma}^{overall}$, $\hat{\sigma}^3$ and $\hat{\sigma}^4$ for $\tilde{\theta}_0^{overall}$, $\tilde{\theta}_0^3$ and $\tilde{\theta}_0^4$ respectively:      */
$sigma\_overall \leftarrow$
 $\sqrt{sum(N\_overall)} \times \sqrt{psi\_squared\_sum\_overall}/sum(psi\_a\_overall)$
$sigma\_year3 \leftarrow \sqrt{sum(N\_year3)} \times \sqrt{psi\_squared\_sum\_year3}/sum(psi\_a\_year3)$
$sigma\_year4 \leftarrow \sqrt{sum(N\_year4)} \times \sqrt{psi\_squared\_sum\_year4}/sum(psi\_a\_year4)$
/* Save estimated SATTs to the record dataset:                                             */
$df\_record\$satt[count + 1] \leftarrow satt\_overall$
$df\_record\$satt[count + 2] \leftarrow satt\_year3$
$df\_record\$satt[count + 3] \leftarrow satt\_year4$
```

Algorithm 8: Use Double machine learning to estimate $SATT_{overall}$ and $SATT_{yearly}$

```
    /* Construct approximate $(1-\alpha)\%$ confidence intervals for SATTs and save the bounds
       into the record dataset:                                                              */
    $df\_record\$lower90[count+1] \leftarrow$
      $satt\_overall - qnorm(1-alpha/2) \times sigma\_overall/\sqrt{sum(N\_overall)}$
    $df\_record\$upper90[count+1] \leftarrow$
      $satt\_overall + qnorm(1-alpha/2) \times sigma\_overall/\sqrt{sum(N\_overall)}$
    $df\_record\$lower90[count+2] \leftarrow$
      $satt\_year3 - qnorm(1-alpha/2) \times sigma\_year3/\sqrt{sum(N\_year3)}$
    $df\_record\$upper90[count+2] \leftarrow$
      $satt\_year3 + qnorm(1-alpha/2) \times sigma\_year3/\sqrt{sum(N\_year3)}$
    $df\_record\$lower90[count+3] \leftarrow$
      $satt\_year4 - qnorm(1-alpha/2) \times sigma\_year4/\sqrt{sum(N\_year4)}$
    $df\_record\$upper90[count+3] \leftarrow$
      $satt\_year4 + qnorm(1-alpha/2) \times sigma\_year4/\sqrt{sum(N\_year4)}$
    Update $count \leftarrow count + 15$
end for
```

Algorithm 9: Use Double machine learning to estimate $SATT(S)$

```
Input: 3400 datasets, record dataset and hyperparameter $K = 2$ or $K = 3$, $\alpha = 0.05$.
Output: Estimated $SATT_{subgroup}$ and their corresponding approximate confidence
        intervals for each dataset.
Initialize $count = 0$ which remembers the index to write the estimations in the record
  dataset
for $i = 1$ to 3400
    Read $i$th dataset
    Read the record dataset
    Construct dataframe $df_{m_0}$ with practice-level covariates $X$, pre-treatment outcomes $Y_1$
      and $Y_2$, number of patients in pre-treatment stage $size_1$ and $size_2$, difference of year1
      outcome and year2 outcome $diff$
    Construct dataframe $df_{year3}$ which consists of the practice-level covariates $X$,
      treatment status variable $Z$, pre-treatment outcomes $Y_1$ and $Y_2$, number of patients in
      pre-treatment stage $size_1$ and $size_2$ and difference of year1 outcome and year2
      outcome $diff$ Construct dataframe $df_{year4}$ which consists of the practice-level
      covariates $X$, treatment status variable $Z$, pre-treatment outcomes $Y_1$ and $Y_2$,
      number of patients in pre-treatment stage $size_1$ and $size_2$, difference of year1
      outcome and year2 outcome $diff$ and year3 outcomes $Y_3$
    Convert categorical variables in dataframes $df_{m_0}$, $df_{year3}$ and $df_{year4}$ to factors
    Use BayesTree::makeind to dummy-encode factors in dataframes $df_{m_0}$, $df_{year3}$ and
      $df_{year4}$
    Change the column of $Z$ in $df_{year3}$ to $(0, 0, \dots, 0)^t$ and save it as $df_{year3}^{target}$
    Change the column of $Z$ in $df_{year4}$ to $(0, 0, \dots, 0)^t$ and save it as $df_{year4}^{target}$
    Run BART::pbart with $Z$ as the response variable for training data, $df_{m_0}$ as predictors
      for training data, $df_{year3}^{target}$ as predictors for test data, predict $\hat{m}_0^k((df_{m_0})_i)$ for each
      $i \in \{1, 2, \dots, 500\}$ via a logistic link function and mean of posterior draws and save
      it as vector $p_score$
    Run BART::wbart with $Y_3$ as the response variable for training data, $df_{year3}$ as
      predictors for training data, $df_{year3}$ as predictors for test data, predict
      $\hat{g}_0^{k,3}(0, (df_{year3} \setminus Z)_i)$ for each $i \in \{1, 2, \dots, 500\}$ via mean of posterior draws and
      save it as vector $g\_0\_year3$
```

Algorithm 9: Use Double machine learning to estimate $SATT(S)$

```
Run BART::wbart with $Y_4$ as the response variable for training data, $df_{year4}$ as
  predictors for training data, $df_{year4}$ as predictors for test data, predict
  $\hat{g}_0^{k,4}(0, (df_{year4} \setminus Z)_i)$ for each $i \in \{1, 2, \ldots, 500\}$ via mean of posterior draws and
  save it as vector $g\_0\_year4$
Extract all $id.practice$ values within Subgroup $S$ and save them in a vector $id\_S$
/* Create 5 folds with R function caret::createFolds:                                  */
$cv\_folds\_train\_S \leftarrow caret :: createFolds(id\_S, k = 5, list =$
  $TRUE, returnTrain = TRUE)$
Initialize numeric vectors $theta\_S$ with length $K$ to save $\check{\theta}_0^{k,S}$ with $k \in \{1, \ldots, K\}$
Initialize numeric vectors $psi\_a\_sum\_S$ with length $K$, to save $\psi_a^{sum,S}(W; \hat{\eta}_0^{k,S})$ with
  $k \in \{1, \ldots, K\}$
Initialize numeric vectors $psi\_b\_sum\_S$ with length $K$, to save $\psi_b^{sum,S}(W; \hat{\eta}_0^{k,S})$ with
  $k \in \{1, \ldots, K\}$
Initialize numeric vectors $N\_S$ with length $K$
Initialize list vectors $psi\_squared\_S$ with length $K$, to save custom functions $f_k^S(x)$
  with $k \in \{1, \ldots, K\}$
for $k = 1$ to $K$
    $train\_index\_S \leftarrow cv\_folds\_train_S[[j]]$
    $test\_index\_S \leftarrow id\_S[-train\_index_S]$
    $p\_score\_test \leftarrow p\_score[test\_index_S]$
    $g\_0\_year3\_test \leftarrow g\_0\_year3[test\_index\_S]$
    $g\_0\_year4\_test \leftarrow g\_0\_year4[test\_index\_S]$
    /* Compute $N_k^S = \sum_{t=3}^{4} \sum_{i \in I_{k,t,S}} n\_patients_{t,i}$:                  */
    $N\_S[k] \leftarrow sum(n\_patients_{year3}[test\_index\_S] + n\_patients_{year4}[test\_index\_S])$
    /* Compute $\hat{u}_0^{k,S} = \frac{1}{\sum_{t=3}^{4} \sum_{i \in I_{k,t,S}^c} n\_patients_{t,i}} \sum_{t=3}^{4} \sum_{i \in I_{k,t,S}^c} Z_i \times n\_patients_{t,i}$:    */
    $u\_0\_S \leftarrow$
      $sum(Z[train\_index\_S] \times n\_patients_{year3}[train\_index\_S] + Z[train\_index\_S] \times$
      $n\_patients_{year4}[train\_index\_S])/sum(n\_patients_{year3}[train\_index\_S] +$
      $n\_patients_{year4}[train\_index\_S])$
```

Algorithm 9: Use Double machine learning to estimate $SATT(S)$

```
        /* Compute $\psi_a^{sum,S}(W;\hat{\eta}_0^{k,S}) = \sum_{t=3}^{4}\sum_{i\in I_{k,t,S}} n\_patients_{t,i} \times \frac{Z_i}{\hat{u}_0^{k,S}}$: */
        $psi\_a\_sum\_S[k] \leftarrow sum(Z[test\_index\_S] \times n\_patients_{year3}[test\_index\_S] + Z[test\_index\_S] \times n\_patients_{year4}[test\_index\_S])/u\_0\_S$
        /* Compute $\psi_b^{sum,S}(W;\hat{\eta}_0^{k,S}) = \sum_{t=3}^{4}\sum_{i\in I_{k,t,S}} n\_patients_{t,i} \times [\frac{Z_i(Y_{t,i}-\hat{g}_0^{k,t}(0,x_i))}{\hat{u}_0^{k,S}} - \frac{\hat{m}_0^k(M_i)(1-Z_i)(Y_{t,i}-\hat{g}_0^{k,t}(0,x_i))}{\hat{u}_0^{k,S}(1-\hat{m}_0^k(M_i))}]$: */
        $psi\_b\_sum\_S[k] \leftarrow sum((Y_3[test\_index\_S] - g\_0\_year3\_test) \times Z[test\_index\_S] \times n\_patients_{year3}[test\_index\_S])/u\_0\_S - sum(p\_score\_test \times (1 - Z[test\_index\_S]) \times (Y_3[test\_index\_S] - g\_0\_year3\_test) \times n\_patients_{year3}[test\_index\_S]/(u\_0\_S \times (1 - p\_score\_test))) + sum((Y_4[test\_index\_S] - g\_0\_year4\_test) \times Z[test\_index\_S] \times n\_patients_{year4}[test\_index\_S])/u\_0\_S - sum(p\_score\_test \times (1 - Z[test\_index\_S]) \times (Y_4[test\_index\_S] - g\_0\_year4\_test) \times n\_patients_{year4}[test\_index\_S]/(u\_0\_S \times (1 - p\_score\_test)))$
        /* Compute $\psi_a(W_i;\hat{\eta}_0^{k,t,S})$ and $\psi_b(W_i;\hat{\eta}_0^{k,t,S})$, $t \in \{3,4\}$ to prepare for function $f_k^S(x)$ definition: */
        $psi\_a \leftarrow Z[test\_index\_S]/u\_0\_S$
        $psi\_b\_year3 \leftarrow (Y_3[test\_index\_S] - g\_0\_year3\_test) \times Z[test\_index\_S]/u\_0\_S - p\_score \times (1 - Z[test\_index\_S]) \times (Y_3[test\_index\_S] - g\_0\_year3\_test)/(u\_0\_S \times (1 - p\_score\_test))$
        $psi\_b\_year4 \leftarrow Y_4[test\_index\_S] - g\_0\_year4\_test) \times Z[test\_index\_S]/u\_0\_S - p\_score\_test \times (1 - Z[test\_index\_S]) \times (Y_4[test\_index\_S] - g\_0\_year4\_test)/(u\_0\_S \times (1 - p\_score\_test))$
        /* Define $f_k^S(x) = \sum_{t=3}^{4}\sum_{i\in I_{k,S,t}}[\psi_a(W_i;\hat{\eta}_0^{k,S,t}) \times x + \psi_b(W_i;\hat{\eta}_0^{k,t,S})]^2 \times n\_patients_{t,i}$: */
        $psi\_squared\_S[[k]] \leftarrow function(x)sum((psi\_b\_year3 - psi_a \times x)^2 \times n\_patients_{year3}[test\_index\_S]) + sum((psi\_b\_year4 - psi_a \times x)^2 \times n\_patients_{year4}[test\_index\_S])$
        /* Compute $\check{\theta}_0^{k,S}$: */
        $theta\_S[k] \leftarrow solve(psi\_sum\_a\_S[k], psi\_sum\_b\_S[k])$
    end for
```

Algorithm 9: Use Double machine learning to estimate $SATT(S)$

```
    for $k = 1$ to $K$
        /* Save the final estimator $\tilde{\theta}_0^S$ via the weighted mean of $K$ estimators: */
        $satt\_S \leftarrow sum(theta\_S \times N\_S)/sum(N\_S)$
        Initialize $psi\_squared\_sum\_S \leftarrow 0$
        for $k = 1$ to $K$
            /* Take $satt\_S$ into function $psi\_squared\_S[[k]]$ and update the sum across the folds: */
            $psi\_squared\_sum\_S \leftarrow psi\_squared\_sum\_S + psi\_squared\_S[[k]](satt\_S)$
        end for
        /* Take $satt\_S$ into function $psi\_squared\_S[[k]]$ and update the sum across the folds: */
        $psi\_squared\_sum\_S \leftarrow psi\_squared\_sum\_S + psi\_squared\_S[[k]](satt\_S)$
    end for
    /* Compute estimated $\hat{\sigma}^S$ for $\tilde{\theta}_0^S$: */
    $sigma\_S \leftarrow \sqrt{(}sum(N\_S)) \times \sqrt{(}psi\_squared\_sum\_S)/sum(psi\_a\_S)$
    Save $satt\_S$ to the record dataset /* Construct approximate $(1-\alpha)\%$ confidence intervals for SATTs and save the bounds into the record dataset: */
    $lower90 \leftarrow satt\_S - qnorm(1 - alpha/2) \times sigma\_S/\sqrt{(}sum(N\_S))$
    $upper90 \leftarrow satt\_S + qnorm(1 - alpha/2) \times sigma\_S/\sqrt{(}sum(N\_S))$
    Save $lower90$ and $upper90$ to the record dataset Update $count \leftarrow count + 15$
end for
```

6 Performance and Evaluation

6.1 Metrics

To assess model performance in estimating treatment effects SATTs, we use Root mean square error (RMSE), absolute bias, uncertainty interval coverage, and uncertainty interval width.

Definition 36 ⟨Root mean square error (RMSE) for SATTs⟩

$SATT_i$ is the ground-truth value of SATT statistics for the ith dataset, and $S\hat{ATT}_i$ is an estimation of this SATT statistics calculated through modelling, then the root mean square error is:

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(S\hat{ATT}_i - SATT_i)^2}{n}}. \tag{6.1}$$

Here, n: number of datasets, $S\hat{ATT}_i$: predicted SATT via modelling in the ith dataset, $SATT_i$: true SATT in the ith dataset

Definition 37 ⟨Uncertainty interval coverage rate for SATTs⟩

$SATT_1, SATT_2, \ldots, SATT_n$ are the ground-truth values of SATT statistics for n datasets, and $[lower_1, upper_1], \ldots, [lower_n, upper_n]$ are their corresponding uncertainty interval calculated through modelling, then the uncertainty interval coverage is:

$$ci\_coverage = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}\{SATT_i \in [lower_i, upper_i]\}. \tag{6.2}$$

Definition 38 ⟨Absolute biases for SATTs⟩

$SATT_i$ is the ground-truth value of SATT statistics for the ith dataset, and $S\hat{ATT}_i$ is an estimation of this SATT statistics calculated through modelling, then the absolute bias for the ith dataset is:

$$\textit{Absolute bias for the } i\textit{th dataset} = \mid S\hat{ATT}_i - SATT_i \mid . \tag{6.3}$$

Here, $\hat{SATT}_i$: predicted SATT via modelling in the ith dataset, $\;SATT_i$*: true SATT in the ith dataset*

Definition 39 ⟨Uncertainty interval widths for SATTs⟩

$SATT_i$ is the ground-truth value of SATT statistics for the ith dataset, and $[lower_i, upper_i]$ is its corresponding uncertainty interval calculated through modelling, then the uncertainty interval width is:

$$ci\_width \text{ for the } i\text{th dataset} = upper_i - lower_i. \tag{6.4}$$

The targeted confidence level for the SATT uncertainty interval estimates is 90 %, which applies to all of my implementations.

6.2 BART with Methods1 VERSUS BART with Method2

In Section 3.5.2, we described two methods for constructing credible intervals for SATTs. We implemented both BART with **Method.1** and BART with **Method.2** in R and their performances concerning $SATT_{overall}$ are shown in the table below.

	BART with **Method.1**	BART with **Method.2**
RMSE	16.8369	16.9078
Uncertainty interval coverage rate	81.41 %	93.62 %
Average uncertainty interval width	43.8814	61.7586

Table 6.1 Table to compare performances of BART with method1 and BART with method2.

From the table, we can see that they have nearly identical RMSEs and considerably high uncertainty interval coverage rates. However, it is clear that BART with **Method.2** results in a much wider uncertainty interval width. We discard BART with **Method.2** and used BART with **Method.1** for further discussions about BART implementation.

6.3 Importance of Propensity Score

When comparing the different performances of BART with **Method.1** and BART without propensity score concerning $SATT_{overall}$, BART with **Method.1** performs better in terms of RMSE. Meanwhile, BART without a propensity score has an uncertainty interval coverage rate of roughly 71%, which is much lower than BART with **Method.1**. We could conclude that propensity score should be included as an additional covariate in BART model implementations.

	BART with **Method.1**	BART without propensity score
RMSE	16.8369	17.9955
Uncertainty interval coverage	81.41 %	71.51 %
Average uncertainty interval width	43.8814	39.0253

Table 6.2 Table to compare performances of BART with method1 and BART without propensity score.

6.4 Visualization of BART Performance

Figure 6.1 shows the boxplot of absolute biases of $SATT_{overall}$, $SATT_{year3}$ and $SATT_{year4}$ which are estimated by BART with **Method.1**.

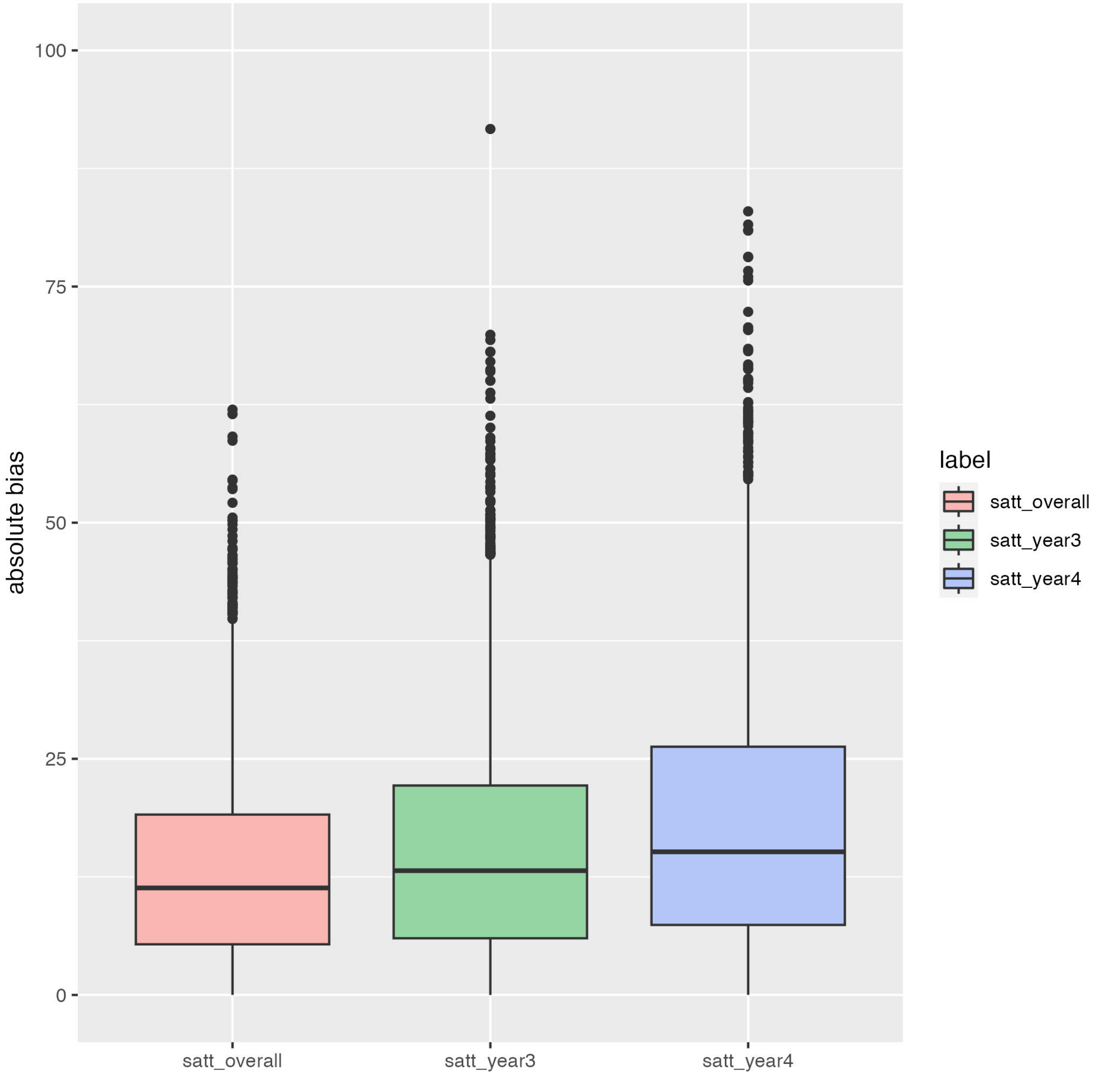

Figure 6.1 Boxplot of absolute biases of SATTs via BART with **Method.1**

Figure 6.2 shows the boxplot of uncertainty interval widths for $SATT_{overall}$, $SATT_{year3}$ and $SATT_{year4}$ which are estimated by BART with **Method.1**.

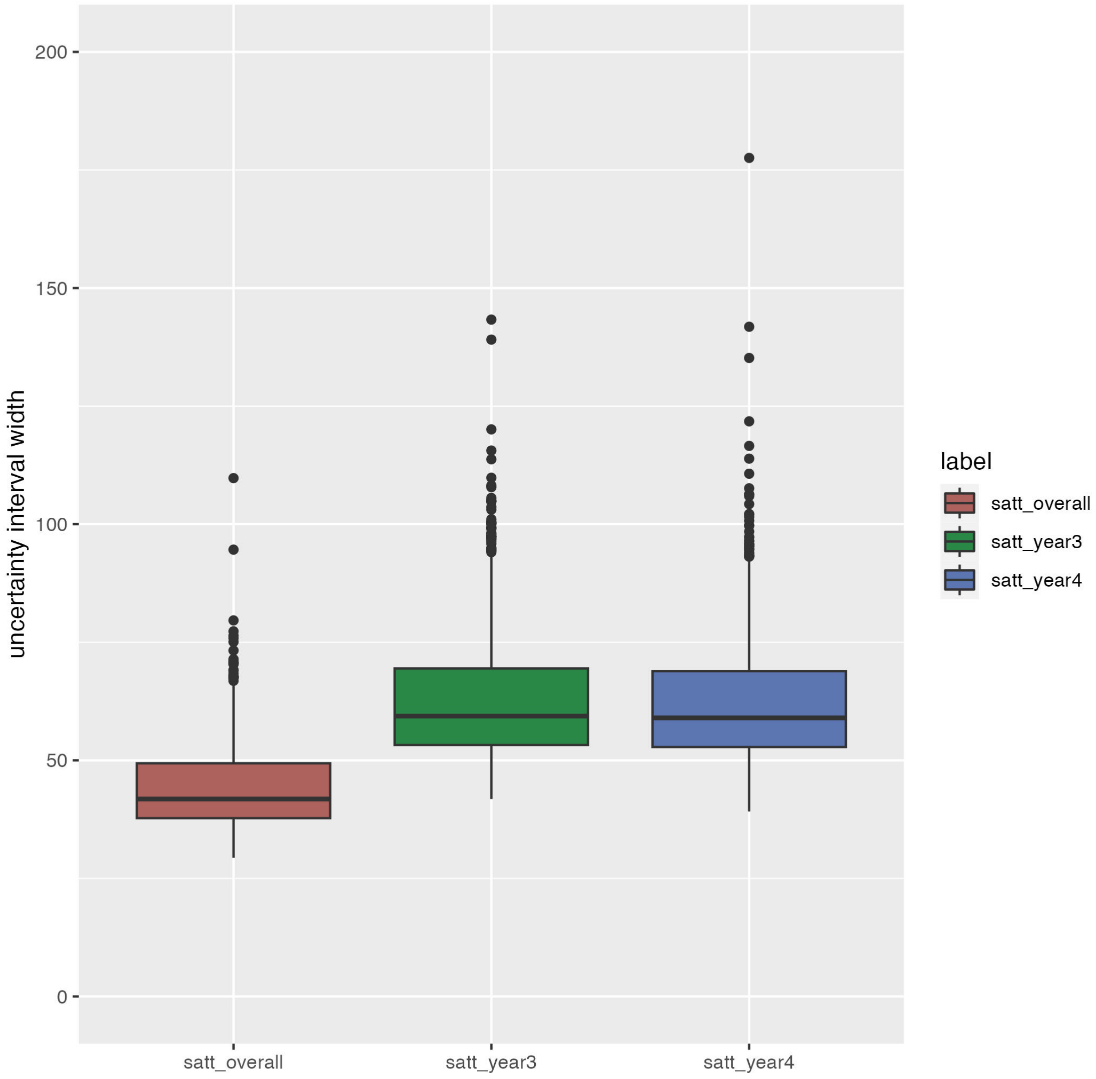

Figure 6.2 Boxplot of uncertainty interval widths of SATTs via BART with **Method.1**

Figure 6.3 shows the scatter plot of estimated $SATT_{overall}$ via BART with **Method.1** against ground-truth $SATT_{overall}$, where grey vertical lines represent the uncertainty intervals of estimated $SATT_{overall}$. If the ground-truth $SATT_{overall}$ falls within the uncertainty intervals, the point is marked with green color. Otherwise, the point is marked with red color.

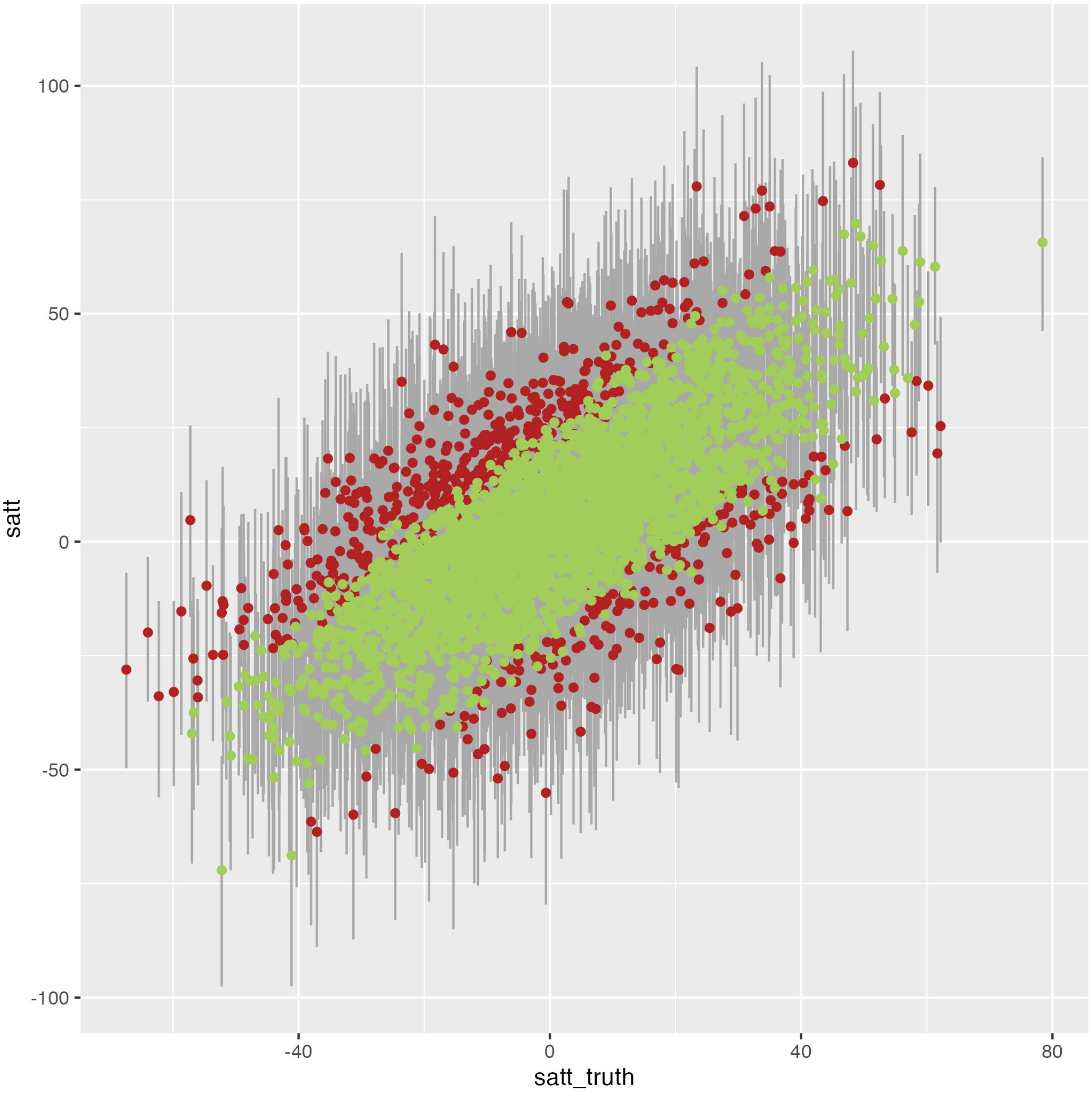

Figure 6.3 Scatter plot of estimated $SATT_{overall}$ via BART with **Method.1** against ground-truth $SATT_{overall}$

6.5 Visualization of BCF Performance

Figure 6.4 shows the boxplot of absolute biases of $SATT_{overall}$, $SATT_{year3}$ and $SATT_{year4}$ which are estimated by BCF.

Figure 6.5 shows the boxplot of uncertainty interval widths for $SATT_{overall}$, $SATT_{year3}$ and $SATT_{year4}$ which are estimated by BCF.

Figure 6.6 shows the scatter plot of estimated $SATT_{overall}$ via BCF against ground-truth $SATT_{overall}$, where grey vertical lines represent the uncertainty intervals of estimated $SATT_{overall}$. If the ground-truth $SATT_{overall}$ falls within the uncertainty intervals, the point is marked with green color. Otherwise, the point is marked with red color.

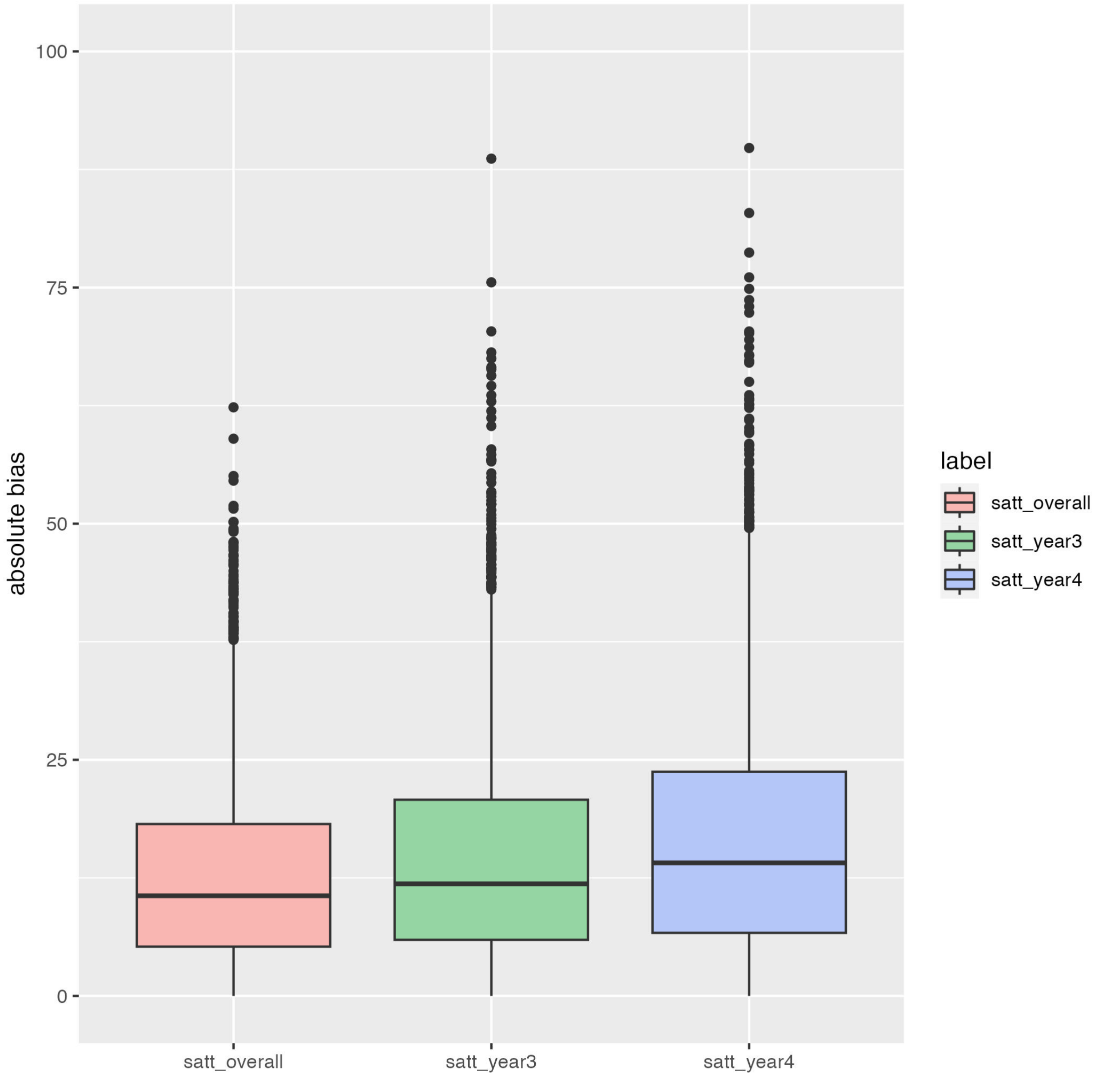

Figure 6.4 Boxplot of absolute biases of SATTs via BCF

6.6 DML with Year as an Additional Covariate

As stated in Section 5.5, we also tried the modified DML model, which included the year as an additional covariate in estimating $SATT_{overall}$. Their results are shown in the table below.

	DML	DML with year
RMSE	22.3636	19.2605
Uncertainty interval coverage rate	2.26 %	2.59 %
Average uncertainty interval width	1.0739	1.1153

Table 6.3 Table to compare performances of DML and DML with year.

Concerning $SATT_{overall}$, they have nearly identical performances of uncertainty interval coverage rates and mean uncertainty interval widths. However, DML with year as an additional

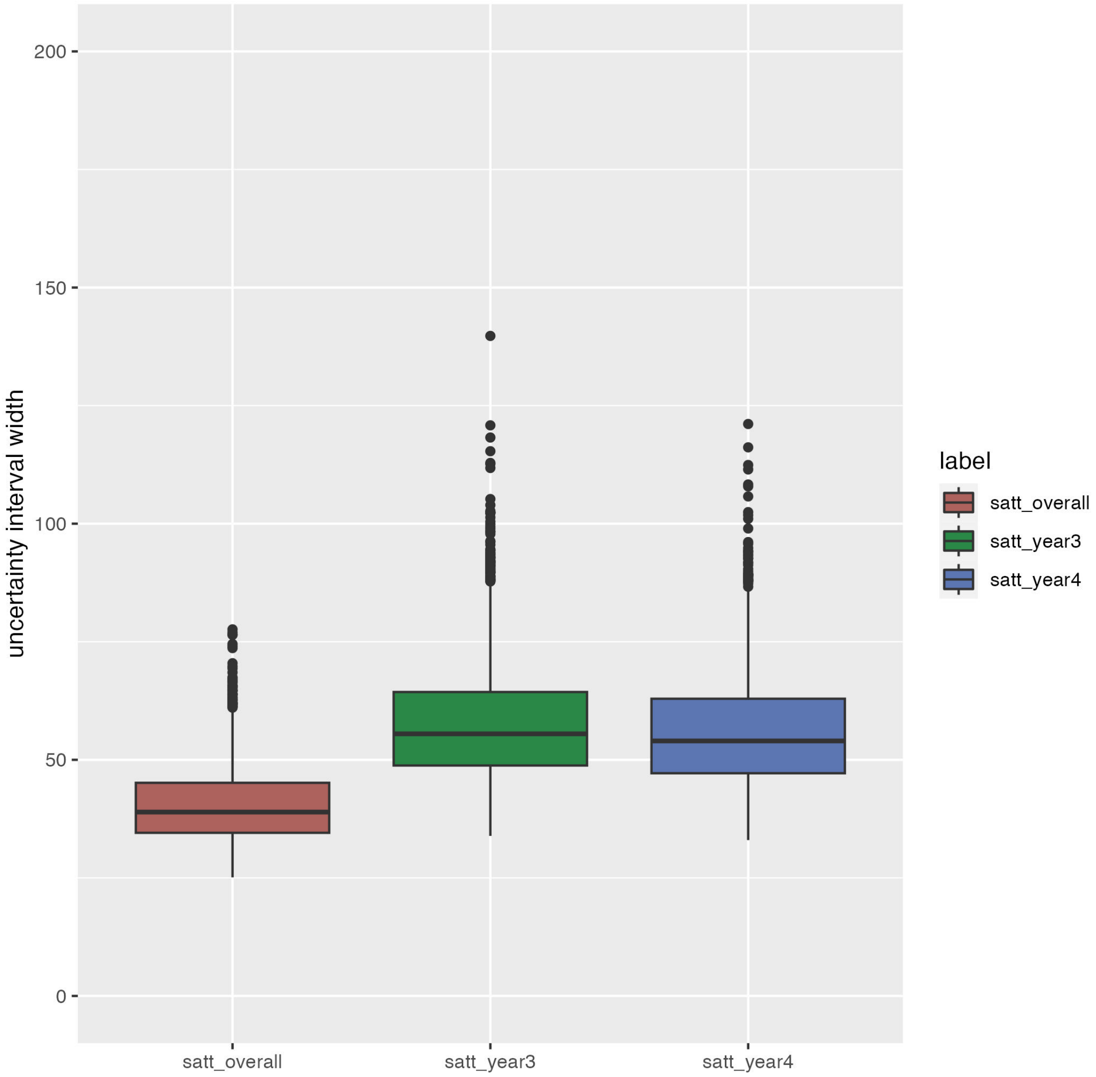

Figure 6.5 Boxplot of uncertainty interval widths of SATTs via BCF

covariate results in lower RMSE. It is consistent with how folds for $SATT_{overall}$ are created. We fold year 3 and year 4 data together if they have the same $id.practice$ value. As a result, it is natural to consider a model that uses year as an additional covariate to fit data from years 3 and 4. In the following discussions, we will use the implementation result of DML with year to estimate $SATT_{overall}$.

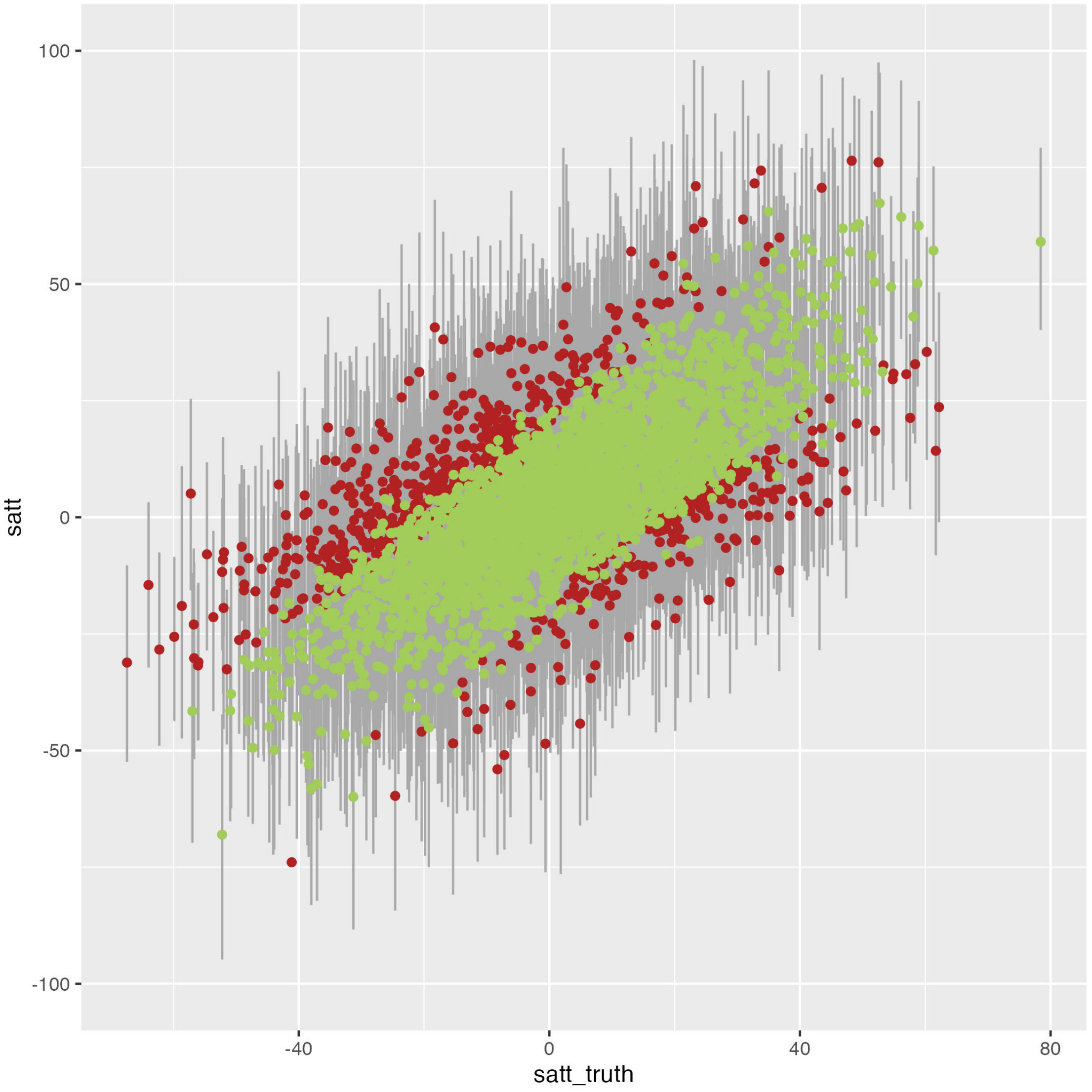

Figure 6.6 Scatter plot of estimated $SATT_{overall}$ via BCF against ground-truth $SATT_{overall}$

6.7 Visualization of DML Performance

Figure 6.7 shows the boxplot of absolute biases of $SATT_{overall}$, $SATT_{year3}$ and $SATT_{year4}$ which are estimated by double machine learning.

Figure 6.8 shows the boxplot of uncertainty interval widths for $SATT_{overall}$, $SATT_{year3}$ and $SATT_{year4}$ which are estimated by DML.

Figure 6.9 shows the scatter plot of estimated $SATT_{overall}$ via DML against ground-truth $SATT_{overall}$, where grey vertical lines represent the uncertainty intervals of estimated $SATT_{overall}$. If the ground-truth $SATT_{overall}$ falls within the uncertainty intervals, the point is marked with green color. Otherwise, the point is marked with red color.

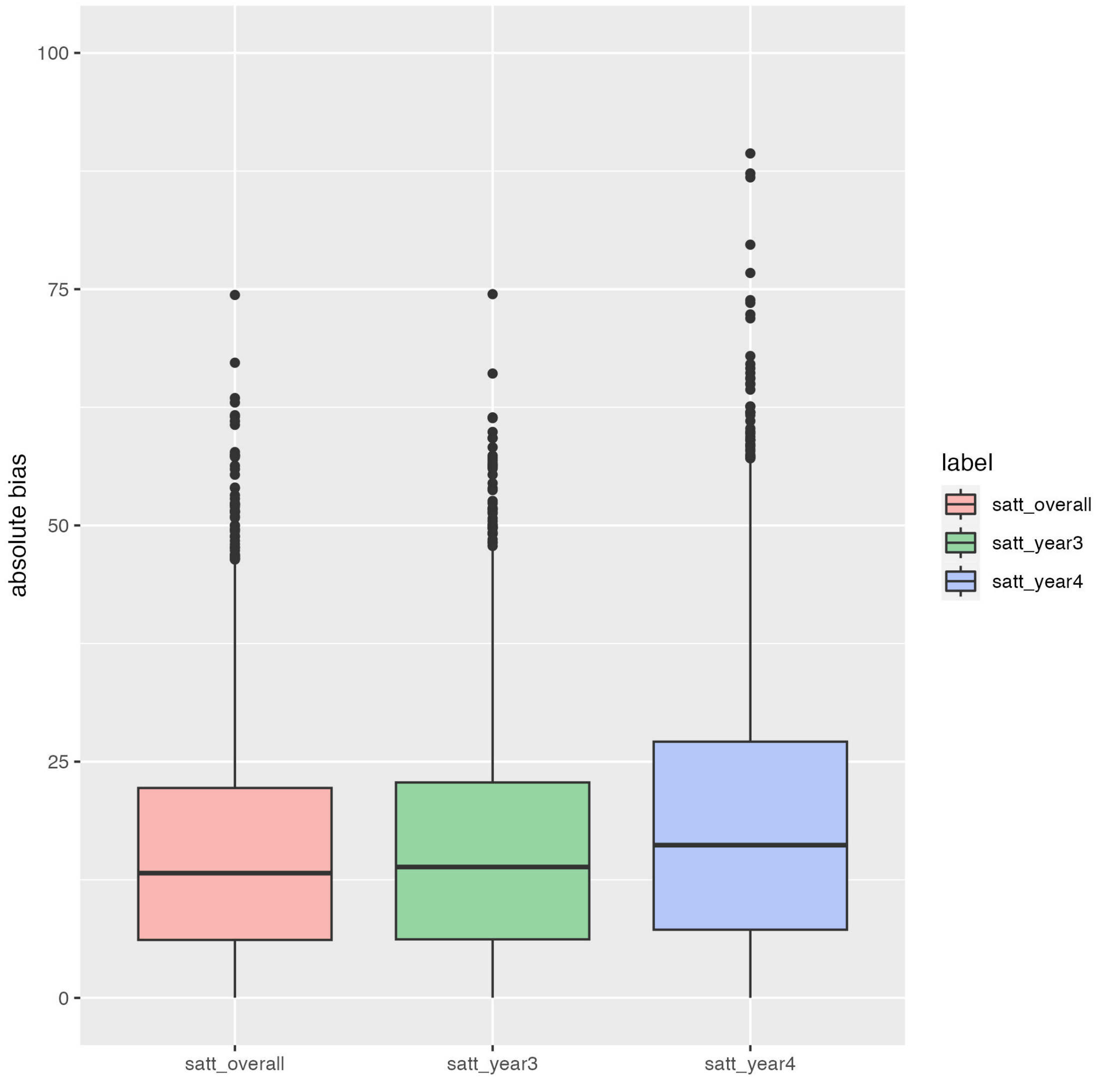

Figure 6.7 Boxplot of absolute biases of SATTs via double machine learning

6.8 Comparison of BART, BCF, and DML Performances

We draw three bar plots(Figure 6.10, 6.11 and 6.12) for the metrics RMSE, uncertainty interval coverage rate and average uncertainty interval width separately. Each plot compares the performances of BART, BCF, and DML implementations in terms of a specific metric.

In terms of RMSE, the performances of BART, BCF, and DML implementations all seem to have a good fit. Compared with the other two models, the uncertainty interval widths for DML implementation are significantly smaller. This results in a rather low uncertainty interval coverage rate. The BART model and BCF model are Bayesian methods, and the lower and upper bounds of uncertainty intervals are the 5% and 95% quantiles of the SATT statistics in posterior draws. Nevertheless, in the DML model, uncertainty intervals are calculated by applying Theroem 5. Com-

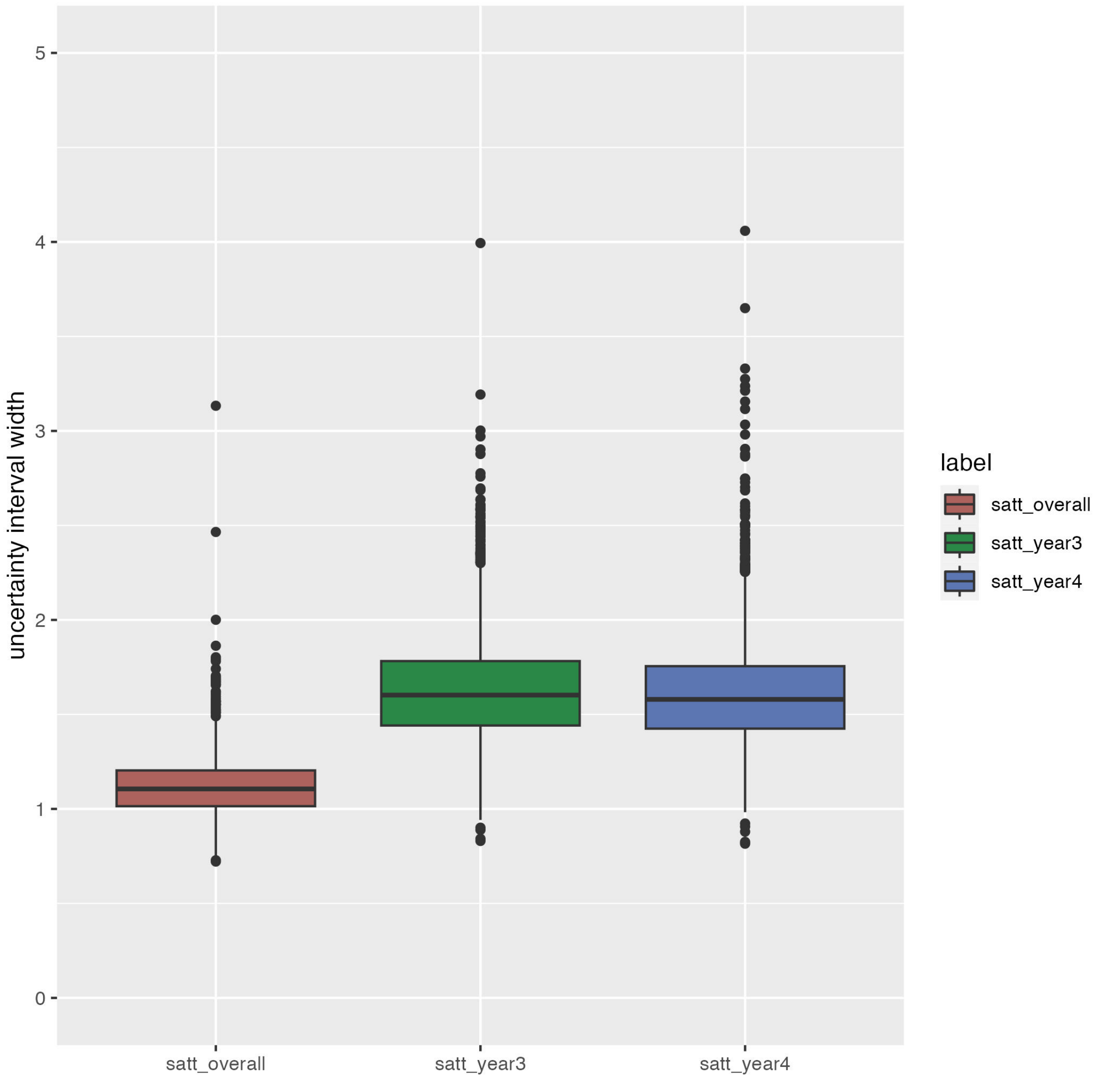

Figure 6.8 Boxplot of uncertainty interval widths of SATTs via double machine learning

pared to BART and BCF implementations, DML implementations result in significantly smaller uncertainty interval widths. This is why DML performs so poorly regarding uncertainty interval coverage rate.

BCF is a variant of BART model. As stated in Section 4.3, it tends to outperform the BART model in treatment effects estimation. The performance comparisons confirmed the statement. The BCF outperforms the other two model implementations in terms of the given metrics.

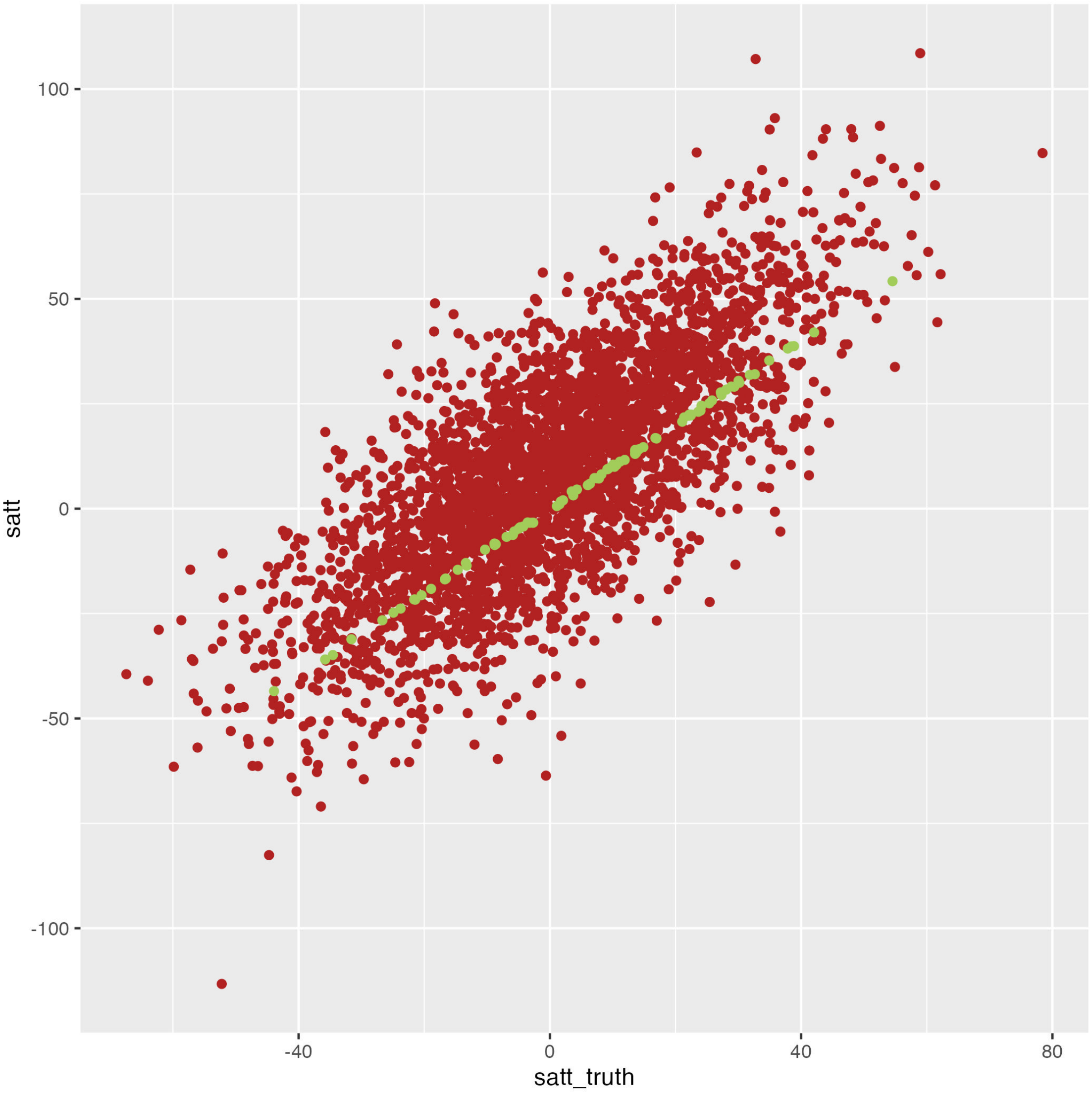

Figure 6.9 Scatter plot of estimated $SATT_{overall}$ via BCF against ground-truth $SATT_{overall}$

6.9 Heterogeneous Treatment Effects

In treatment effects analysis, it is important to investigate effect heterogeneity to find which groups are more or less likely to benefit from treatment. Subgroup analysis is a method for investigating heterogeneity. We calculate the treatment effects for each subgroup and see if they differ significantly from one another. Thus, the performances of model implementations concerning $SATT_{subgroup}$ are also crucial. The 2022 ACIC Data Challenge defined 12 subgroups and we calculate the average sample size for each subgroup by averaging the total number of patients within each subgroup across 3400 datasets and the whole intervention period. We plot the RMSE of a specific subgroup against the sample size of the subgroup and obtained Figure 6.13. We can infer from the plot that one model performs better for subgroups with larger sample sizes. BART-

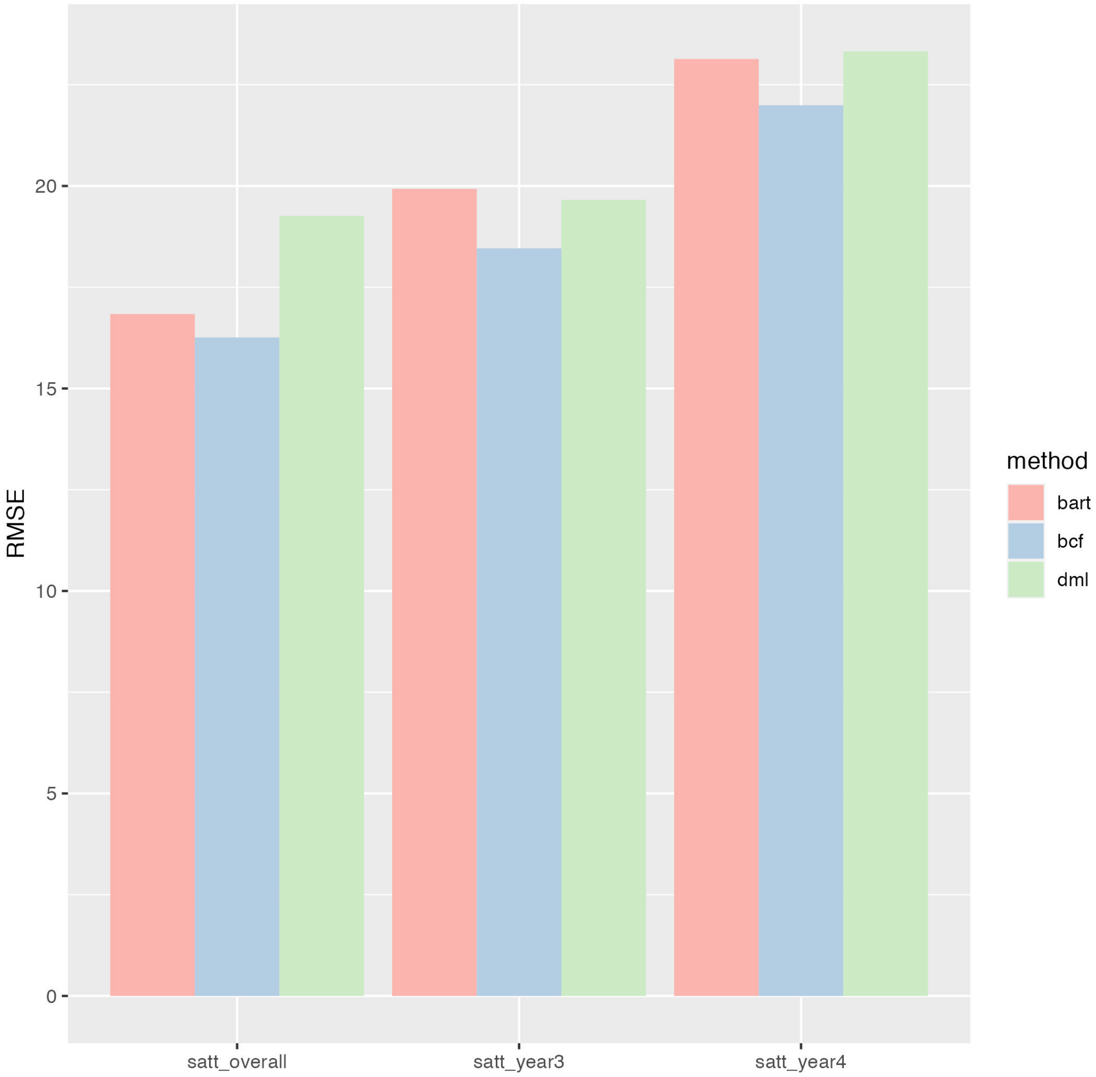

Figure 6.10 Comparison the performances of BART, BCF, and DML concerning RMSE

based models (BART and BCF) outperform DML concerning estimations of $SATT_{subgroup}$. This means that when dealing with treatment effect heterogeneity, we should prioritize BART-based models over DML.

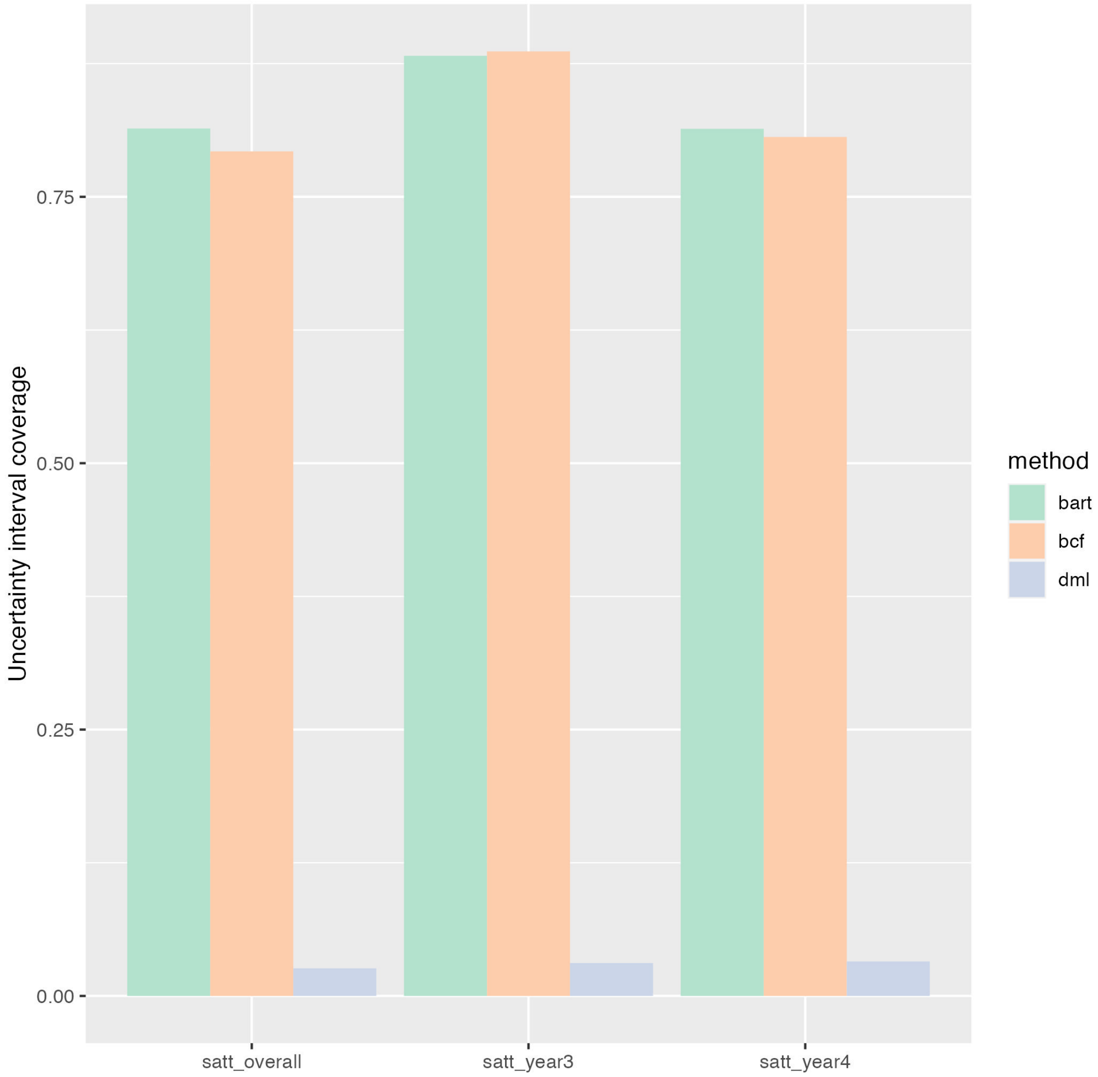

Figure 6.11 Comparison the performances of BART, BCF, and DML concerning uncertainty interval coverage rate

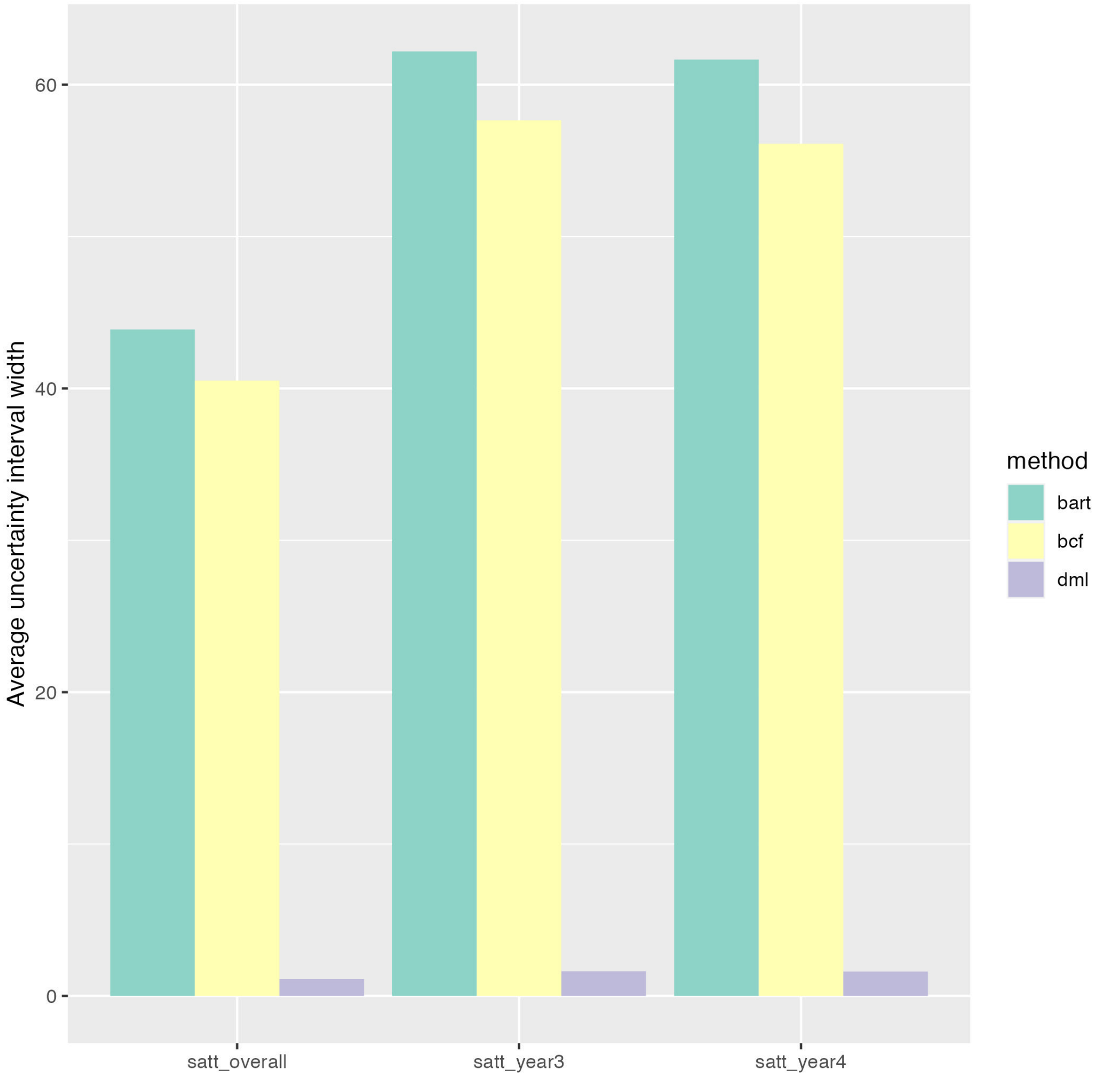

Figure 6.12 Comparison the performances of BART, BCF, and DML concerning average uncertainty interval width

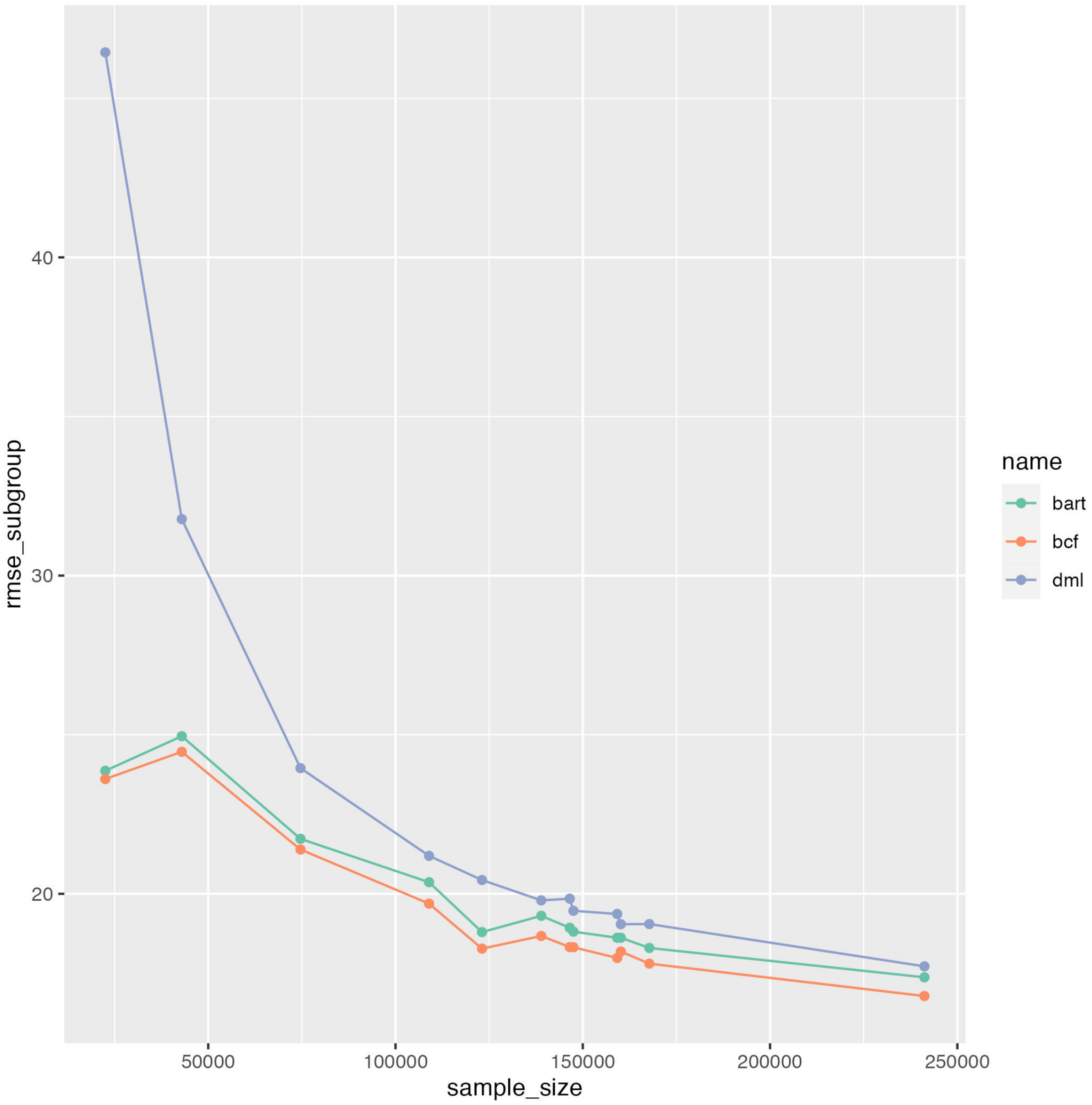

Figure 6.13 RMSE of 12 subgroups against the sample size of the 12 subgroups

Bibliography

[1] Bleich, Justin, et al. "Variable selection for BART: an application to gene regulation." (2014): 1750-1781.

[2] Tan, Yaoyuan Vincent, and Jason Roy. "Bayesian additive regression trees and the General BART model." Statistics in medicine 38.25 (2019): 5048-5069.

[3] Kapelner, Adam, and Justin Bleich. "bartMachine: Machine learning with Bayesian additive regression trees." arXiv preprint arXiv:1312.2171 (2013).

[4] Chipman, Hugh A., Edward I. George, and Robert E. McCulloch. "BART: Bayesian additive regression trees." (2010): 266-298.

[5] Butcher, Brandon David. "MCMC Diagnostics for Bayesian Additive Regression Trees and Methods for Flexible Modeling of Predictors." The University of Iowa, 2020.

[6] Sparapani, Rodney, Charles Spanbauer, and Robert McCulloch. "Nonparametric machine learning and efficient computation with Bayesian additive regression trees: the BART R package." Journal of Statistical Software 97 (2021): 1-66.

[7] Hahn, P. R., J. S. Murray, and C. M. Carvalho. "bcf: causal inference for a binary treatment and continuous outcome using Bayesian causal forests." R package version 1 (2019).

[8] Hahn, P. Richard, Jared S. Murray, and Carlos M. Carvalho. "Bayesian regression tree models for causal inference: Regularization, confounding, and heterogeneous effects (with discussion)." Bayesian Analysis 15.3 (2020): 965-1056.

[9] Carnegie, Nicole, Vincent Dorie, and Jennifer L. Hill. "Examining treatment effect heterogeneity using BART." Observational Studies 5.2 (2019): 52-70.

[10] Krantsevich, Nikolay, Jingyu He, and P. Richard Hahn. "Stochastic tree ensembles for estimating heterogeneous effects." arXiv preprint arXiv:2209.06998 (2022).

[11] Caron, Alberto, Gianluca Baio, and Ioanna Manolopoulou. "Shrinkage Bayesian Causal Forests for heterogeneous treatment effects estimation." Journal of Computational and Graphical Statistics 31.4 (2022): 1202-1214.

[12] Chernozhukov, Victor, et al. "Double/debiased/neyman machine learning of treatment effects." American Economic Review 107.5 (2017): 261-265.

[13] Bach, Philipp, et al. "DoubleML–An Object-Oriented Implementation of Double Machine Learning in R." arXiv preprint arXiv:2103.09603 (2021).

[14] Chernozhukov, V., et al. "Double Machine Learning for Treatment and Causal Parameters, 2016." arXiv preprint arXiv:1608.00060.

[15] Chernozhukov, Victor, et al. "Double/debiased machine learning for treatment and structural parameters." (2018): C1-C68.

[16] Nowacki, Amy S., et al. "Adding propensity scores to pure prediction models fails to improve predictive performance." PeerJ 1 (2013): e123.

[17] Pratola, Matthew T., et al. "Parallel Bayesian additive regression trees." Journal of Computational and Graphical Statistics 23.3 (2014): 830-852.

[18] Felton, Chris. Chernozhukov Et Al. on Double / Debiased Machine Learning - Princeton. https://scholar.princeton.edu/sites/default/files/bstewart/files/felton.chern_.slides.20190318.pdf.

[19] Wager, Stefan. Stats 361: Causal Inference - Stanford University. https://web.stanford.edu/ swager/stats361.pdf.

[20] ACIC Competition, https://acic2022.mathematica.org/.

[21] Yao, Liuyi, et al. "A survey on causal inference." ACM Transactions on Knowledge Discovery from Data (TKDD) 15.5 (2021): 1-46.

A Appendix

The following R scripts are attached in the submission folder.

➢ **p_score.r**: Implementation of Algorithm 5.

➢ **bart_method1.r**: Implementation of Algorithm 6.

➢ **bart_method2.r**: Implementation of Algorithm 8.

➢ **bcf_test.r**: Implementation of Algorithm 12.

➢ **dml_test.r**: Implementation of Algorithm 14.

➢ **dml_time.r**: Estimation of $SATT_{overall}$ by double machine learning with year as an additional covariate.

➢ **dml_subgroup.r**: Implementation of Algorithm 21

➢ **evaluation.r**: Functions about metrics RMSE and uncertainty interval coverage rate.