

## Distribution Agreement

In presenting this thesis or dissertation as a partial fulfillment of the requirements for an advanced degree from Emory University, I hereby grant to Emory University and its agents the non-exclusive license to archive, make accessible, and display my thesis or dissertation in whole or in part in all forms of media, now or hereafter known, including display on the world wide web. I understand that I may select some access restrictions as part of the online submission of this thesis or dissertation. I retain all ownership rights to the copyright of the thesis or dissertation. I also retain the right to use in future works (such as articles or books) all or part of this thesis or dissertation.

DocuSigned by:  
Signature:   
C153E11ACA66447...

Sohail Nizam  
Name

4/6/2023 | 11:28 AM EDT  
Date


**Title** Methods for Improving the Interpretability and Evaluation of Machine Learning Models and Decision Making Systems

**Author** Sohail Nizam

**Degree** Doctor of Philosophy

**Program** Biostatistics


**Approved by the Committee**

DocuSigned by:  
  
98BCFD2EB08A4C8...


David Benkeser  
*Advisor*

DocuSigned by:  
  
02B73DA5C4634D9...

Razieh Nabi  
*Committee Member*

DocuSigned by:  
  
40C06BE10E754E2...

Benjamin Risk  
*Committee Member*

DocuSigned by:  
  
50B5C2978F4746A...

Abeer Sarker  
*Committee Member*

**Accepted by the Laney Graduate School**

Kimberly Jacob Arriola, PhD, MPH  
*Dean, James T. Laney School of Graduate Studies*

Methods for Improving the Interpretability and Evaluation of Machine Learning  
Models and Decision Making Systems

By

Sohail Nizam  
B.A., University of Chicago, IL, 2018

Advisor: David Benkeser, Ph.D.

An abstract of  
A dissertation submitted to the Faculty of the  
James T. Laney School of Graduate Studies of Emory University  
in partial fulfillment of the requirements for the degree of  
Doctor of Philosophy  
in Biostatistics and Bioinformatics  
2023

## Abstract

The ability to interpret and evaluate machine learning models is of great importance, particularly when such models will be used as the foundation for decision making systems. Practitioners must have a high degree of certainty about the level of performance that their models will achieve and be able to explain decisions rendered by them. This work aims to improve capabilities in both of these areas.

In the first chapter, we address the problem of model evaluation in binary classification settings. There, the Area Under the Precision Recall Curve (AUPRC) is often of interest in settings with extreme class imbalance. Estimation of metrics like AUPRC is often paired with cross-validation. We formally define the Cross-Validated AUPRC (CVAUPRC), a data-adaptive target parameter, and provide closed-form inference for its non-parametric maximum likelihood estimator. Additionally, we propose a more efficient estimation strategy based on nested cross-validation that offers dramatic improvement in situations with extreme class imbalance.

In the second chapter, we introduce a method for building an interpretable representation of the Highly Adaptive Lasso (HAL). HAL is a machine learning method that has been shown to have predictive performance on par with state-of-the-art algorithms and can be represented as a non-recursive partitioning of the feature space. We propose a method for mapping this partitioning implied by HAL to a recursive partitioning, which then allows for the representation of HAL as a decision tree. We refer to this post-hoc method for interpretability as Highly Adaptive Regression Trees.

In the third chapter, we address the problem of interpretable Conditional Average Treatment Effect (CATE) and, by extension, Optimal Treatment Policy (OTP) estimation. Many machine learning-based frameworks for CATE estimation have been proposed. However, few of these methods are interpretable, and those that are often suffer in terms of performance. We extend HART's capabilities and build on existing Meta-Learning algorithms to produce CATE and OTP estimates which can be represented as trees. We introduce this method for settings with an arbitrary number of treatment arms. We provide regret rates for the proposed methods and show that they outperform popular methods, both interpretable and not.

Methods for Improving the Interpretability and Evaluation of Machine Learning  
Models and Decision Making Systems

By

Sohail Nizam  
B.A., University of Chicago, IL, 2018

Advisor: David Benkeser, Ph.D.

A dissertation submitted to the Faculty of the  
James T. Laney School of Graduate Studies of Emory University  
in partial fulfillment of the requirements for the degree of  
Doctor of Philosophy  
in Biostatistics and Bioinformatics  
2023

## Acknowledgments

I'd like to thank my advisor, Dr. David Benkeser for his guidance, support, and the endless time he sat patiently answering my random questions about causal inference and semi-parametric efficiency theory. I'd also like to thank my family. My dad, Azhar, is the first and best teacher I've ever had. From times tables to probability and calculus, everything I know and love about math traces back to time spent with him. My mom, Janet is the kindest and most supportive person on the planet. If I can call myself a good person, it's probably because of her. My sister, Zainab has always been someone to look up to and beat me to finishing the PhD by a few months. My girlfriend, Tomris, reminds me that work isn't everything and that its important to live an exciting life. She's managed to make Law School look easy while still taking me all over the world. Finally, I'd like to thank my dog, Rosie who is telling me to wrap it up so we can go on a walk.

# Contents

<b>1</b>	<b>Estimation and inference for cross-validated area under the precision recall curve</b>	<b>1</b>
1.1	Introduction . . . . .	1
1.2	Notation and definition of target parameters . . . . .	4
1.2.1	AUPRC as a target parameter . . . . .	4
1.2.2	Cross-validated AUPRC as a target parameter . . . . .	4
1.3	Estimation and inference . . . . .	6
1.3.1	Estimation and inference for AUPRC . . . . .	6
1.3.2	Estimation and inference for cross-validated AUPRC . . . . .	7
1.3.3	Improved estimation and inference for cross-validated AUPRC	8
1.4	Results . . . . .	10
1.4.1	Simulations . . . . .	10
1.4.2	Data analysis . . . . .	11
1.5	Discussion . . . . .	18
<b>2</b>	<b>Highly Adaptive Regression Trees: A post-hoc method for interpreting the Highly Adaptive Lasso</b>	<b>19</b>
2.1	Introduction . . . . .	19
2.2	Highly adaptive lasso . . . . .	21
2.3	Notation . . . . .	22

2.4	From HAL to HART . . . . .	22
2.5	Examples . . . . .	26
2.5.1	Scenario (i): naturally recursive partitioning . . . . .	27
2.5.2	Scenario (ii): non-recursive partitioning, minimal redundancy . . . . .	28
2.5.3	Scenario (iii): non-recursive partitioning, minimal and non-minimal redundancy . . . . .	29
2.6	Highly adaptive regression trees . . . . .	30
2.6.1	Growing trees . . . . .	30
2.6.2	Policies . . . . .	32
2.6.3	Sub-HARTs and Prediction Smoothing . . . . .	34
2.7	Algorithm Complexity . . . . .	36
2.7.1	HART Complexity . . . . .	36
2.7.2	Bin and Aggregate Predictions Complexity . . . . .	37
2.8	Data Analysis . . . . .	37
2.8.1	Performance . . . . .	38
2.8.2	Trees . . . . .	38
2.8.3	Complexity . . . . .	41
2.9	Discussion . . . . .	42

**3 Highly Adaptive Treatment Trees: interpretable estimation of heterogeneous treatment effects and treatment policies 43**

3.1	Introduction . . . . .	43
3.2	Problem Setup and Estimation Frameworks . . . . .	45
3.2.1	Setup . . . . .	45
3.2.2	Meta-Learning With Two Treatment Arms . . . . .	46
3.2.3	Meta-Learning with Multiple Treatment Arms . . . . .	48
3.2.4	Causal Trees and Forests . . . . .	49
3.3	The Highly Adaptive Lasso . . . . .	50



3.3.1	Background . . . . .	50
3.3.2	Highly Adaptive Regression Trees . . . . .	51
3.4	CATE Estimation with HAL . . . . .	52
3.4.1	Theoretical Guarantees for S and DR learners . . . . .	52
3.4.2	Tree Representations . . . . .	54
3.5	Simulations . . . . .	55
3.5.1	Data generating process . . . . .	55
3.5.2	Evaluation . . . . .	57
3.5.3	Linear CATE Results . . . . .	57
3.5.4	Polynomial CATE Results . . . . .	57
3.5.5	Sinusoidal CATE Results . . . . .	59
3.6	Data Analysis . . . . .	60
3.7	Discussion . . . . .	61
<b>Appendix A Chapter 1 Supplementary Material</b>		<b>63</b>
A.1	Notation . . . . .	63
A.2	Theorem 1.3.1 Proof . . . . .	64
A.3	Theorem 1.3.1 Condition Examination . . . . .	73
A.4	Theorem 1.3.2 Proof . . . . .	75
A.5	Supplementary Figures . . . . .	76
<b>Appendix B Chapter 3 Supplementary Material</b>		<b>89</b>
B.1	Theorem 3.2.1 Proof . . . . .	89
B.2	Theorem 3.4.1 Proof . . . . .	90
B.3	Theorem 3.4.2 Proof . . . . .	93
B.4	Further 2-Arm Simulation Results . . . . .	97
B.5	Multi-Arm Simulation Results . . . . .	97
<b>Bibliography</b>		<b>102</b>

# List of Figures

- 1.1 Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Random Forest on data with **no imbalance**. 5 and 10 on the x-axis indicate number of outer folds  $K$ . In each case results are based on 1000 simulations. . . . . 12
- 1.2 Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Random Forest on data with **large imbalance**. 5 and 10 on the x-axis indicate number of outer folds  $K$ . In each case results are based on 1000 simulations. . . 13
- 1.3 Coverage probabilities for 95% confidence intervals. LR and RF stand for Logistic Regression and Random Forest respectively. Probabilities represent the proportion of 95% confidence intervals covering the true value out of 1000 estimation procedures. . . . . 14
- 2.1 Left: Illustration of partitioning of the feature space according to the HAL basis expansion. The sets of  $\hat{\beta}$  (excluding the intercept) correspond to those in equation (2.1) that fall in each region. Right: An illustrative HAL partitioning, when  $\hat{\beta}_{X_{11}} = \hat{\beta}_{X_{21}} = \hat{\beta}_{X_{22}} = \hat{\beta}_{X_{12}, X_{22}} = 0$ , which results in a non-recursive partitioning of the feature space. To represent this partitioning as a decision tree, we must determine a parsimonious recursive structure for the L-shaped left-most region. . . . . 24

2.2	Decision tree representation of the full feature space partitioning implied by the HAL basis expansion. . . . .	25
2.3	Two possible recursive partitions to represent the non-recursive partition implied by the HAL model when $\hat{\beta}_{X_{11}} = \hat{\beta}_{X_{21}} = \hat{\beta}_{X_{22}} = \hat{\beta}_{X_{12}, X_{22}} = 0$ . .	26
2.4	Left: The feature space partitioned according to the HAL fit with all $\hat{\beta} \neq 0$ . These partitions will be described by values from $X$ . Right: Feature space with $X_{11}$ describing the first partition. . . . .	27
2.5	An illustrative HAL partitioning, when $\hat{\beta}_{X_{11}} = \hat{\beta}_{X_{12}} = \hat{\beta}_{X_{22}} = \hat{\beta}_{X_{11}, X_{21}} = 0$ , which results in a non-recursive partitioning of the feature space. .	30
2.6	Two recursive partitionings to represent the non-recursive partitioning implied by HAL when $\hat{\beta}_{X_{11}} = \hat{\beta}_{X_{12}} = \hat{\beta}_{X_{22}} = \hat{\beta}_{X_{11}, X_{21}} = 0$ . The left partitioning is not minimally redundant. The right partitioning is minimally redundant. . . . .	31
2.7	Left: CART fit to Breast Cancer dataset. Right: HART built from HAL fit to Breast Cancer dataset. Feature space is restricted to subjects aged 20-29 having tumors greater than 5mm in diameter. Algorithm 2 was applied with $Q = 3$ . . . . .	40
2.8	HART representing HAL fit to Breast Cancer data. Algorithm 2 has been applied with $Q = 2$ bins, equivalent to thresholding the predicted probability of recurrence at 0.5. . . . .	41
3.1	Simulation results under the Linear setting. The DR-Learner with HAL used Super Learning to construct nuisance parameter estimates. Results at each sample size are averaged over 1000 replications. . . .	58
3.2	Simulation results under the Polynomial setting. The DR-Learner with HAL used Super Learning to construct nuisance parameter estimates. Results at each sample size are averaged over 1000 replications. . . .	58

3.3	Simulation results under the Sinusoidal setting with lower variation norm. The DR-Learner with HAL used Super Learning to construct outcome regression estimates. All propensity score estimates were constructed with simple means. At $n = 50$ , Causal Forest showed abnormally low performance and is omitted from the plot to keep other learners distinguishable. Results at each sample size are averaged over 1000 replications. . . . .	59
3.4	Simulation results under the Sinusoidal setting with higher variation norm. The DR-Learner with HAL used Super Learning to construct outcome regression estimates. All propensity score estimates were constructed with simple means. Results at each sample size are averaged over 1000 replications. . . . .	60
3.5	Tree representing the effect of receiving a mailer with a message meant to assuage fears about ballot secrecy. M-dev age represent the number of years above the median study age. The terminal nodes represent differences in estimated counterfactual probability of voting under treatment versus control. . . . .	61
A.1	Coverage probabilities of 95% confidence intervals for the AUPRC. Probabilities are calculated as the proportion of times the interval covered the true parameter out of 5000 iterations. . . . .	74
A.2	Scaled absolute bias of the NPMLE of the AUPRC at several sample sizes. Results at a given sample size are averaged over 5000 iterations.	75
A.3	Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Random Forest on data with <b>no imbalance</b> . 5 and 10 on the x-axis indicate number of outer folds $K$ . In each case results are based on 1000 simulations. . . . .	77

A.4	Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Random Forest on data with <b>imbalance</b> . 5 and 10 on the x-axis indicate number of outer folds $K$ . In each case results are based on 1000 simulations. . . . .	78
A.5	Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Random Forest on data with <b>large imbalance</b> . 5 and 10 on the x-axis indicate number of outer folds $K$ . In each case results are based on 1000 simulations. . .	79
A.6	Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Logistic Regression on data with <b>no imbalance</b> . 5 and 10 on the x-axis indicate number of outer folds $K$ . In each case results are based on 1000 simulations. . .	80
A.7	Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Logistic Regression on data with <b>imbalance</b> . 5 and 10 on the x-axis indicate number of outer folds $K$ . In each case results are based on 1000 simulations. . . . .	81
A.8	Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Logistic Regression on data with <b>large imbalance</b> . 5 and 10 on the x-axis indicate number of outer folds $K$ . In each case results are based on 1000 simulations. .	82
A.9	Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Random Forest on data with <b>no imbalance</b> . 5 and 10 on the x-axis indicate number of outer folds $K$ . In each case results are based on 1000 simulations. Restricted to $n = 200, 4000, 6000$ for easier viewing. . . . .	83

A.10 Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Random Forest on data with <b>imbalance</b> . 5 and 10 on the x-axis indicate number of outer folds $K$ . In each case results are based on 1000 simulations. Restricted to $n = 200, 4000, 6000$ for easier viewing. . . . .	84
A.11 Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Random Forest on data with <b>large imbalance</b> . 5 and 10 on the x-axis indicate number of outer folds $K$ . In each case results are based on 1000 simulations. Restricted to $n = 200, 4000, 6000$ for easier viewing. . . . .	85
A.12 Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Logistic Regression on data with <b>no imbalance</b> . 5 and 10 on the x-axis indicate number of outer folds $K$ . In each case results are based on 1000 simulations. Restricted to $n = 200, 4000, 6000$ for easier viewing. . . . .	86
A.13 Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Logistic Regression on data with <b>imbalance</b> . 5 and 10 on the x-axis indicate number of outer folds $K$ . In each case results are based on 1000 simulations. Restricted to $n = 200, 4000, 6000$ for easier viewing. . . . .	87
A.14 Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Logistic Regression on data with <b>large imbalance</b> . 5 and 10 on the x-axis indicate number of outer folds $K$ . In each case results are based on 1000 simulations. Restricted to $n = 200, 4000, 6000$ for easier viewing. . . . .	88
B.1 Accuracy, Sensitivity, and Specificity results for learning the learning the binary optimal treatment rule in the Linear CATE setting. . . . .	97

B.2	Accuracy, Sensitivity, and Specificity results for learning the learning the binary optimal treatment rule in the Polynomial CATE setting. .	98
B.3	Accuracy, Sensitivity, and Specificity results for learning the learning the binary optimal treatment rule in the Sinusoidal CATE setting with $C = 1$ . At $n = 50$ , Causal Forest showed abnormally low performance and is omitted from the plot to keep other learners distinguishable. .	99
B.4	Accuracy, Sensitivity, and Specificity results for learning the learning the binary optimal treatment rule in the Sinusoidal CATE setting with $C = 10$ . . . . .	100
B.5	Multi-arm simulation results in the Linear setting. A simple mean estimator was used to estimate the propensity score for the DR-Learner with HAL. The bottom left and bottom right panels show $L_2$ norms for $\hat{\psi}(1   v)$ and $\hat{\psi}(2   v)$ respectively. . . . .	100
B.6	Multi-arm simulation results in the polynomial setting. A simple mean estimator was used to estimate the propensity score for the DR-Learner with HAL. The bottom left and bottom right panels show $L_2$ norms for $\hat{\psi}(1   v)$ and $\hat{\psi}(2   v)$ respectively. . . . .	101

# List of Tables

1.1	UCI dataset characteristics. $N$ is sample size, $p$ is number of features.	15
1.2	Analysis results using $n = 250$ samples to train and evaluate learners. Remaining observations were used to calculate the true CVAUPRC values. Est. is the average parameter estimate, Width is the average 95% confidence interval width, and Cov. is the proportion of 95% confidence intervals covering the truth. Each entry is a result averaged over 1000 iterations. . . . .	16
1.3	Analysis results using $n = 1000$ samples to train and evaluate learners. Remaining observations were used to calculate the true CVAUPRC values. Est. is the average parameter estimate, Width is the average 95% confidence interval width, and Cov. is the proportion of 95% confidence intervals covering the truth. Each entry is a result averaged over 1000 iterations. . . . .	17
2.1	Notation used throughout the paper to describe aspects of feature space partitioning and tree creation. . . . .	23
2.2	UCI dataset characteristics. $N$ is sample size, $p$ is number of features.	38



2.3	10-fold cross-validated metrics are reported for Breast Cancer, Cardio, and Drugs. 3-fold cross validated metrics are reported for Wine. For CART, Random Forest, and XGBoost, we carried out grid searches over 10 tuning parameter settings. Results correspond to models with the highest performing tuning parameters. Computations were carried out on a High Performance Computing cluster. . . . .	39
2.4	Breast Cancer dataset feature names and descriptions. . . . .	39

# List of Algorithms

1	HART . . . . .	31
2	Bin Predictions . . . . .	35
3	Aggregate Predictions . . . . .	35
4	CATE Tree Construction . . . . .	54
5	Treatment Policy Tree Construction . . . . .	55

# Chapter 1

## Estimation and inference for cross-validated area under the precision recall curve

### 1.1 Introduction

Suppose we are given a function  $\psi$  that maps from a covariate space  $\mathcal{X}$  to the unit interval. For example,  $\psi$  could be a clinical risk score, where  $X \in \mathcal{X}$  represents a vector of underlying demographic and clinical information and  $\psi(X)$  represents predicted probability of disease. Let  $Y$  denote the binary outcome of interest, where we use *case* to refer to observations with  $Y = 1$  and *non-case* to refer to those with  $Y = 0$ . A binary classifier could be formed from  $\psi$ , for example by classifying individuals with covariate values  $x$  as a case if  $\psi(x) > z$  and all others as non-cases. Many metrics are available for evaluating the performance of  $\psi$ , either as a predicted probability that  $Y = 1$  or as the basis for a binary classifier. For example, the Accuracy of a classifier based on  $\psi$  is  $P(\psi(X) > z, Y = 1) + P(\psi(X) \leq z, Y = 0)$ , which describes the proportion of “correct” predictions that the classifier makes. Alternatively, Area Under the Receiver

Operating Characteristic Curve (AUC) =  $P(\psi(X_1) > \psi(X_2) \mid Y_1 = 1, Y_2 = 0)$  for two independent observations  $(X_1, Y_1), (X_2, Y_2)$ , which describes the probability that a randomly-selected case has a higher predicted risk than a randomly-selected control. AUC can also be represented as the area under a parametric curve, plotting the Recall  $P(\psi(X) > z \mid Y = 1)$  and the False Positive Rate  $P(\psi(X) > z \mid Y = 0)$  as a function of  $z$ . The AUC may provide a more reasonable account of model performance than Accuracy in cases with class imbalance [16]. This is because, with large class imbalance, high Accuracy can be achieved by simply predicting the majority class with high probability. While the AUC does offer some improvement in this case, its dependence on the true negative observations through the False Positive Rate may still yield overly optimistic model assessments in cases where  $P(Y = 0) \gg P(Y = 1)$  [34]. In these settings, many researchers prefer an alternative metric, the Area Under the Precision Recall Curve (AUPRC). AUPRC is similar to AUC in that it can be represented as the area under a parametric curve of Precision  $P(Y = 1 \mid \psi(X) > z)$  and Recall as a function of  $z$ . However, the AUC parameter does not depend on the marginal distribution of  $Y$ , and thus may be overly optimistic about model performance in settings with case imbalance. In these settings, AUPRC may provide a more balanced view of tradeoffs of using  $\psi$  as a classifier.

In many settings, we need to develop  $\psi$  and evaluate its performance using a single data set. For this task we use cross-validation. For example,  $K$ -fold cross-validation involves splitting the data into  $K$  folds, each comprised of a training sample and a validation sample. For each  $k = 1, \dots, K$ , we develop a function  $\psi_{n,k}$  using the  $k$ -th training sample, and we evaluate  $\psi_{n,k}$  by calculating our chosen metric using the  $k$ -th validation sample. We then average the  $K$  estimates of performance of the  $K$  models. A statistical framework for evaluating such estimators is presented in [52]. In this paper, we apply this framework to the Cross-Validated AUPRC (CVAUPRC) parameter. We formally define this parameter, derive the influence function of its nonparametric

maximum likelihood estimator (NPMLE), and establish weak convergence of the NPMLE. Together these results yield closed-form, asymptotically justified inference for CV-AUPRC, including confidence intervals about the performance of a given model and hypothesis tests comparing the performance of two or more models. While these approaches are justified in large samples, we argue that they may perform poorly in settings with small samples and/or high case imbalance. Because this latter situation specifically motivates the use of AUPRC as an evaluation metric, we are motivated to explore alternative estimation strategies.

We then note that, while use of the AUPRC is most advantageous in situations where the negative class has a large majority, estimation of the CVAUPRC could be extremely difficult in that exact situation. The reliance of the CV estimator on  $K$  validation samples of approximate size  $n/K$  to estimate the the Recall could prove problematic if the data already contain few cases. This issue could be especially prominent in small samples.

To address this issue, we also propose an estimation strategy based on nested cross-validation wherein we use the full training data within a fold both to build the classification model and evaluate it by calculating the AUPRC. Without correction, this strategy will yield biased estimates of model performance. Thus we examine the form of this bias and use it to construct a correction term using the data in the validation samples. We call this final estimate incorporating the correction term the CV One-Step (CV-OS) estimator.

We evaluate and compare the performances of both the CV and CV-OS estimators through simulation in scenarios with varying degrees of class imbalance and several sample sizes. These simulations verify the asymptotic results of both estimators, and demonstrate that the CV-OS can offer dramatic improvement in scenarios with class imbalance and/or small samples. We also apply both methods to the analysis of real data, which corroborates the results of the simulations.

## 1.2 Notation and definition of target parameters

### 1.2.1 AUPRC as a target parameter

Suppose we observe  $n$  independent realizations of the random variable  $O = (X, Y) \sim P_0$  where  $P_0$  is an element of nonparametric model  $\mathcal{M}$ ,  $X \in \mathcal{X}$  is a vector of predictors and  $Y \in \{0, 1\}$  is the outcome of interest. Suppose we have access to a prediction function  $x \mapsto \psi(x)$  for  $x \in \mathcal{X}$ . Our goal is to evaluate the performance of a given function  $\psi$  as a risk score by calculating the AUPRC. We can represent the AUPRC of  $\psi$  as a functional  $\Phi : \mathcal{M} \rightarrow [0, 1]$  of the data generating distribution. Under sampling from a given distribution  $P \in \mathcal{M}$ , the AUPRC of  $\psi$  is

$$\Phi_\psi(P) = \int P(Y = 1 \mid \psi(X) > z) dF_{\psi(X)|Y=1}(z) . \quad (1.1)$$

From equation (1.1), we see that AUPRC depends on two functionals of the distribution  $P$ : the precision  $P(Y = 1 \mid \psi(X) > z)$  and the complement of the recall  $F_{\psi(X)|Y=1}(z) := P(\psi(X) \leq z \mid Y = 1)$ . We use the zero subscript to denote the value of these functionals under  $P_0$  and define  $\phi_{\psi,0} := \Phi_\psi(P_0)$  as the true AUPRC of  $\psi$  under sampling from the true distribution of the data  $P_0$ .

### 1.2.2 Cross-validated AUPRC as a target parameter

Rather than always having access to a pre-trained  $\psi$  developed on external data, it is common to have a single data set that must be used to build and evaluate  $\psi$ . We denote by  $\Psi : \mathcal{M} \rightarrow \boldsymbol{\psi}$  a method for training a prediction algorithm where  $\boldsymbol{\psi}$  is the space of functions mapping  $\mathcal{X}$  to  $[0, 1]$ . Thus, for example, if  $P_n$  denotes the empirical distribution of  $(O_1, \dots, O_n)$  then  $\Psi(P_n)$  denotes the prediction algorithm trained using  $(O_1, \dots, O_n)$ . A naive strategy would be to obtain  $\Psi(P_n)$  using data  $(O_1, \dots, O_n)$  and then evaluate  $\Psi(P_n)$  by estimating the AUPRC using the *same data*. Such an estimate

of  $\Phi_{\Psi(P_n)}(P_0)$  would likely be biased and may give an overly optimistic sense of  $\Psi(P_n)$ 's performance.

This motivates the use of cross-validation. Cross-validation allows us to estimate performance in a way that is generalizable to *new* data from the same distribution. We focus on a  $K$ -fold cross-validation procedure [23], though our framework also applies to other forms of cross-validation such as leave-one-out [46, 3, 23] and leave- $p$ -out [45]. Consider a procedure in which we partition the data into  $K$  mutually exclusive blocks of approximately equal size. We then define  $K$  folds, where each fold consists of a training set that includes  $K - 1$  blocks of data and a validation set that includes the remaining block of data. For  $k = 1, \dots, K$ , we denote by  $P_{n,k}^0$  the empirical distribution of observations in  $k$ -th training set and define  $\psi_{n,k} := \hat{\Psi}(P_{n,k}^0)$  as the prediction algorithm trained using the training set. The cross-validated AUPRC (CV-AUPRC) at a distribution  $P \in \mathcal{M}$  is defined as:

$$\phi_{\text{cv}} := \frac{1}{K} \sum_{k=1}^K \Phi_{\psi_{n,k}}(P) . \quad (1.2)$$

This new target parameter is the average of  $K$  different parameters  $\Phi_{\psi_{n,k}}(P)$ , each of which describes the performance of a different prediction function  $\psi_{n,k}$ . Each of these prediction functions in turn depends on the training data used to develop it. Thus  $\phi_{\text{cv}}$  depends on the random splitting of the data and can therefore be called a *data-adaptive* target parameter [52]. We can interpret this strategy as targeting the performance of a *method*  $\Psi$  for developing prediction functions rather than targeting the performance of a single, externally developed function  $\psi$  as in subsection 1.2.1.

## 1.3 Estimation and inference

### 1.3.1 Estimation and inference for AUPRC

A natural estimator for the AUPRC defined in (1.1) is the plug-in estimator

$$\Phi_\psi(P_n) = \int P_n(Y = 1 \mid \psi(X) > z) dF_{n,\psi(X)|Y=1}(z) \quad , \quad (1.3)$$

where  $P_n$  denotes the empirical measure of  $O_1, \dots, O_n$ ,  $P_n(Y = 1 \mid \psi(X) > z)$  denotes the sample proportion of cases with  $\psi(X_i) > z$  and  $F_{n,\psi(X)|Y=1}$  denotes the empirical cumulative distribution function of  $\psi(X)$  amongst the cases.

The large-sample behavior of  $\phi_n := \Phi_\psi(P_n)$  can be characterized by its influence function. We recall that an estimator  $\theta_n$  of a parameter  $\theta_0$  is said to be asymptotically linear if

$$\theta_n - \theta_0 = \frac{1}{n} \sum_{i=1}^n D(O_i) + o_p(n^{-1/2})$$

for some function  $D$  such that  $\int D(o) dP(o) = 0$  and  $\int D(o)^2 dP(o) < \infty$ . The object  $D$  is referred to as the *influence function* of  $\theta_n$ . Asymptotic linearity is a convenient property of estimators in that study of the estimator is facilitated via classical results pertaining to the large sample behavior of sample means, such as the weak law of large numbers and the central limit theorem. Additionally, for so-called *regular* estimators, influence functions are intimately connected to asymptotic efficiency.

*Theorem 1.3.1.* If  $\int \frac{dP_0(\psi(X) \leq z | Y=1)}{[P_0(\psi(X) \geq z | Y=1)]^4} < \infty$ , then  $\phi_n$  is asymptotically linear with influence function

$$\begin{aligned} D_{\Phi_\psi}(P_0)(O_i) := & \\ & \int_{z < \psi(X_i)} \left[ Y_i - \frac{P_0(\psi(X) > z, Y = 1)}{P_0(\psi(X) > z)} \right] \left( \frac{1}{P_0(\psi(X) > z)} \right) dF_{0,\psi(X)|Y=1}(z) \quad (1.4) \\ & + \frac{Y_i}{P_0(Y = 1)} \left[ \frac{P_0(\psi(X) > \psi(X_i), Y = 1)}{P_0(\psi(X) > \psi(X_i))} - \Phi_\psi(P_0) \right] . \end{aligned}$$



The proof of Theorem 1.3.1 is included in the appendix. An immediate corollary of Theorem 1.3.1 is that  $D_{\Phi_\psi}$  is the *efficient* influence function in a nonparametric model. Thus, the plug-in estimator  $\phi_n$  has the smallest asymptotic variance amongst all regular, asymptotically linear estimators of  $\phi_0$ . A second corollary of the theorem is that the Central Limit Theorem implies  $n^{1/2}(\phi_n - \phi_0) \xrightarrow{d} \mathcal{N}(0, \sigma_0^2)$ , where  $\sigma_0^2 := \int D_{\Phi_\psi}(P_0)(o)^2 dP_0(o)$ . We can consistently estimate  $\sigma_0^2$  with  $\sigma_n^2 := \frac{1}{n} \sum_{i=1}^n D_{\Phi_\psi}(P_n)(O_i)^2$ . Thus, if  $z_r$  is the  $r$ th quantile of the standard normal distribution, a  $(1 - \alpha) \times 100\%$  confidence interval for  $\phi_n$  is

$$\left( \phi_n - z_{1-\alpha/2} \frac{\sigma_n}{n^{1/2}}, \phi_n + z_{1-\alpha/2} \frac{\sigma_n}{n^{1/2}} \right). \quad (1.5)$$

The condition in Theorem 1.3.1 is likely to be violated in degenerate scenarios where the machine learning model is fully uninformative for predicting the outcome. For example, if the model generates a score between 0 and 1 uniformly at random and independent of  $Y$ , the condition will not hold. However, we note that this condition is sufficient but likely not necessary. We demonstrate this in the appendix by showing that we still obtain valid inference even when  $\psi(X) \sim \text{Unif}(0, 1)$ . We leave the identification of a less stringent condition for future work.

### 1.3.2 Estimation and inference for cross-validated AUPRC

In order to estimate the CV-AUPRC, we first propose a natural extension to the estimator presented in the previous section, hereafter referred to as the CV estimator. For  $k = 1, \dots, K$ , we denote by  $P_{n,k}^1$  the empirical distribution of observations in the  $k$ -th validation set. The CV estimator is

$$\phi_{n,\text{cv}} := \frac{1}{K} \sum_{k=1}^K \Phi_{\psi_{n,k}}(P_{n,k}^1). \quad (1.6)$$

The  $k$ -th parameter in the sum in (1.2) is estimated by calculating the AUPRC of  $\psi_{n,k}$  using the data in the  $k$ -th validation set.

*Theorem 1.3.2.* Assume that for a given  $k \in \{1, \dots, K\}$ ,

$$\int \{D_{\Phi_{\psi_{n,k}}}(P_0)(o) - D_{\Phi_{\psi_k}}(P_0)(o)\}^2 dP_0(o) \xrightarrow{P} 0,$$

where  $D_{\Phi_{\psi_k}}(P_0)(o)$  is a limiting influence curve indexed by  $k$ . Then, we can write the asymptotic distribution of the CV estimator as  $n^{1/2}(\phi_{n,cv} - \phi_{0,cv}) \xrightarrow{d} \mathcal{N}(0, \sigma_{cv}^2)$  where  $\sigma_{cv}^2 := \frac{1}{k} \sum_{k=1}^K \int D_{\Phi_{\psi_k}}(P_0)(o)^2 dP_0(o)$ . Letting  $n_k^1$  be the number of observations in the  $k$ th validation fold, we can consistently estimate  $\sigma_{cv}^2$  with  $\sigma_{n,cv}^2 := \frac{1}{k} \sum_{k=1}^K \frac{1}{n_k^1} \sum_{i=1}^{n_k^1} D_{\Phi_{\psi_{n,k}}}(P_{n,k}^1)(O_i)$ .

The proof for Theorem 1.3.2 can be found in the Appendix. It follows from Theorem 1.3.2 that, if  $z_r$  is the  $r$ th quantile of the standard normal distribution, a  $(1 - \alpha) \times 100\%$  confidence interval for  $\phi_{n,cv}$  is

$$\left( \phi_{n,cv} - z_{1-\alpha/2} \frac{\sigma_{n,cv}}{n^{1/2}}, \phi_{n,cv} + z_{1-\alpha/2} \frac{\sigma_{n,cv}}{n^{1/2}} \right). \quad (1.7)$$

### 1.3.3 Improved estimation and inference for cross-validated AUPRC

Here we explore an alternative estimation strategy for the CV-AUPRC. As discussed, one of the desirable qualities of the AUPRC as an evaluation metric is that it overcomes the over-optimism sometimes implied by other metrics when there is class imbalance in the data. However, the proposed estimator (1.6) depends on  $P_{n,k}^1(\psi_{n,k}(X) \leq z \mid Y = 1)$ , the empirical probability of classifying an observation as a control amongst *cases* in the  $k$ -th validation set. If there are very few cases in the data as a whole, splitting the data into  $K$  folds might result in  $P_{n,k}^1(\psi_{n,k}(X) \leq z \mid Y = 1)$  being an unstable estimate of  $P_0(\psi_{n,k}(X) \leq z \mid Y = 1)$  and therefore reducing the quality of  $\phi_{n,cv}$ .

Ideally we would be able to utilize more data when estimating each  $\Phi_{\psi_{n,k}}$  rather than relying solely on the  $k$ -th validation fold. To this end, we propose to use a strategy first described by [9]. This strategy involves training *and* evaluating  $\psi_{n,k}$  using  $k$ -th training sample. Training and evaluating using the same data will result in a biased estimate. Therefore, a correction term based on the EIF is built using the  $k$ -th validation sample and is added to the initial estimate.

Specifically, [9] propose an estimator based on nested cross-validation which involves splitting the  $k$ -th training sample into  $V$  additional folds. Let  $P_{n,k,v}^0$  and  $P_{n,k,v}^1$  represent the empirical distributions of the  $v$ -th training and validation samples, respectively, nested within the  $k$ -th outer training sample. We then estimate the  $k$ -th parameter in the sum in (1.2) in the following way. For each  $v \in \{1, \dots, V\}$ , we train the prediction algorithm on the  $v$ -th nested training fold to obtain  $\psi_{n,k,v} := \hat{\Psi}(P_{n,k,v}^0)$ . We then estimate the necessary components of the distribution of  $\psi_{n,k}$  using

$$\frac{1}{V} \sum_{v=1}^V P_{n,k,v}^1(Y = 1 \mid \psi_{n,k,v}(X) > z) \quad (1.8)$$

and

$$\frac{1}{V} \sum_{v=1}^V P_{n,k,v}^1(\psi_{n,k,v}(X) \leq z \mid Y = 1). \quad (1.9)$$

Then, letting  $\hat{P}_{n,k}^0$  be any distribution compatible with (1.8) and (1.9), we construct a plug-in estimate of the  $k$ -th parameter,  $\Phi_{\psi_{n,k}}(\hat{P}_{n,k}^0)$ . The one-step bias correction term constructed using the  $k$ -th validation fold takes the form  $P_{n,k}^1 D_{\Phi_{\psi_{n,k}}}(\hat{P}_{n,k}^0)(o)$ . Thus, the final, bias corrected estimate which we will refer to as the CV-OS can be written as

$$\phi_{n,\text{cv-os}} := \frac{1}{K} \sum_{k=1}^K \left[ \Phi_{\psi_{n,k}}(\hat{P}_{n,k}^0) + \int D_{\Phi_{\psi_{n,k}}}(\hat{P}_{n,k}^0)(o) dP_{n,k}^1(o) \right], \quad (1.10)$$

and we have that  $n^{1/2}(\phi_{n,\text{cv-os}} - \phi_{0,\text{cv-os}}) \xrightarrow{d} \mathcal{N}(0, \sigma_{\text{cv}}^2)$  where

$$\sigma_{\text{cv}}^2 := \frac{1}{K} \sum_{k=1}^K \int D_{\psi_k}(P_0)(o)^2 dP_0(o) .$$

Letting  $n_k^1$  be the number of observations in the  $k$ th validation fold, a consistent variance estimator is

$$\sigma_{n,\text{cv-os}}^2 := \frac{1}{k} \sum_{k=1}^K \frac{1}{n_k^1} \sum_{i=1}^{n_k^1} D_{\Phi_{\psi_{n,k}}}(\hat{P}_{n,k}^0)(O_i) .$$

Thus, if  $z_r$  is the  $r$ th quantile of the standard normal distribution, a  $(1 - \alpha) \times 100\%$  confidence interval for  $\phi_{n,\text{cv-os}}$  is

$$\left( \phi_{n,\text{cv-os}} - z_{1-\alpha/2} \frac{\sigma_{n,\text{cv-os}}}{n^{1/2}}, \phi_{n,\text{cv-os}} + z_{1-\alpha/2} \frac{\sigma_{n,\text{cv-os}}}{n^{1/2}} \right) .$$

## 1.4 Results

### 1.4.1 Simulations

Simulations were performed to examine the performance of each of the proposed estimation and inference strategies using the R Language [41] (version 4.0.2). Three data generating processes were used resulting in data sets with cases comprising approximately 50%, 20%, and 10% of observations. We will refer to these as the *balance*, *imbalance*, and *large imbalance* processes respectively. At each of six sample sizes, 1000 datasets were simulated using each imbalance scheme. For each dataset, we evaluated the performance of Logistic Regression and Random Forest [12] (as implemented in the R package `ranger` [58]) by calculating point estimates and confidence intervals of CV-AUPRC using the CV and CV-OS strategies. This process was repeated using  $K = 5$  and 10 (outer) cross-validation folds. For the CV-OS estimator, the number of

inner folds was set to  $V = 5$  in each case.

Finally, true CVAUPRC values were generated for each  $K$ , learner  $\ell$ , data generating process  $d$ , and sample size  $n$  combination in order to evaluate estimates and 95% confidence interval coverage. That was achieved for a given combination as follows. A dataset of size  $N = 500000$  was generated under process  $d$ . The  $K$  model fits of learner  $\ell$  saved from doing cross-validation on the corresponding dataset of size  $n$  were used to predict on the  $N$  observations of the large dataset. These  $K$  prediction sets were then used to calculate the  $K$  AUPRC values found in the sum in (1.2).

We found that, at smaller sample sizes, the CV-OS estimator shows better performance in terms of bias and MSE with the discrepancy once again growing as imbalance increases (Figures 1.1 and 1.2). It should also be noted that the variance of the CV-OS estimator is consistently larger than that of the CV estimator regardless of imbalance. In the interest of space, we only include results for Random Forest under the no imbalance and large imbalance schemes. However, these results are consistent for both learners and all imbalance schemes. Figures displaying the remaining results as well as zoomed in plots for results at the larger sample sizes can be found in the Appendix (A.5). While we found that coverage probability converges to the nominal level in all cases, it is clear that the CV-OS estimator performs better at smaller sample sizes than the CV estimator (Figure 1.3). Furthermore, performance of the CV-OS estimator is nearly identical when using  $K = 5$  and  $K = 10$  outer folds. By contrast, choosing  $K = 10$  noticeably reduces performance of the CV estimator with the issue becoming more pronounced as imbalance in the data increases.

## 1.4.2 Data analysis

Performance of the CV and CV-OS estimators was evaluated on the Adult and Bank datasets from the publicly available UCI repository [19]. Characteristics of each dataset are summarized in Table 1.1. 1000 datasets of size  $n = 250$  and 1000 datasets of size

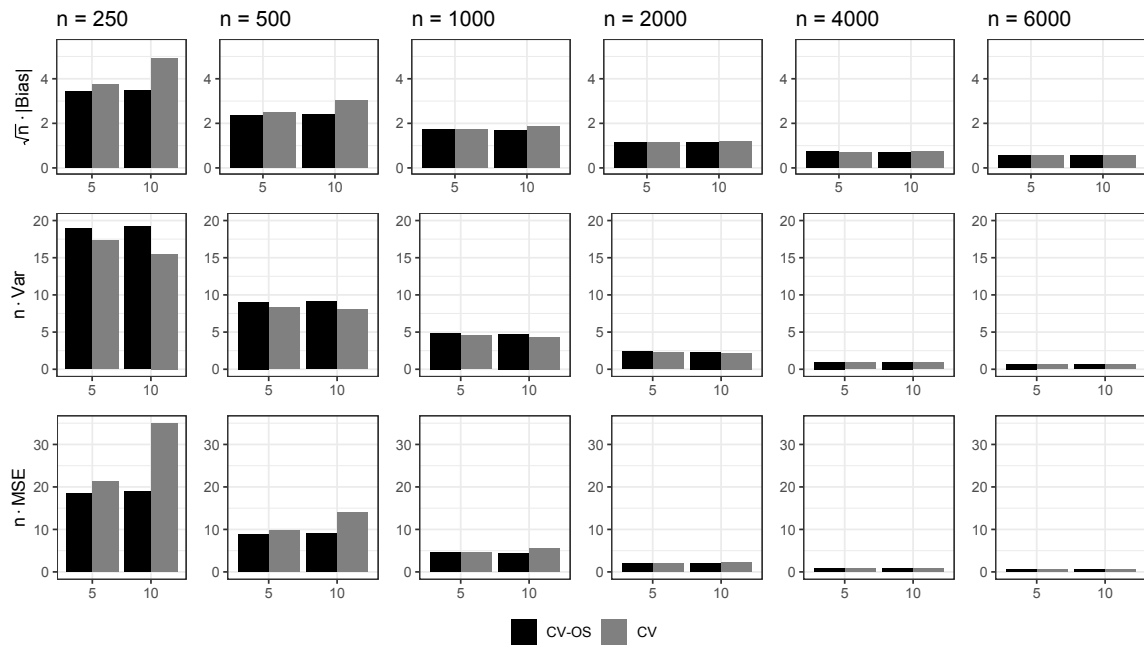


Figure 1.1: Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Random Forest on data with **no imbalance**. 5 and 10 on the x-axis indicate number of outer folds  $K$ . In each case results are based on 1000 simulations.

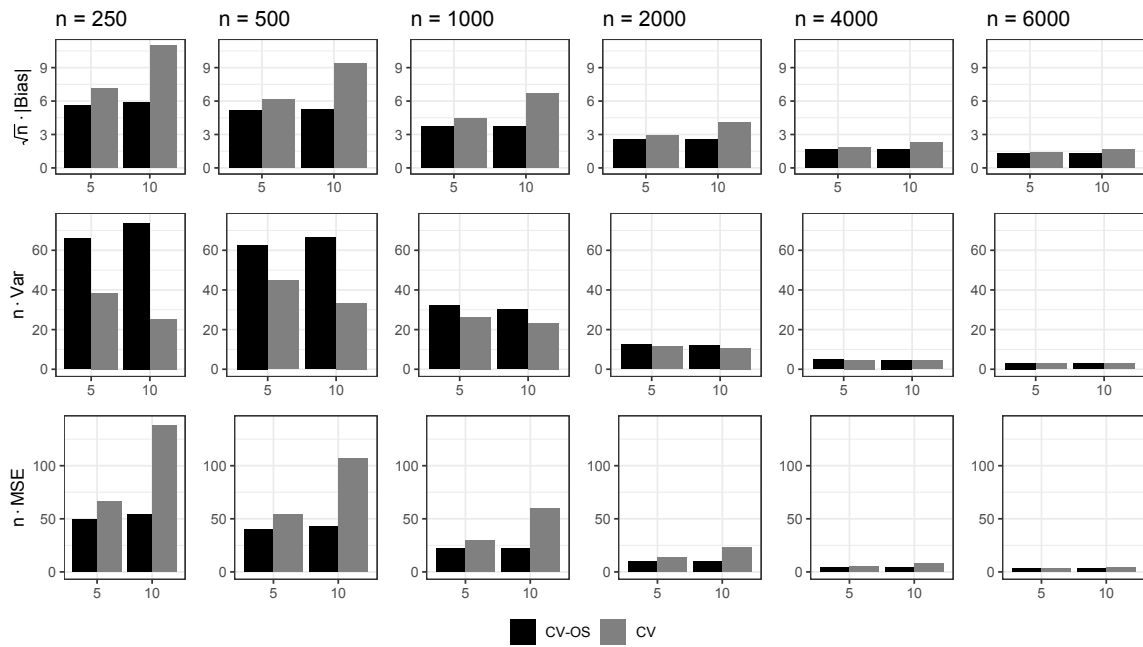


Figure 1.2: Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Random Forest on data with **large imbalance**. 5 and 10 on the x-axis indicate number of outer folds  $K$ . In each case results are based on 1000 simulations.

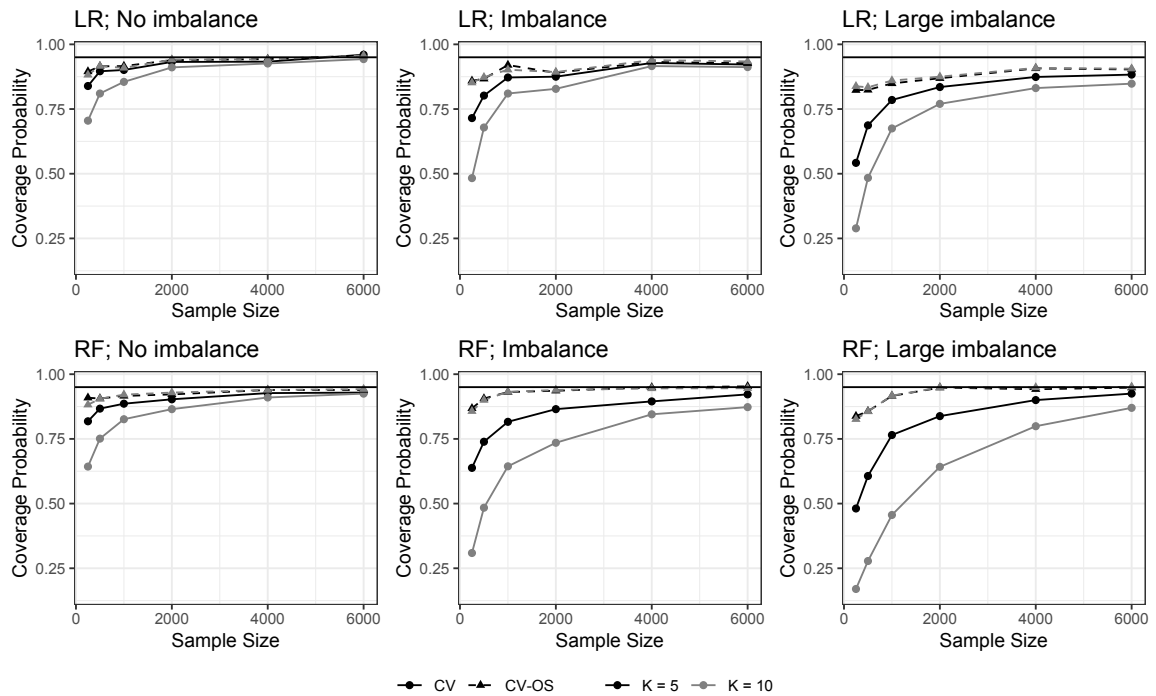


Figure 1.3: Coverage probabilities for 95% confidence intervals. LR and RF stand for Logistic Regression and Random Forest respectively. Probabilities represent the proportion of 95% confidence intervals covering the true value out of 1000 estimation procedures.



$n = 1000$  were sampled from each of Adult and Bank. CV and CV-OS estimators were built along with 95% confidence intervals. Similar to the simulations conducted in Section 1.4.1, model fits were saved during cross-validation and were applied to the remaining  $N - n$  observations in order to calculate true CVAUPRC values. Tables 1.2 and 1.3 compare results of the estimation strategies using Random Forest, Lasso [47, 22], and the Highly Adaptive Lasso (HAL) [8, 25] for sample sizes  $n = 250$  and  $n = 1000$  respectively. True CVAUPRC values and estimates were averaged over the 1000 iterations of the procedure.

Dataset	$N$	$p$	$P_N(Y = 1)$
Adult	32561	86	0.241
Bank	41188	55	0.113

Table 1.1: UCI dataset characteristics.  $N$  is sample size,  $p$  is number of features.

The findings of these analyses are similar to those of the simulations. The average CV-OS estimator is closer to the average true CVAUPRC for each dataset, sample size  $n$ , learner, and choice of  $K$ . The CV estimator tends to underestimate the truth, and that discrepancy is exacerbated by spreading the cases thinner. Bank has much larger class imbalance than Adult, and the CV estimator performance drops correspondingly when applied to Bank. Once again, choosing  $K = 10$  also decreased CV performance. The CV-OS estimator does show drops in performance with increased  $K$  and larger class imbalance, but the drops are much smaller than those of the CV estimator and are nearly non-existent when  $n = 1000$ .

95% confidence interval coverage is uniformly better for the CV-OS estimator, with coverage often coming very close to the nominal level and never dropping below 0.855. Coverage for the CV estimator never surpasses 0.875 and drops as low as 0.157 in the case of using Ranger with data Bank and  $K = 10$  folds.

Data	Model	K	Truth	CV			CV-OS		
				Est.	Width	Cov.	Est.	Width	Cov.
Adult	Ranger	5	0.712	0.627	0.212	0.654	0.712	0.282	0.939
		10	0.717	0.559	0.204	0.198	0.715	0.279	0.949
	Lasso	5	0.682	0.604	0.213	0.687	0.668	0.286	0.934
		10	0.687	0.541	0.202	0.245	0.674	0.285	0.924
	HAL	5	0.660	0.585	0.213	0.699	0.661	0.296	0.940
		10	0.671	0.528	0.201	0.274	0.669	0.295	0.939
Bank	Ranger	5	0.506	0.392	0.260	0.581	0.497	0.410	0.902
		10	0.514	0.301	0.227	0.157	0.505	0.407	0.898
	Lasso	5	0.520	0.411	0.265	0.590	0.513	0.405	0.899
		10	0.527	0.314	0.233	0.174	0.518	0.403	0.855
	HAL	5	0.491	0.381	0.257	0.566	0.480	0.402	0.892
		10	0.500	0.293	0.222	0.172	0.491	0.404	0.867

Table 1.2: Analysis results using  $n = 250$  samples to train and evaluate learners. Remaining observations were used to calculate the true CVAUPRC values. Est. is the average parameter estimate, Width is the average 95% confidence interval width, and Cov. is the proportion of 95% confidence intervals covering the truth. Each entry is a result averaged over 1000 iterations.

Data	Model	K	Truth	CV			CV-OS		
				Est.	Width	Cov.	Est.	Width	Cov.
Adult	Ranger	5	0.759	0.737	0.212	0.862	0.761	0.116	0.938
		10	0.763	0.719	0.204	0.629	0.765	0.114	0.945
	Lasso	5	0.735	0.713	0.213	0.864	0.735	0.120	0.938
		10	0.737	0.695	0.202	0.664	0.739	0.118	0.931
	HAL	5	0.747	0.725	0.213	0.875	0.750	0.119	0.940
		10	0.751	0.708	0.201	0.628	0.753	0.116	0.938
Bank	Ranger	5	0.569	0.534	0.260	0.804	0.567	0.210	0.895
		10	0.574	0.507	0.227	0.580	0.572	0.205	0.898
	Lasso	5	0.569	0.544	0.265	0.842	0.571	0.206	0.922
		10	0.571	0.517	0.233	0.683	0.573	0.207	0.922
	HAL	5	0.572	0.540	0.257	0.810	0.568	0.207	0.903
		10	0.577	0.515	0.222	0.597	0.571	0.206	0.904

Table 1.3: Analysis results using  $n = 1000$  samples to train and evaluate learners. Remaining observations were used to calculate the true CVAUPRC values. Est. is the average parameter estimate, Width is the average 95% confidence interval width, and Cov. is the proportion of 95% confidence intervals covering the truth. Each entry is a result averaged over 1000 iterations.

## 1.5 Discussion

In this study, we found that both the CV and CV-OS estimators perform well in large samples in accordance with the asymptotic distributions shown in sections 1.3.2 and 1.3.3. However, the CV-OS estimator was shown to have superior performance in terms of point estimation and confidence interval coverage in situations with small samples and class imbalance. The results also suggest the CV-OS may be preferable when one wishes to use a larger number of folds  $K$ . CV-OS performance is not seriously impacted by the choice of  $K$ , even in smaller samples. This is a useful property for a number of reasons. First, in practice, this number is often chosen arbitrarily with popular choices being  $K = 3, 5, 10$ . Thus it is valuable to have a method whose performance is invariant to that choice. The CV estimator results in Tables 1.2 and 1.3 suggest that its use will often lead to severe underestimation of model performance if  $K$  is too large. Second, it may actually be preferable to select as large a  $K$  as possible without spreading the data too thin. The reason for this is that, in choosing larger  $K$ , we may be more closely mimicking the non-data-adaptive target parameter that most practitioners likely think they are estimating when carrying out this procedure.

To elaborate on this, we stress the point that the target parameter itself depends not only the data and learning algorithm we are using, but also on  $K$ . Typically after using cross-validation to estimate a metric and compare models, the chosen algorithm will then be fit to the entire dataset to build the model that will be deployed. Thus, in practice, it is this model trained on the full data  $\psi_n := \hat{\Psi}(P_n^0)$  that people wish to evaluate rather than the average performance of  $K$  different models. Evaluating  $\psi_n$  is equivalent to estimating  $\Phi_{\psi_n}(P_0)$ . As  $K$  increases, the CVAUPRC  $\frac{1}{K} \sum_{k=1}^K \Phi_{\psi_n, k}(P_0)$  will approach  $\Phi_{\psi_n}(P_0)$ . In the future it would be worth investigating if the CV-OS procedure with large  $K$  offers a good estimate and a high level of confidence interval coverage for  $\Phi_{\psi_n}(P_0)$  despite targeting a different parameter.

## Chapter 2

# Highly Adaptive Regression Trees: A post-hoc method for interpreting the Highly Adaptive Lasso

### 2.1 Introduction

Recent advances in theoretical statistics and computer science have led to the development of machine learning algorithms that can improve professionals' abilities to accomplish tasks in a myriad of fields. For example, such algorithms have been used to assess risks of future adverse health outcomes [39, 15, 1, 43], estimate consumer demand [7], forecast currency exchange rates [4], predict student academic success [59], and improve communication systems [27]. Machine learning may be used by professionals in these fields to appropriately guide decision making. However, use of machine learning in these contexts often involves weighing the choice of accuracy versus interpretability of a prediction algorithm. Simple algorithms (e.g., based on logistic regression) are easy-to-interpret, but rely on relatively inflexible statistical models and so may not make accurate predictions. More complex algorithms (e.g., based on deep

learning) may predict more accurately, but are often difficult to interpret.

The need for accuracy in these settings is manifest. For example, before machine learning-based technology can be deployed in healthcare settings, providers and patients must have faith in the fidelity of the predictions. Interpretability is also crucial from an ethical standpoint [32]. Consider an algorithm to triage intensive care unit patients based on predicted risk of death. In this case, it is important to carefully scrutinize the triage decision making process to ensure that care is being delivered in an equitable manner.

Decision trees have long been a popular tool for creating interpretable prediction functions and are particularly popular in clinical research [40, 14, 55, 60]. However, traditional techniques for learning these trees generally fail to yield predictions that are as accurate as competitor algorithms [21]. The poor performance may be due to the greedy approach used to fit the trees, which can quickly spread data thin even in relatively data rich settings. This has led to the abandonment of regression trees in settings where accuracy is a primary consideration, in favor of other, more difficult-to-interpret algorithms.

Here we describe a method for constructing classification and regression trees that delivers predictive accuracy comparable to that of random forests and boosting. Our proposal centers around the highly adaptive lasso (HAL) algorithm [8], which has been previously established to have competitive predictive performance compared with state-of-the-art algorithms. Here, we show that HAL can be represented as a decision tree and provide an algorithm for building such a tree from a trained HAL model. We verify that HAL can indeed perform prediction tasks as well as state-of-the-art algorithms, and we demonstrate the interpretability of trees built using real data.

## 2.2 Highly adaptive lasso

We consider a prediction problem where the observed data consist of  $n$  independent copies of the random variable  $O = (X, Y) \sim P_0$ , where  $X \in \mathcal{X}$  is a vector of features,  $Y \in \mathcal{Y}$  is the outcome of interest, and  $P_0$  represents the probability distribution of the observed data. Here,  $\mathcal{Y}$  could be  $\{0, 1\}$  (as in binary classification),  $\{\mathbb{Y}_1, \dots, \mathbb{Y}_k\}$  (multi-class classification), or  $\mathbb{R}$  (regression). For simplicity, we take  $\mathcal{Y} = \{0, 1\}$ .

For a given function  $\psi \in \Psi$ , the space of functions mapping from  $\mathcal{X}$  to the unit interval, we define the negative log-likelihood loss function,  $L(\psi, o) = -\log[\psi(x)^y\{1 - \psi(x)\}^{1-y}]$ . The *risk* of  $\psi$  is the expected value of  $L(\psi, O)$ ,  $R_0(\psi) = \int L(\psi, o)dP_0(o)$ . It is straightforward to show that the minimizer of this risk over  $\Psi$  is  $\psi_0$ , the conditional probability that  $Y = 1$  given  $X$ . The theory of HAL is built around two key assumptions about  $\psi_0$ : (i)  $\psi_0$  is right-continuous with left-hand limits (i.e., is a *cadlag* function) and (ii)  $\psi_0$  has finite variation norm. To precisely define variation norm, we note that any cadlag function  $\psi$  generates a signed measure, which allows us to write integrals with respect to  $\psi$ . The variation norm of  $\psi$  is  $\|\psi\|_v = \int |d\psi(u)|$ .

Previous works [8, 49] demonstrated how to find the minimum loss-based estimator (MLE) in the class of functions with variation norm smaller than a fixed number  $M$ . This MLE estimates  $\psi_{0,M} = \operatorname{argmin}_{\psi: \|\psi\|_v < M} R_0(\psi)$  and may be computed using the fact that any cadlag function with finite variation norm can be arbitrarily well-approximated by a tensor product of indicator basis functions. In the univariate case, we observe  $X_1, \dots, X_n$ , independent copies of a scalar-valued variable  $X$ . The linear combination of basis functions is of the form  $\psi_{n,\beta}(x) = \beta_0 + \beta_X^\top b(x)$ , where  $b(x)$  is an  $n$ -length vector with  $i$ -th entry equal to  $\mathbb{1}_{[X_i, \infty)}(x)$ . Here, we use the indicator notation: for  $(c, d) \in \mathbb{R}^2$ ,  $\mathbb{1}_{[c,d)}(x) = 1$  if  $x \in [c, d)$  and equals 0 otherwise. As the dimension of  $X$  increases, we include tensor product basis functions. For example, if we observe  $(X_{1i}, X_{2i}), i = 1, \dots, n$ , independent copies of  $X = (X_1, X_2)$ , where  $X_1, X_2 \in \mathbb{R}^2$ , then  $\psi_{n,\beta}(x) = \beta_0 + \beta_{X_1}^\top b_1(x) + \beta_{X_2}^\top b_2(x) + \beta_{X_1, X_2}^\top b_1(x)b_2(x)$ , where for  $j = 1, 2$ , the  $i$ -th

entry of  $b_i(x)$  is equal to  $\mathbb{1}_{[X_{j_i}, \infty)}(x_j)$ . The idea generalizes to arbitrary  $p$  with at most  $n(2^p - 1)$  basis functions included.

Because the  $L_1$ -norm of  $\beta$  equals the variation norm of  $\psi_{n,\beta}$ , the MLE in the class of functions with variation norm smaller than  $M$  may be computed by regressing  $Y$  onto this set of basis functions, and ensuring that the  $\sum_{i=1}^n |\beta_i| < M$ . This can be achieved by making use of software for the popular lasso algorithm [47]. We note that with a univariate feature, this approach is well established in the signal de-noising literature under the name of (zero-order) trend filtering. Benkeser and van der Laan (2016) [8] generalized to multiple dimensions and proved that HAL converges to  $\psi_0$  in terms of regret at a fast rate,  $R_0(\psi_{n,M}) - R_0(\psi_{0,M}) = o_p(n^{-[1/4+1/\{8p+1\}]})$ . This theory establishes HAL as a strong candidate for accurate machine learning. In Section 2.4, we establish that HAL is also a good candidate for performing interpretable machine learning.

## 2.3 Notation

The remainder of the paper includes a large set of notation that is necessary for describing the process of building tree representations of HAL models. For this reason, we include Table 2.1 which contains descriptions of all new notation and vocabulary introduced. Each piece of notation is defined in the text, but readers may find it convenient to refer to all definitions in one place.

## 2.4 From HAL to HART

Classification and Regression Tree (CART) [13] is a prediction algorithm that has long been employed across many diverse fields [40]. Decision trees built using this approach typically start with a fully un-partitioned feature space and, through a greedy algorithm, recursively partition the (sub)space into smaller units until a



Term	Description
$\mathbb{X}$	Candidate split data structure to be recursively pruned
$\mathcal{R}$	A feature space region
$b_{ji}(x_j)$	Indicator basis function set to 1 if $X_j \geq x_{ji}$
subscript $G$	A label applied to indicate right-hand sub-region
subscript $L$	A label applied to indicate left-hand sub-region
$\Phi_{ji}$	Set of basis expansion terms involving value $X_{ji}$
$\Phi$	Set of all $\Phi_{ji}$ (represents the whole HAL model fit)
$\pi$	Policy for choosing tree split
Tree	A HART tree object
Tree.label	Text displayed in the root node of Tree
Tree.preds	Set of predictions given by terminal nodes in Tree
Node(label)	Indicates creation of a non-terminal node displaying label
TerminalNode	Indicates creation of a terminal node displaying label
Tree.left-child	Left child node of the root node of Tree
Tree.right-child	Right child node of the root node of Tree
subtree.parent = node	Indicates assignment of node as parent of object subtree

Table 2.1: Notation used throughout the paper to describe aspects of feature space partitioning and tree creation.

specified stopping criterion is met. By nature of the partitioning process, CART yields a histogram approximation of  $\psi_0$ . Furthermore, because of the recursive process employed in training, decision trees are naturally created. However, CARTs rarely achieve high predictive performance and can be prone to overfitting [21].

HAL provides an alternative approach to learning a histogram approximation of a function, though its training process is in stark contrast to CART. HAL starts with a dense partitioning of the feature space (implied by the tensor product basis expansion), that is subsequently shrunk to achieve the best global fit to the data. Because HAL is optimizing a global smoothness criteria, it is able to partition the feature space more efficiently than CART and generally provides a much more accurate representation of  $\psi_0$ . However, the histogram approximation created by HAL partitions will not in general represent a recursive partitioning of the feature space and thus will not be immediately amenable to representation as a decision tree. The goal of this paper is to present an algorithm that maps a trained HAL model into a decision tree. We call

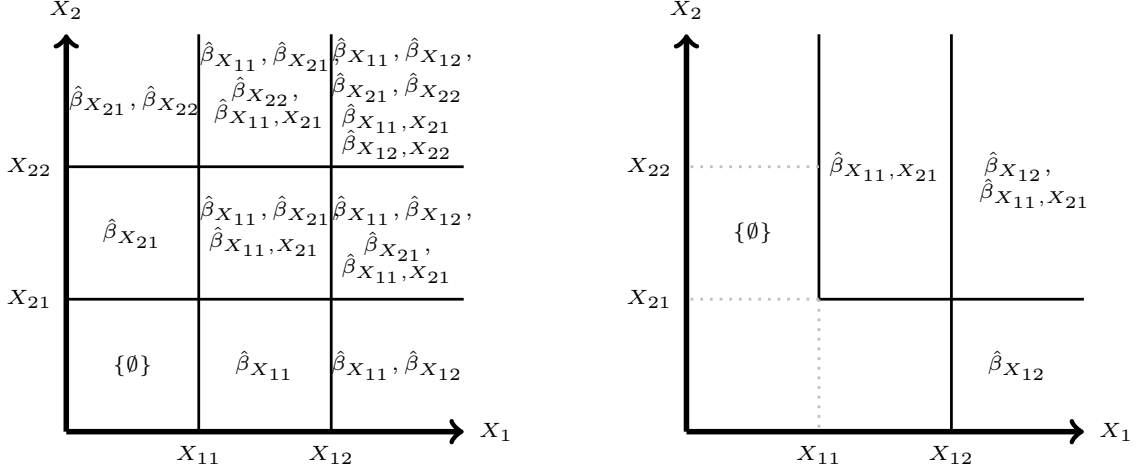


Figure 2.1: Left: Illustration of partitioning of the feature space according to the HAL basis expansion. The sets of  $\hat{\beta}$  (excluding the intercept) correspond to those in equation (2.1) that fall in each region. Right: An illustrative HAL partitioning, when  $\hat{\beta}_{X_{11}} = \hat{\beta}_{X_{21}} = \hat{\beta}_{X_{22}} = \hat{\beta}_{X_{12}, X_{22}} = 0$ , which results in a non-recursive partitioning of the feature space. To represent this partitioning as a decision tree, we must determine a parsimonious recursive structure for the L-shaped left-most region.

the resulting trees Highly Adaptive Regression Trees (HART).

To illustrate the significant challenges associated with this goal, we consider an immensely simplified example in which we observe  $(X_{1i}, X_{2i}), i = 1, 2, n = 2$  observations of the two-dimensional feature-vector  $X = (X_1, X_2)$ . Further assume that  $X_{1i} < X_{2i}$  for  $i = 1, 2$ . Define  $b_{ji}(x) = \mathbb{1}_{[X_{ji}, \infty)}(x_j)$ . The HAL basis expansion is

$$\begin{aligned} \psi_{n, \beta}(x) = & \beta_0 + \beta_{X_{11}} b_{11}(x_1) + \beta_{X_{12}} b_{12}(x_1) + \beta_{X_{21}} b_{21}(x_2) + \beta_{X_{22}} b_{22}(x_2) \\ & + \beta_{X_{11}, X_{21}} b_{11}(x_1) b_{21}(x_2) + \beta_{X_{12}, X_{22}} b_{12}(x_1) b_{22}(x_2) \end{aligned} \quad (2.1)$$

The left panel of Figure 2.1 illustrates the partitioning of the feature space implied by this basis expansion. For example, given  $\hat{\beta} \in \mathbb{R}^7$ , the HAL prediction for observations falling in the lowest left portion of the feature space is  $\hat{\beta}_0$ , for the upper left portion the prediction is  $\hat{\beta}_0 + \hat{\beta}_{X_{21}} + \hat{\beta}_{X_{22}}$ , and so on. Note that the full partitioning implied by the basis expansion (equivalent to the partitioning implied if all coefficient estimates are nonzero) is naturally recursive and thus can immediately be represented

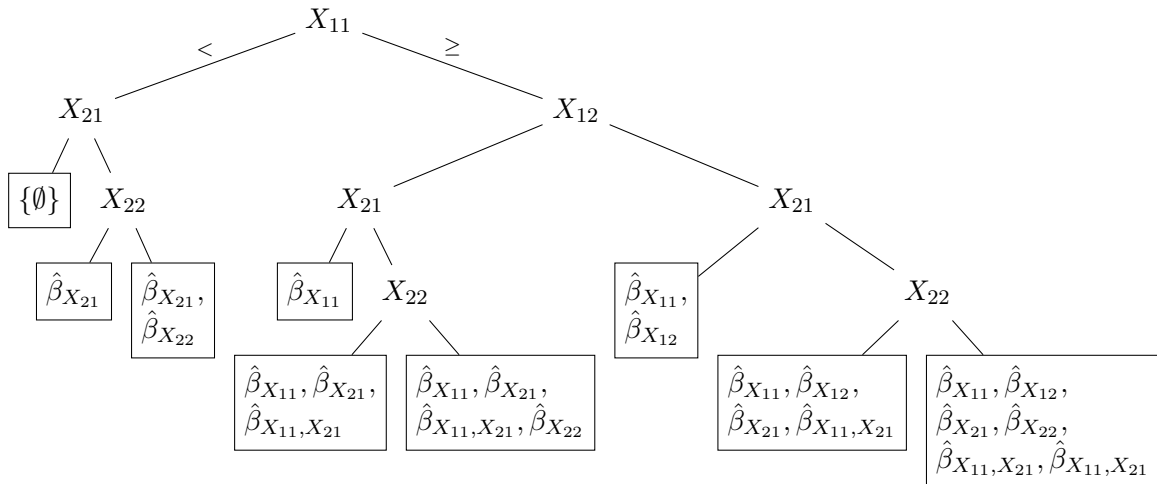


Figure 2.2: Decision tree representation of the full feature space partitioning implied by the HAL basis expansion.

by the decision tree in Figure 2.2. In this and subsequent trees, each node contains an observed value; moving to the left (right) of the node implies a value less (equal to or greater) than the node value.

Because HAL uses  $L_1$ -penalization of the coefficient vector, many coefficients are shrunk to zero when the model is trained; this causes some partitions of the feature space to collapse. For example, suppose we find that  $\hat{\beta}_{X_{11}} = \hat{\beta}_{X_{21}} = \hat{\beta}_{X_{22}} = \hat{\beta}_{X_{12}, X_{22}} = 0$ . In this case, the partitioning collapses (right panel, Figure 2.1) and *does not have a recursive structure*. In this case, any decision tree representing this partitioned space will have at least some redundancy in the terminal nodes. Figure 2.3 illustrates two possible recursive partitions that could be used to approximate the non-recursive partitioning implied by HAL. Note that these are both valid representations in that they would yield predictions identical to those generated by the HAL model. However, each has one redundant partition. This can be observed by noting that each has separate regions with identical predictions.

In higher dimensions, the partitions cannot be visualized, and some recursive partitions may have a large amount of redundancy. In the next sections we motivate

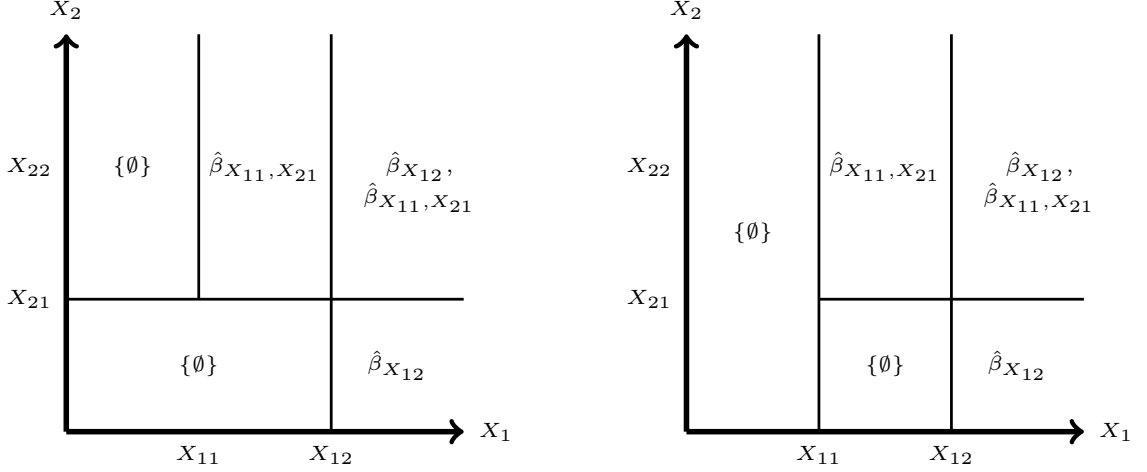


Figure 2.3: Two possible recursive partitions to represent the non-recursive partition implied by the HAL model when  $\hat{\beta}_{X_{11}} = \hat{\beta}_{X_{21}} = \hat{\beta}_{X_{22}} = \hat{\beta}_{X_{12}, X_{22}} = 0$ .

and describe our algorithm for mapping an arbitrary set of non-zero basis functions into a recursive partitioning of the feature space.

Throughout this work, we consider the set  $\mathbb{X} = \{X_{ji} : j = 1, \dots, p, i = 1, \dots, n\}$  which contains the values of our data. The elements of  $\mathbb{X}$  act as candidates for describing partitions implied by HAL. At each stage in the proposed process, a single value is chosen from  $\mathbb{X}$  to describe a partition, and other values are pruned if they are not needed to describe the remaining partitions implied by HAL. The crux of our problem is then identifying which values from our data should be candidates for describing partitions and the order in which we should choose them. As we will see, both of these factors impact the amount of redundancy contained in our final recursive partitioning.

## 2.5 Examples

To motivate our approach for reducing redundancy in the HAL tree growing process, we consider three scenarios: (i) a set of basis functions that naturally yields a recursive structure; (ii) a set of basis functions that yields a non-recursive structure for which

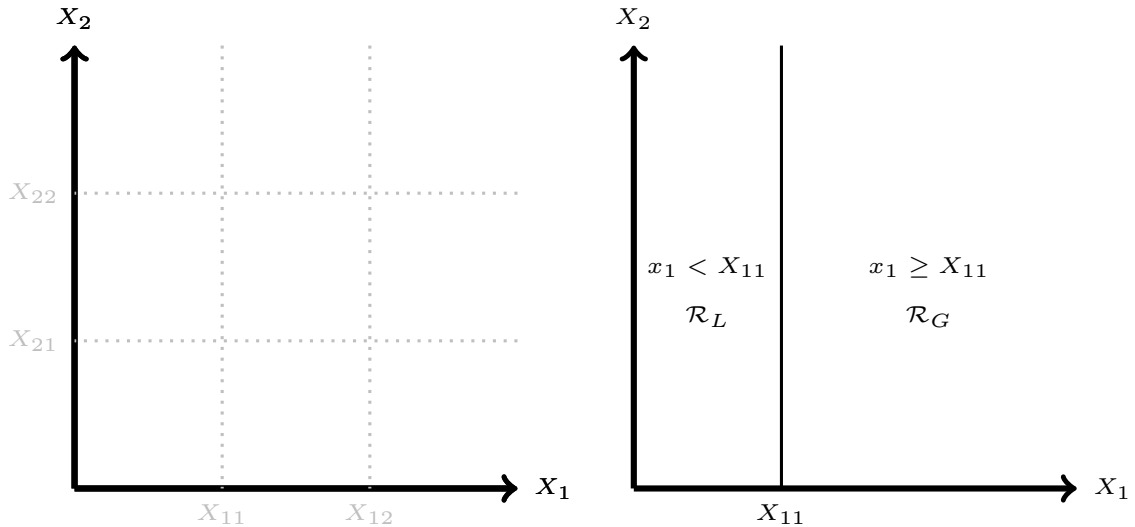


Figure 2.4: Left: The feature space partitioned according to the HAL fit with all  $\hat{\beta} \neq 0$ . These partitions will be described by values from  $X$ . Right: Feature space with  $X_{11}$  describing the first partition.

there are many minimally redundant recursive approximations; (iii) a set of basis functions that yields a non-recursive partitioning for which there are both minimally redundant and highly redundant recursive representations. With each increase in complexity, we will highlight necessary algorithmic changes to the tree growing process.

### 2.5.1 Scenario (i): naturally recursive partitioning

Consider a situation where estimates of the coefficients in the basis expansion in equation (2.1) are all nonzero. The partitioning implied by that model is the full partitioning of the feature space (left panel of Figure 2.1). The data structure in our recursive process is  $\mathbb{X} = \{X_{11}, X_{12}, X_{21}, X_{22}\}$ , the elements of which represent potential splits in our feature space and potential nodes in our tree. We can select values from  $\mathbb{X}$  to describe feature space partitions in any order. Suppose we begin with value  $X_{11}$ . A line is then drawn dividing the feature space a region  $\mathcal{R}_L$  where  $x_1 < X_{11}$  and a region  $\mathcal{R}_G$  where  $x_1 \geq X_{11}$  (Figure 2.4).

In each of the two new sub-regions of the feature space, we can begin the process

again, now with a reduced data structure. In both sub-regions, we set  $\mathbb{X} = \mathbb{X} \setminus \{X_{11}\}$  since  $X_{11}$  has just been partitioned on. Recall now that we specified  $X_{11} < X_{12}$ . In sub-region  $\mathcal{R}_L$  where  $x_1 < X_{11}$ , it must be true that  $x_1 < X_{12}$ . Thus, it is clearly not necessary to further partition  $\mathcal{R}_L$  using the value  $X_{22}$ , and we can set  $\mathbb{X}_L = \mathbb{X} \setminus \{X_{12}\} = \{X_{21}, X_{22}\}$ . By contrast, in  $\mathcal{R}_G$ ,  $x_1 \geq X_{11}$ , but it may or may not be true that  $x_1 \geq X_{12}$ . Thus, we set  $\mathbb{X}_G = \{X_{12}, X_{21}, X_{22}\}$ .

Suppose we carry out this recursive process until  $\mathbb{X}$  is empty in every sub-region. Predictions are then constructed by summing the appropriate coefficient estimates for that region. The resulting partitioned feature space and the corresponding decision tree perfectly represent the fully partitioned feature space in the left panel of Figure 2.1 and its decision tree in Figure 2.2 respectively.

### 2.5.2 Scenario (ii): non-recursive partitioning, minimal redundancy

Suppose we estimate  $\hat{\beta}_{X_{11}} = \hat{\beta}_{X_{21}} = \hat{\beta}_{X_{22}} = \hat{\beta}_{X_{12}, X_{22}} = 0$  and the remaining parameters in 2.1 to be nonzero. The partitioning implied by this model is pictured in the right panel of Figure 2.1. It is worth noting that the full partitioning in the left panel of Figure 2.1 is a valid representation of this non-recursive partitioning. However, the associated decision tree would have *considerable* redundancy. Consider two examples of redundancy in that partition under this HAL fit.

First we can see that,  $X_{22}$  is not needed to describe any of the partitions implied by the HAL fit. This suggests that our initial data structure  $\mathbb{X}$  need not contain  $X_{22}$ ; we could set  $\mathbb{X} = \{X_{11}, X_{12}, X_{21}\}$ . Suppose again that we choose  $X_{11}$  to describe the first partition. Two sub-regions  $\mathcal{R}_L$  and  $\mathcal{R}_G$  are created where  $x_1 < X_{11}$  and  $x_1 \geq X_{11}$  respectively. As in Scenario (i), in  $\mathcal{R}_G$  we can simply prune  $X_{11}$  and set  $\mathbb{X} = \mathbb{X} \setminus \{X_{11}\}$ . However, in  $\mathcal{R}_L$ , there are no remaining partitions suggesting that in this region  $X_{11}, X_{12}$ , and,  $X_{21}$  should be pruned from  $\mathbb{X}$ . This example highlights the

necessity for a more comprehensive rule for pruning values from  $\mathbb{X}$  in a certain region, given a HAL model fit.

In particular, a value  $X_{ji}$  is unnecessary for describing partitions in a given region  $\mathcal{R}$  if it meets one of two criteria. First,  $X_{ji}$  is unnecessary if all coefficient estimate-basis function product terms involving  $X_{ji}$  in the HAL basis expansion evaluate to zero. For example, consider the region where  $x_1 < X_{11}$ . There,  $X_{21}$  is unnecessary because  $\hat{\beta}_{X_{21}} b_{21}(x_2) = 0$  and  $\hat{\beta}_{X_{11}, X_{21}} b_{11}(x_1) b_{21}(x_2) = 0$ . The second criterion is that  $b_{ji}(x_j) = 1$  for all  $x_j$ . For example,  $X_{11}$  is unnecessary in the region where  $x_1 \geq X_{11}$  because it is known that  $b_{11}(x_1) = 1$ . Following these criteria, and assuming we select values for partitioning in the order that they appear in  $\mathbb{X}$ , we obtain the partitioning shown in the left panel of Figure 2.3. As we showed in Section 2.4, this is just one possible minimally redundant recursive approximation of the non-recursive partitioning implied by the HAL model. In fact, after initially pruning  $X_{22}$ , we could order the elements of  $\mathbb{X}$  in six different ways and therefore obtain six different, minimally redundant trees.

### 2.5.3 Scenario (iii): non-recursive partitioning, minimal and non-minimal redundancy

Now assume that we estimate  $\hat{\beta}_{X_{11}} = \hat{\beta}_{X_{12}} = \hat{\beta}_{X_{22}} = \hat{\beta}_{X_{11}, X_{21}} = 0$  and the remaining coefficients as nonzero. The feature space partitioning implied by this model is shown in Figure 2.5. According to our criteria from Section 2.5.2, we begin by pruning  $X_{11}$  from  $\mathbb{X}$ . The left and right panels of Figure 2.6 show partitionings resulting from proceeding with orderings  $\{X_{12}, X_{21}, X_{22}\}$  and  $\{X_{21}, X_{22}, X_{12}\}$  respectively. Choosing the first ordering results in an unnecessary partition.

This motivates the introduction of some policy  $\pi$  that dictates which value should be chosen to describe the next partition given the data  $\mathbb{X}$ , the current region  $\mathcal{R}$ , and the model fit information. The problem of finding the most parsimonious recursive partitioning then reduces to finding an optimal policy  $\pi^*$ . Potential policies are

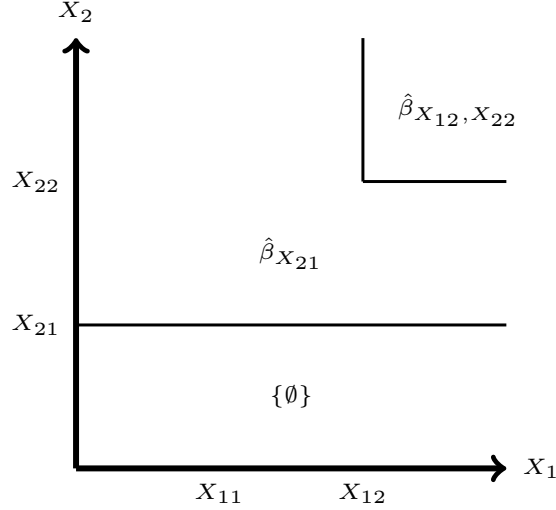


Figure 2.5: An illustrative HAL partitioning, when  $\hat{\beta}_{X_{11}} = \hat{\beta}_{X_{12}} = \hat{\beta}_{X_{22}} = \hat{\beta}_{X_{11}, X_{21}} = 0$ , which results in a non-recursive partitioning of the feature space.

discussed in Section 2.6.2.

## 2.6 Highly adaptive regression trees

### 2.6.1 Growing trees

Algorithm 1 presents a formal algorithm for finding parsimonious recursive representations of feature space partitionings implied by HAL models. The algorithm is presented for a design matrix involving  $n$  observations of  $p$  elements,  $\mathbb{X} = \{X_{ji} : j = 1, \dots, p, i = 1, \dots, n\}$ . Let  $\Phi_{ji}$  be the set of all coefficient estimate-basis function product terms involving the value  $X_{ji}$  in the HAL basis expansion, and define  $\Phi = \{\Phi_{ji} : j = 1, \dots, p, i = 1, \dots, n\}$ , a set of sets that represents the whole HAL model fit. A given region of the feature space can be represented by the tuple  $\mathcal{R} = (\Delta, \Gamma)$  where  $\Delta(x) = \{X_{ji} : x_j < X_{ji}\}$  and  $\Gamma(x) = \{X_{ji} : x_j \geq X_{ji}\}$ . At the beginning of the process, we initialize  $\Delta = \Gamma = \{\emptyset\}$  and let  $\mathcal{R} = (\{\emptyset\}, \{\emptyset\})$  represent the entire feature space.

Once a partition is described using a value  $X_{ji}$ , it naturally implies two new regions



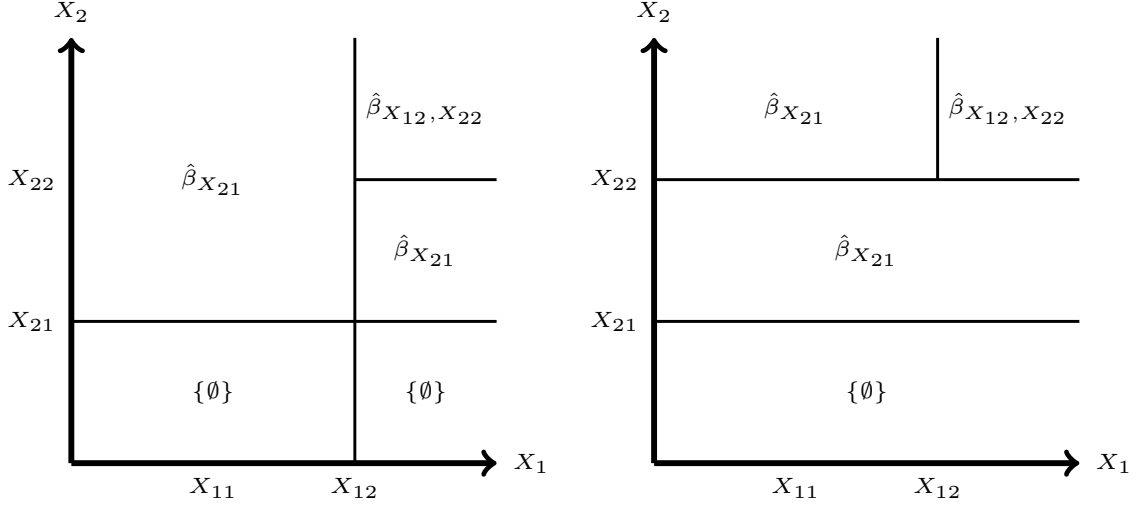


Figure 2.6: Two recursive partitionings to represent the non-recursive partitioning implied by HAL when  $\hat{\beta}_{X_{11}} = \hat{\beta}_{X_{12}} = \hat{\beta}_{X_{22}} = \hat{\beta}_{X_{11}, X_{21}} = 0$ . The left partitioning is not minimally redundant. The right partitioning is minimally redundant.

---

**Algorithm 1:** HART

---

```

1:  $\mathbb{X} = \{X_{ji} : j = 1, \dots, p, i = 1, \dots, n\}$ 
2:  $\Phi = \{\Phi_{ji} : j = 1, \dots, p, i = 1, \dots, n\}$ 
3:  $\Delta = \Gamma = \{\emptyset\}$ 
4:  $\mathcal{R} = (\Delta, \Gamma)$ 
5:  $\mathbb{X} = \mathbb{X} \setminus \{X_{ji} : \Phi_{ji} = \{0, \dots, 0\} \mid \mathcal{R}\}$ 
6: def Grow( $\mathbb{X}, \Phi, \mathcal{R}$ )
7: if  $\mathbb{X} = \{\emptyset\}$  then
8:    $\hat{\psi} = \sum_{s \in \Gamma} \hat{\beta}_s$ 
9:   return TerminalNode( $\hat{\psi}$ )
10: else
11:    $X_{j^*i^*} = \pi(\mathbb{X}, \Phi, \mathcal{R})$ 
12:   node = Node( $X_{j^*i^*}$ )
13:    $\Delta_L = \Delta \cup \{X_{j^*i} : X_{j^*i} \geq X_{j^*i^*}\}$ 
14:    $\mathcal{R}_L = (\Delta_L, \Gamma)$ 
15:    $\mathbb{X}_L = \mathbb{X} \setminus \{X_{ji} : \Phi_{ji} = \{0, \dots, 0\} \mid \mathcal{R}_L\}$ 
16:   node.child = Grow( $\mathbb{X}_L, \Phi, \mathcal{R}_L$ )
17:    $\Gamma_G = \Gamma \cup \{X_{j^*i} : X_{j^*i} \leq X_{j^*i^*}\}$ 
18:    $\mathcal{R}_G = (\Delta, \Gamma_G)$ 
19:    $\mathbb{X}_G = \mathbb{X} \setminus \{X_{ji} : b_{ji} = 1 \mid \mathcal{R}_G\}$ 
20:   node.child = Grow( $\mathbb{X}_G, \Phi, \mathcal{R}_G$ )
21:   return node
22: end if
23: call Grow( $\mathbb{X}, \Phi, \mathcal{R}$ )

```

---

within the feature space.  $\mathcal{R}_L$  describes the region in which  $x_j < X_{ji}$  and  $\mathcal{R}_G$  describes the region in which  $x_j \geq X_{ji}$ . These new regions can be formalized by updating the components of the tuple  $\Delta$  and  $\Gamma$  to include relevant values. As values are added to these sets, basis functions from the HAL basis expansion take on realizations of one or zero thus changing values in  $\Phi$ . We use the notation  $\Phi_{ji} \mid \mathcal{R}$  to refer to the specific realization of a set of product terms  $\Phi_{ji}$  within a specific region  $\mathcal{R}$ . Together, the information in the HAL fit and the current region inform which values should be pruned from  $\mathbb{X}$  before further partitions are described. We use  $\pi$  to denote a general policy that dictates which value in  $\mathbb{X}$  should be chosen to describe the next partition.  $\pi$  is a mapping from the current state of the partitioning, described by  $(\mathbb{X}, \Phi, \mathcal{R})$ , to the value that will describe the next partition  $X_{j^*i^*}$ .

Finally, in Algorithm 1 we make the connection between feature space partitions and nodes in a decision tree. Each time we describe a partition using a value  $X_{ji}$ , we indicate the creation of a tree node with  $\text{Node}(X_{ji})$ . The two children of that node will correspond to the partitions in the resulting sub-regions where  $x_j < X_{ji}$  and  $x_j \geq X_{ji}$  respectively.

## 2.6.2 Policies

As stated in Section 2.4, the two sources of redundancy in our tree representations are the identification of candidates for partitioning and the order in which we should choose them. Algorithm 1 fully accounts for the former and ensures that no unnecessary feature values are identified as candidates in a given stage of the growing process. The ordering of those candidates and the resulting redundancy is dictated solely by the policy  $\pi$ . Ideally we would choose  $\pi$  to achieve an optimally parsimonious representation of  $\hat{\psi}$ . However, ordering binary variables to achieve optimally parsimonious representations of functions is known to be an NP-complete problem [10]. We instead rely on heuristics to reduce redundancy.

We propose a heuristic that orders split candidates  $X_{ji}$  based on the number of non-zero coefficient estimate-basis function product terms involving  $X_{ji}$ . Intuitively, this is the number of times  $X_{ji}$  appears in the basis expansion for a given region and can be thought of as a measure of importance. Formally, for split candidates  $\mathbb{X}$  and HAL fit  $\Phi$  in region  $\mathcal{R}$  of the feature space, the next split is chosen as

$$\pi_h(\mathbb{X}, \Phi, \mathcal{R}) := X_{j^*i^*} \text{ where } (j^*, i^*) = \operatorname{argmax}_{j,i} |\nu \in \phi_{ji} \mid \mathcal{R} : \nu \neq 0| \quad (2.2)$$

As an alternative, we propose to take advantage of popular existing decision tree algorithms and define  $\pi$  based on their splitting criteria. This strategy has the benefit of imbuing HART splits with the same interpretation as those of the chosen algorithm as well as acting as a heuristic way of achieving parsimony. For example, in the classification setting, CART chooses splits to minimize Gini Impurity. Heuristically, minimizing Gini Impurity is akin to achieving the ‘best separation’ of the classes in that region. We can build a HART carrying this same interpretation by employing Algorithm 1 and letting  $\pi$  choose the the minimizer of the Gini Impurity. Formally, if we let  $P_{n,\mathcal{R}}$  represent the empirical measure based on data falling in region  $\mathcal{R}$ , then a Gini Impurity based policy can be defined in the following way. For split candidates  $\mathbb{X}$  and HAL fit  $\Phi$  in region  $\mathcal{R}$  of the feature space, the next split is chosen as

$$\begin{aligned} \pi_{\text{gini}}(\mathbb{X}, \Phi, \mathcal{R}) := \operatorname{argmin}_{x_{ji} \in \mathbb{X}} \{ & 2P_{n,\mathcal{R}}(X_j < x_{ji})P_{n,\mathcal{R}_L}(Y = 1)P_{n,\mathcal{R}_L}(Y = 0) \\ & + 2P_{n,\mathcal{R}}(X_j \geq x_{ji})P_{n,\mathcal{R}_G}(Y = 1)P_{n,\mathcal{R}_G}(Y = 0)\} \end{aligned}$$

We could similarly define  $\pi$  based on the splitting criteria of C4.5 [44], CHAID [29], Minimum Message Length based Decision Graphs [38], or any other algorithm. We can also adapt any of these policies to the needs of the problem setting. Say we are

predicting some disease outcome based on treatment status and other features. We could enforce that treatment status is always the final split in the tree and otherwise use  $\pi_{\text{gini}}$ . This would allow us to examine how the predicted outcome differs by treatment status for any given subgroup.

### 2.6.3 Sub-HARTs and Prediction Smoothing

Even with a well chosen policy, the full HART may be large and difficult to interpret globally. We can use two simple strategies to mitigate the resultant complexity. First, we can restrict the region of the feature space in which to visualize the model. In Equation 2.1, if we restrict  $X_1 < X_{11}$ , then  $b_{X_{11}} = 0$ , and we have a simplified function. This simplified function still follows the structure of a HAL model. Therefore Algorithm 1 can be applied to build a Sub-HART that represents the model only in the region where  $X_1 < X_{11}$ . The Sub-HART will necessarily be smaller and easier to interpret. The choice of how to restrict the feature space can be motivated by the specific problem setting. For example, consider fitting HAL to predict the recurrence of breast cancer based on demographic features and features related to the original tumors. If we are mainly interested in how the estimated function classifies older subjects, we could restrict  $\text{Age} \geq 65$ . If we are interested in how the function differs between older and younger subjects, we could compare Sub-HARTs restricting  $\text{Age} < 20$  and  $\text{Age} \geq 65$  respectively.

A second strategy is to smooth the predicted outcomes so that there are fewer unique predictions to display. We can then collapse regions of the tree that no longer show any heterogeneity. HART may display many adjacent terminal nodes with predictions that differ by small, clinically insignificant amounts. This level of granularity might unnecessarily add to the size of the tree. One method for smoothing predictions is to break the prediction space into a set number of intervals and only display the correct interval in each terminal node (Algorithm 2). For example, in the

---

**Algorithm 2:** Bin Predictions
 

---

```

1: def Bin(Tree, Q):
2:   intervals =  $\{[0, \frac{1}{Q}), [\frac{1}{Q}, \frac{2}{Q}), \dots, [\frac{Q-1}{Q}, 1]\}$ 
3:   if Tree is TerminalNode then
4:     max =  $\frac{1}{Q} \cdot \text{ceiling}(\frac{\text{Tree.pred}}{Q})$ 
5:     min = max -  $\frac{1}{Q}$ 
6:     return TerminalNode([min, max))
7:   else if All Tree.preds in i for i in intervals then
8:     return TerminalNode(i)
9:   else
10:    node = Node(Tree.label)
11:    left-subtree = Bin(Tree.left-child, Q)
12:    right-subtree = Bin(Tree.right-child, Q)
13:    left-subtree.parent = node
14:    right-subtree.parent = node
15:    return(node)
16:   end if

```

---



---

**Algorithm 3:** Aggregate Predictions
 

---

```

1: def Agg(Tree, K):
2:   if Tree is TerminalNode then
3:     return TerminalNode(Tree.pred)
4:   else if max(Tree.preds) - min(Tree.preds)  $\leq K$  then
5:     interval = [min(Tree.preds), max(Tree.preds)]
6:     return TerminalNode(interval)
7:   else
8:     node = Node(Tree.label)
9:     left-subtree = Agg(Tree.left-child, K)
10:    right-subtree = Agg(Tree.right-child, K)
11:    left-subtree.parent = node
12:    right-subtree.parent = node
13:    return(node)
14:   end if

```

---

binary classification setting, we may bin the predictions into low ( $[0, .333)$ ), medium ( $[0.333, 0.667)$ ), and high ( $[0.667, 1.0)$ ) probability of success. A second method is to choose some minimum amount of prediction heterogeneity required to introduce new splits (Algorithm 3). In the binary classification setting, we may enforce that splits only occur if they separate predicted probabilities having a difference higher than, say, 0.20. If a region of the tree has many terminal nodes and yet the predicted probabilities range only from .05 to 0.25, we can group that region into a single terminal node and display the prediction range. Even if one is still interested in viewing the predictions at the most granular level, a HART with smoothed predictions could be a convenient first step to identify interesting sub-regions to investigate further.

## 2.7 Algorithm Complexity

Here we examine the theoretical time complexities of Algorithms 1, 2, and 3.

### 2.7.1 HART Complexity

The exact time complexity of Algorithm 1 is difficult to specify as it depends on the proportion of candidate splits pruned after each node creation in the tree. That number is entirely dependent on the chosen splitting policy  $\pi$  and the fitted HAL model. Our analysis here is therefore limited to a range of worst-case and more optimistic scenarios.

We begin by considering the per-node computation in terms of the number of initial split candidates, which we will refer to as  $c$ . Given a split value, a node is created first by updating the set  $\mathcal{R}$  to pinpoint the feature space region being considered. That involves comparing the split value to a set of values on the order of  $O(n)$  within the same feature as the split value. Once the region is updated we shrink the candidate set which, involves looking at all candidates. This is a computation on the order of

$O(c)$ . We then take the remaining list of candidates, and compute a score for each based on the model fit information. In the case of  $\pi_{\text{gini}}$  that score is the gini impurity induced by splitting the data on that candidate. In the case of  $\pi_h$  that is the number of times the given candidate appears in the HAL basis expansion. Since we must examine each candidate once, this computation is on the order of  $O(c)$ . Once the best candidate is identified, we create a new node in the tree which is done in constant time,  $O(1)$ . We perform this set of operations for each non-terminal node in tree. In the absolute worse case in which only the chosen split is pruned at each stage, we have  $2^{c-1}$  non-terminal nodes. In the case of a balanced, binary tree in which half of the candidates are pruned after each split, there are  $2^{\log_2(c)-1} = c$  non-terminal nodes. Depending on the splitting policy  $\pi$  and the fitted model, there may be cases in which more or fewer than half of the candidates are pruned after each split. Thus, we estimate that the overall time complexity of Algorithm 1 is between the worst case of  $O(nc^2 \cdot 2^{c-1})$  and a more optimistic case of  $O(nc^3)$ .

### 2.7.2 Bin and Aggregate Predictions Complexity

Next we discuss the complexity of Algorithms 3 and 2. In both of these algorithms, a constant amount of work is done for each non-terminal node examined. In the worst case scenario, all non-terminal nodes in the tree are examined, although in practice it will be fewer if there is any binning or aggregation to be done. Therefore, the worst case time complexity is  $O(t)$  where  $t$  is the total number of non-terminal nodes.

## 2.8 Data Analysis

We examined the predictive performance of HAL, CART, Random Forest (Breiman, 2001), and XGBoost (Chen and Guestrin, 2016) using four publicly available data sets from the UCI Machine Learning Repository (Table 2.2) all of which have binary

outcomes. Next, we focus on one data set and compare the decision tree produced by CART to those produced by HART using the heuristic policy shown in equation (2.2).

Name	Citation	$N$	$p$	$P_N(Y = 1)$
Breast Cancer	Zwitter et al. (1988) [61]	285	9	0.298
Cardio	Ayres-de Campos et al. (2000) [6]	2126	21	0.139
Drugs	Fehrman et al. (2017) [20]	1885	12	0.186
Wine	Aeberhard et al. (1992) [2]	6497	12	0.197

Table 2.2: UCI dataset characteristics.  $N$  is sample size,  $p$  is number of features.

### 2.8.1 Performance

For each dataset in Table 2.2, HAL, CART, Random Forest, and XGBoost were evaluated by calculating several 10-fold cross-validated performance metrics. For each of CART, Random Forest, and XGBoost, models were built under a grid of 10 possible tuning parameter settings. We only report the results corresponding to models built with the parameter settings that resulted in the highest CV-AUCs for each learner. We found that HAL, Random Forest, and XGBoost typically provided large improvement over CART in all metrics (Table 2.3). Moreover, we found that the performance of HAL is comparable to Random Forest and XGBoost, which are considered state-of-the-art. For a more extensive examination of HAL’s performance, see Benkeser and van der Laan (2016) [8].

### 2.8.2 Trees

Here we focus on the Breast Cancer dataset to demonstrate the interpretability of HART. We aim to predict the recurrence of breast cancer with nine features, descriptions for which can be found in Table 2.4. The left panel of Figure 2.7 shows the tree built using CART with tuning parameters selected using 10-fold cross-validation. Here, the fitted CART model implies that, of the 9 available features, only the



Data	Learner	AUC	Accuracy	Precision	Recall	Time
Breast Cancer	HAL	0.710	0.751	0.709	0.307	13.415 sec
	CART	0.669	0.702	0.500	0.366	0.297 sec
	XGboost	0.712	0.737	0.593	0.376	13.43 min
	Random Forest	0.696	0.722	0.588	0.235	32.882 sec
Cardio	HAL	0.973	0.953	0.886	0.763	1.73 min
	CART	0.933	0.935	0.798	0.712	0.611 sec
	XGboost	0.981	0.961	0.898	0.810	46.89 min
	Random Forest	0.979	0.950	0.879	0.739	3.594 min
Drugs	HAL	0.748	0.814	0.604	0.057	28.58 min
	CART	0.662	0.762	0.306	0.220	0.635 sec
	XGboost	0.746	0.814	0.486	0.051	39.453 min
	Random Forest	0.741	0.816	0.615	0.023	3.472 mins
Wine	HAL	0.973	0.953	0.886	0.763	1.76 hour
	CART	0.813	0.817	0.537	0.482	0.380 sec
	XGboost	0.904	0.874	0.741	0.554	30.380 min
	Random Forest	0.921	0.888	0.862	0.510	3.563 min

Table 2.3: 10-fold cross-validated metrics are reported for Breast Cancer, Cardio, and Drugs. 3-fold cross validated metrics are reported for Wine. For CART, Random Forest, and XGBoost, we carried out grid searches over 10 tuning parameter settings. Results correspond to models with the highest performing tuning parameters. Computations were carried out on a High Performance Computing cluster.

Table 2.4: Breast Cancer dataset feature names and descriptions.

Feature	Description
age	Discretized subject age in years
early meno	Whether the subject reach menopause early
pre-meno	Whether the subject is pre-menopausal
tumor size	Size of breast cancer tumor in mm
aux-nodes	# auxiliary lymph nodes containing metastatic cancer
in node cap	Whether metastatic tumors are encased in lymph node capsule
deg-malig	Histological degree of the tumor malignancy (range 1-3)
breast	Which breast cancer resides in
breast quadrant	Quadrant of breast cancer resides in
rad therapy	Whether the subject received radiation therapy

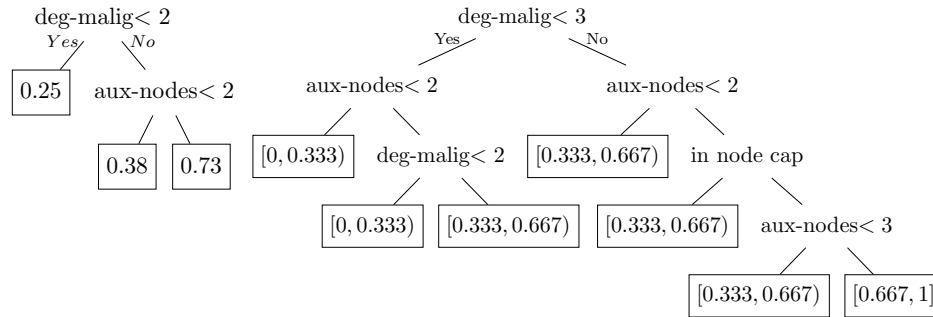


Figure 2.7: Left: CART fit to Breast Cancer dataset. Right: HART built from HAL fit to Breast Cancer dataset. Feature space is restricted to subjects aged 20-29 having tumors greater than 5mm in diameter. Algorithm 2 was applied with  $Q = 3$ .

degree of malignancy and the number of auxiliary lymph nodes containing metastatic breast cancer are needed to make a prediction about risk for breast cancer recurrence. However, the HART built using the same data shows a much more complex function and has much higher performance. The right panel of Figure 2.7 shows a Sub-HART visualized for subjects aged 20 – 29 having tumors greater than 5 mm in diameter. We have binned the predicted probabilities into three intervals:  $[0, 0.333)$ ,  $[0.333, .667)$ , and  $[0.667, 1]$ . This model implies a higher degree of heterogeneity in the predicted probability that relies on more features and values. We could investigate any of the terminal regions in Figure 2.7 further without binning the predictions. Alternatively, we could restrict to subjects older than 29 to see how the function changes with age.

If we are interested in using HART to make binary decisions based on a probability threshold, we can simplify the overall tree and visualize it without restricting the feature space. Figure 2.8 visualizes the full HART with 1 displayed in regions where predicted probabilities are above 0.5 and 0 displayed otherwise. Again, we see that increased performance is associated with a much more complex function. However, we can gain insights into HART’s structure that could motivate different visualizations or even further study. Consider the role of menopause status in the prediction. Figure 2.8 suggests that having reached menopause before age 40 is associated with lower risk for breast cancer recurrence versus having reached it after 40 or being pre-menopause.

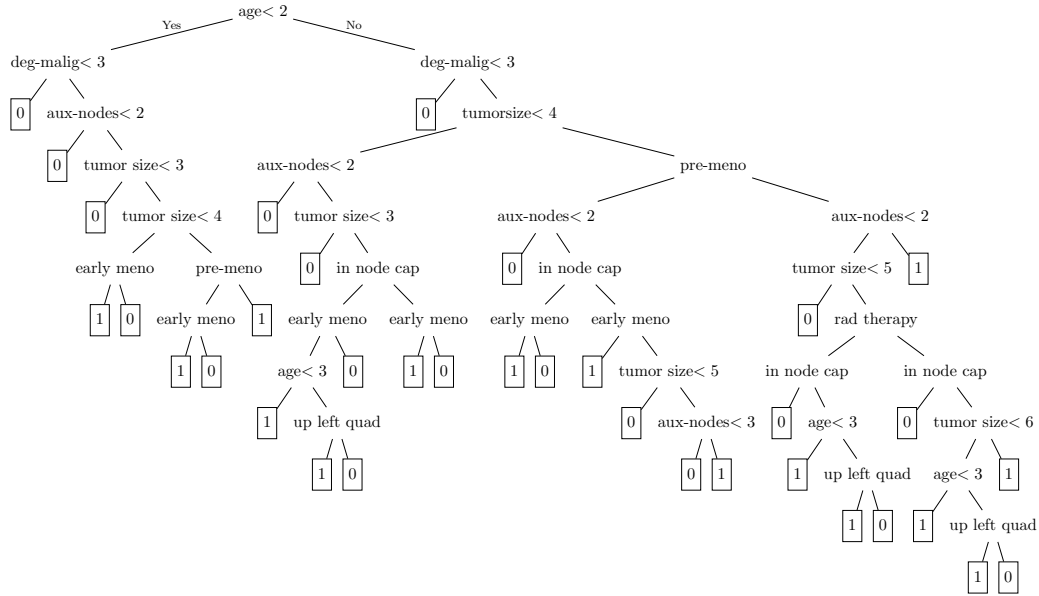


Figure 2.8: HART representing HAL fit to Breast Cancer data. Algorithm 2 has been applied with  $Q = 2$  bins, equivalent to thresholding the predicted probability of recurrence at 0.5.

This association is consistent with breast cancer research [33].

### 2.8.3 Complexity

We examined the time it took to apply Algorithms 1, 2, and 3 to both the Breast Cancer and Drugs datasets. After fitting HAL to the Breast Cancer data, there were 15 split candidates. The full HART constructed from the model fit was constructed in 0.752 seconds. Binning and Aggregating the predictions in the resulting tree took 0.015 seconds each. Applying the algorithms to the Cardio dataset took significantly longer. After fitting HAL to the Cardio data, there were 104 candidate splits. Constructing the full HART took 20.210 minutes. Binning and Aggregating the predictions in the resulting tree took 1.103 and 1.215 minutes respectively. All computations were carried out on an Apple Macbook Air containing an ARM processor with 8GB of RAM.

## 2.9 Discussion

In this paper we have presented a tool for post-hoc interpretation of the Highly Adaptive Lasso. HAL has the potential to learn more complex functions than CART without overfitting the data. HART, via Algorithms 1, 2, and 3, provides methodology for understanding these complex functions. An additional advantage of HART is that it allows one to tailor visualizations to the needs of the problem via the splitting policy  $\pi$  and Algorithms 2. In general, we recommend that HART be used less as a static tool for visualizing simple decision making processes and more as a dynamic way to visualize and understand a complex decision making process. In the future, it would be of interest to apply HART to problems that require transparent decision making such as medical treatment assignment and fair social policy design.

## Chapter 3

# Highly Adaptive Treatment Trees: interpretable estimation of heterogeneous treatment effects and treatment policies

### 3.1 Introduction

Methods for the estimation of heterogeneous treatment effects have been widely studied in recent years [28, 26, 42]. The development of such methods is vital in fields like healthcare where treatments may have different effects when administered to people with different characteristics. Developing treatment policies based on population-averaged effects could lead to unnecessary or even harmful treatment decisions for certain individuals. Ideally, we would like to derive optimal guidelines for practitioners in recommending the best treatment based on the clinical presentation of the patient.

A common estimand for addressing this question is the Conditional Average Treatment Effect (CATE), which measures the difference in counterfactual outcomes

under two treatments conditional on a set of covariates. A natural extension of the CATE is the Optimal Treatment Policy (OTP) which gives the treatment that will maximize the counterfactual outcome given a set of covariates. The OTP can be easily estimated given an estimate of the CATE.

One popular set of methods for estimating the CATE that has emerged is called Meta-Learning. Meta-Learning encompasses methods that repurpose off-the-shelf machine-learning algorithms for the estimation of the CATE. These include the S, T, X, R, and DR learners [31, 36, 30]. Athey and Imbens [5] introduced Causal Trees, a method for CATE estimation that involve recursive partitioning of the covariate space. That partitioning can then be represented as an interpretable decision tree. They extended this method to Causal Forest [56], which improves performance but loses interpretability.

With the exception of Causal Trees, there are limited nonparametric methods for CATE estimation that emphasize interpretability. However, interpretability in such estimates is often of paramount importance, especially if they are to be used to guide treatment decisions. If an estimated policy suggests to a doctor that a patient should (not) be treated, the doctor should be able to discern *why* that decision was rendered. If the policy is a black box, it is impossible to tell what characteristics the decisions are based on and whether those decisions are fair and logical.

In this paper, we introduce methods for CATE and OTP estimation that are both interpretable and high performing. Our methods combine the S and DR learning Meta-Learning frameworks with the Highly Adaptive Lasso (HAL) algorithm. We are able to show that the resultant estimates achieve fast regret rates. Previous work has also showed that HAL regression fits can be represented as trees [37]. We adapt those representation algorithms for the cases of CATE and OTP estimation.

Our contributions can thus be summarized as new algorithms for developing flexible, yet interpretable estimates of heterogeneous treatment effects and OTP. An additional

strength of our proposed methods is that we allow for any number of treatment arms and estimation based on any subset of covariates.

## 3.2 Problem Setup and Estimation Frameworks

### 3.2.1 Setup

Assume we observe an independent and identically distributed sample of observations  $O_i = (W_i, A_i, Y_i) \sim P_0 \in \mathcal{M}$  where  $P_0$  is an unknown distribution that lies in non-parametric model  $\mathcal{M}$ .  $W \in \mathbb{R}^p$  are covariates,  $A \in \mathcal{A} = \{0, 1, \dots, K\}$  is a treatment, and  $Y \in \mathbb{R}$  is a real-valued outcome. Further, let  $V \subseteq W$  be some subset of covariates with  $V \in \mathbb{R}^q$  for  $q \leq p$ . For each  $a \in \mathcal{A}$ , we define a counterfactual data unit  $X_i(a) = (W_i, Y_i(a)) \sim P_{0,a}$  corresponding to the data that would have been observed on unit  $i$  if, possibly counter to fact, the unit had been assigned treatment  $a$ . The Conditional Average Treatment Effect (CATE) is defined as  $\psi_{0,a}(v) := E_{0,a}[Y(a) | V = v] - E_{0,a}[Y(0) | V = v]$ , where for a  $P_{0,a}$ -measurable functions  $f$ , we write the *expectation* of  $f(O)$  as  $E_{0,a}[f(O)] = \int f dP_{0,a}$ . We note that our notation for the CATE explicitly makes level 0 of the treatment a referent level against which all other treatments are compared and we define  $\mathcal{A}_0 := \mathcal{A} \setminus \{0\}$ .

The CATE quantifies the expected difference in counterfactual outcome under treatment  $a \in \mathcal{A}_0$  versus treatment 0 for data units with  $V = v$ . Also of interest is the Optimal Treatment Policy (OTP),  $d_{0,\cdot}(v) := \operatorname{argmax}_{a \in \mathcal{A}} E_{0,a}[Y(a) | V = v]$ . Given an estimate  $\hat{\psi}(a | v)$  of  $\psi_0(a | v)$ , we can construct an estimate of the OTP as

$$\hat{d}(a | v) := \begin{cases} 0 & \text{if } \hat{\psi}(a | v) < 0 \quad \forall a \in \mathcal{A}_0 \\ \operatorname{argmax}_{a \in \mathcal{A}_0} \hat{\psi}(a | v) & \text{else} \end{cases}.$$

Estimation of the CATE and OTP require a set of causal assumptions that allow

us to rewrite the parameter without unobservable counterfactuals. One set of such assumptions is as follows.

1. Positivity:  $P_0\{0 < P_0(A = a | W) < 1\} = 1 \forall a \in \mathcal{A}$
2. No confounding:  $Y(a) \perp A | W \quad \forall a \in \mathcal{A}$
3. Consistency:  $Y_i = \sum_{a \in \mathcal{A}} \mathbb{1}_a(A_i) Y_i(a)$
4. No interference: the counterfactual outcome of each individual under each treatment does not depend on treatments assigned to other individuals in the population

If these assumptions are satisfied the CATE is *identified* by a parameter of the distribution of the observed data  $P_0$ . We define  $\mu_0(a, w) := E_0[Y | A = a, W = w]$ , where  $E_0$  denotes expectation under  $P_0$ . The CATE is identified by  $\psi_0(a | v) = E_0[\mu_0(a, W) - \mu_0(0, W) | V = v]$ . The object  $\mu_0$  is referred to as the *outcome regression*. Below, we also discuss the *propensity score* for treatment  $a$ ,  $\pi_0(a | w) := P_0(A = a | W = w)$ .

### 3.2.2 Meta-Learning With Two Treatment Arms

This CATE estimation problem is commonly presented in the case of two treatment arms,  $\mathcal{A} = \{0, 1\}$ , in which case the target parameter for estimation is  $\psi_0(1 | \cdot)$ , a function only of covariates  $V$ . A common approach to estimation of  $\psi_0(1 | \cdot)$  is the S-Learner [31], so called because it involves a single outcome regression estimator. The estimate  $\hat{\mu}$  of  $\mu_0$  can be obtained using any regression technique. For example, many such techniques estimate  $\mu_0$  by minimizing an empirical risk criteria based on a loss  $(\psi, o) \rightarrow L(\mu)(o)$ . We focus throughout on results for squared-error loss with the understanding that other loss functions could be used. For the purpose of learning  $\mu_0$ , squared error loss may be written  $L(\mu)(o) := \{y - \mu(a, v)\}^2$ .



Regardless of the learning approach adopted to generate  $\hat{\mu}$ , the S-Learner CATE estimate can be obtained post-hoc. If  $V = W$ , then the estimate is  $\hat{\psi}_S(1 | w) := \hat{\mu}(1, w) - \hat{\mu}(0, w)$ . If instead,  $V \subset W$ , then a second-stage of learning is required. In particular, we define a pseudo-outcome  $D_{\hat{\mu}}(O_i) := \hat{\mu}(1, W_i) - \hat{\mu}(0, W_i)$  and regress this pseudo-outcome on covariates  $V$  to obtain the CATE estimate  $\hat{\psi}_S(1 | v) := \hat{E}[D_{\hat{\mu}}(O) | V = v]$ . Again, the regression learning technique is interchangeable. For example, we could minimize empirical risk based on squared error loss  $L_S(\psi)(o) = \{D_{\mu}(o) - \psi(1 | v)\}^2$ .

An alternative strategy sometimes adopted is to split the data and regress  $Y$  on  $W$  separately for each treatment group to obtain estimates  $\hat{\mu}(0, w)$  and  $\hat{\mu}(1 | w)$  of  $\mu_0(1, w)$  and  $\mu(0, w)$  respectively. This method is referred to as the T-Learner [31]. Again, if  $V = W$ , we can estimate the CATE by subtracting the two regression estimates:  $\hat{\psi}_T(1 | w) = \hat{\mu}(1, w) - \hat{\mu}(0, w)$ . If  $V \subset W$ , this difference constitutes a pseudo-outcome which we further regress  $V$  to obtain our CATE estimate. The T-learner typically has lower performance due to the lack of information sharing across treatment arms when estimating the outcome regression [30]. Thus, it is not considered further here.

van der Laan [50] describes a Doubly Robust (DR)-Learner, which involves estimation not only of the outcome regression  $\mu_0$ , but also the propensity score  $\pi_0$ . The two nuisance parameter estimates  $\hat{\mu}$  or  $\hat{\pi}$  are both used to construct the pseudo-outcome:

$$D_{\hat{\mu}, \hat{\pi}}(O_i) = \left[ \frac{2A_i - 1}{\hat{\pi}(A_i | W_i)} \right] \{Y_i - \hat{\mu}(A_i, W_i)\} + \hat{\mu}(1, W_i) - \hat{\mu}(0, W_i) . \quad (3.1)$$

$D_{\hat{\mu}, \hat{\pi}}(O)$  is then regressed on  $V$  to obtain the CATE estimate  $\hat{\psi}_{DR}(1 | v) := \hat{E}[D_{\mu, \pi}(O) | V = v]$ .

We can obtain this estimate, for example, via empirical risk minimization based on a loss  $L_{DR}(\psi)(o)$ . We focus on squared-error loss,  $L_{DR}(\psi)(o) = \{D_{\mu, \pi}(o) - \psi(1 | v)\}^2$ .

This estimator is referred to as *doubly robust* because  $E_0[D_{\mu, \pi}(O) | V = v] = \psi_0(1 | v)$

if either  $\mu = \mu_0$  or  $\pi = \pi_0$ . Thus, we generally expect that  $\hat{\psi}_{\text{DR}}$  will be consistent for  $\psi_0$  if either  $\hat{\mu}$  is consistent for  $\mu_0$  or  $\hat{\pi}$  is consistent for  $\pi_0$ .

### 3.2.3 Meta-Learning with Multiple Treatment Arms

These methods readily extend to the case when  $|\mathcal{A}| > 2$ . We describe analogs of each of the Meta-Learners when there are multiple treatment arms. First, we must choose a reference treatment against which we will compare all others. This may be a control group or it may just be one arm chosen arbitrarily. Throughout this paper, we will let  $A = 0$  be the reference treatment.

In the case when  $V = W$ , the S-Learner procedure is largely unchanged. We regress  $Y$  on  $A$  and  $W$  to obtain  $\hat{\mu}(a, w)$ , which can then be used to construct  $\hat{\psi}_{\text{S}}(a | w) := \hat{\mu}(a, w) - \hat{\mu}(0, w)$  for each  $a \in \mathcal{A}_0$ .

In the case when  $V \subset W$ , the S-Learner requires estimation of  $\mu_0(a, w)$  and second stage regression as before. For a given  $a$ , we could learn  $\hat{\psi}_{\text{S}}(a | v)$  by empirical risk minimization, based on the loss  $L_{\text{S},\mu}^a(\psi)(o)$ , such as mean squared-error,  $L_{\text{S},\mu}^a(\psi)(o) = \{D_{\mu}^a(o) - \psi(a | v)\}^2$ .

Alternatively, we could learn each  $\hat{\psi}(a | v)$  simultaneously by considering a summed loss function  $L_{\text{S},\mu}(\psi) := \sum_{a \in \mathcal{A}_0} \omega(a) L_{\text{S},\mu}^a(\psi)$ . Here,  $\omega(a)$  is any positive weight. We index the loss by  $\mu$  to enable distinction between the regression using the estimated nuisance parameters and the regression using the true nuisance parameters.

To minimize this summed loss, we create  $|\mathcal{A}| - 1$  pseudo-outcomes for each individual, with the form of the pseudo-outcome for the  $i^{\text{th}}$  individual associated with  $\psi(a | v)$  calculated as

$$D_{\hat{\mu}}^a(O_i) = \hat{\mu}(a, W_i) - \hat{\mu}(0, W_i) . \quad (3.2)$$

This results in a modified dataset with  $n \cdot (|\mathcal{A}| - 1)$  rows. We then regress the stacked pseudo-outcome values on covariates plus one-hot-encoded  $a$  values.

To construct the DR-Learner with multiple treatments, we similarly create  $|\mathcal{A}| - 1$  pseudo-outcomes for each individual. In this case, the pseudo-outcome for the  $i^{\text{th}}$  individual associated with  $\psi_0(a | v)$  is

$$D_{\hat{\mu}, \hat{\pi}}^a(O_i) = \left[ \frac{\mathbb{1}_a(A_i)}{\hat{\pi}(a | W_i)} - \frac{\mathbb{1}_0(A_i)}{\hat{\pi}(0 | W_i)} \right] \cdot \{Y_i - \hat{\mu}(A_i, W_i)\} + \hat{\mu}(a, W_i) - \hat{\mu}(0, W_i) . \quad (3.3)$$

Note that if an individual  $i$  has  $A_i = a \neq 0$ , (3.3) reduces to

$$D_{\hat{\mu}, \hat{\pi}}^a(O_i) = \frac{\{Y_i - \hat{\mu}(A_i, W_i)\}}{\hat{\pi}(a | W_i)} + \hat{\mu}(a, W_i) - \hat{\mu}(0, W_i) .$$

If instead  $A_i = 0$ , then (3.3) reduces to

$$D_{\hat{\mu}, \hat{\pi}}^a(O_i) = -\frac{\{Y_i - \hat{\mu}(A_i, W_i)\}}{\hat{\pi}(0 | W_i)} + \hat{\mu}(a, W_i) - \hat{\mu}(0, W_i) .$$

Finally, if individual  $i$  has  $A_i \notin \{0, a\}$ , (3.3) reduces to

$$D_{\hat{\mu}, \hat{\pi}}^a(O_i) = \hat{\mu}(a, W_i) - \hat{\mu}(0, W_i) .$$

This pseudo-outcome form is doubly robust as established below.

**Theorem 3.2.1.** *For each  $a \in \mathcal{A}_0$ ,  $E_0[D_{\hat{\mu}, \hat{\pi}}^a(O) | V = v] = \psi_0(a | v)$  if either  $\mu = \mu_0$  or  $\pi = \pi_0$ .*

The proof of Theorem 3.2.1 can be found in section B.1 of the Appendix. As with the S-Learner, the CATE estimate of is obtained by regressing the stacked pseudo-outcomes  $D_{\hat{\mu}, \hat{\pi}}^a(O)$  against the covariates  $V$  and one-hot-encoded  $a$  values.

### 3.2.4 Causal Trees and Forests

Athey and Imbens [5] developed Causal Trees for estimating the CATE in a way that

can be represented as a decision tree. The method involves recursively partitioning the feature space into regions that have constant treatment effect. They employ what they call “honest estimation” of the CATE by using separate sets of data to partition the feature space and estimate the treatment effects within the resulting partitions. Partitioning is done in a greedy way to minimize an estimate of the expected mean square error of the treatment effect. The optimal depth of the tree is selected via cross-validation, and the tree is then pruned to the selected depth. Finally, treatment effects are estimated within each region of the partition using a held out dataset. An extension of this method, Causal Forests, applies bootstrap aggregation to Causal Trees in order to obtain smoother estimates of the CATE [56]. We compare the performance of both Causal Trees and Causal Forests to our method in section 3.5.

### 3.3 The Highly Adaptive Lasso

#### 3.3.1 Background

Our method centers on use of the Highly Adaptive Lasso (HAL) for regression in the final stage of the chosen Meta-Learning algorithm. We briefly introduce HAL and its properties in the context of a general supervised learning task. Later, we extend these properties to estimation of the CATE.

HAL is a general approach for estimation of functional parameters using regularized empirical risk minimization. The theory of HAL is built around two assumptions about the underlying functional parameter: (i) that it is right-continuous with left-hand limits and (ii) that it has finite variation norm. The assumption of smoothness defined by a function’s variation norm has been long studied in the statistical learning literature [17].

Benkeser and Van Der Laan [8] and van der Laan [49] discuss an approach for finding the minimum loss-based estimator (MLE) in the class of functions with

variation norm smaller than a finite constant. The MLE is computed based on the fact that any cadlag function with finite variation norm can be arbitrarily well-approximated by a tensor product of indicator basis functions. If learning, for example, the mean of a random variable  $Z$  conditional on a set of variables  $C$ , the functional form of the estimate is  $\hat{\beta}_0 + \hat{\beta}_C^\top b(c)$ , where  $b(c)$  is an  $n$ -length vector with  $i$ -th entry equal to  $\mathbb{1}_{\geq C_i}(c)$ . As the dimension of  $X$  increases, we include tensor product basis functions. For example, in the bivariate case  $C = (C_1, C_2)$  the functional form is  $\hat{\beta}_0 + \hat{\beta}_{C_1}^\top c_1(x) + \hat{\beta}_{C_2}^\top b_2(c) + \hat{\beta}_{C_1, C_2}^\top b_1(c)b_2(c)$ , where for  $j = 1, 2$ , the  $i$ -th entry of  $b_j$  is equal to  $\mathbb{1}_{\geq C_{j,i}}$ . The idea generalizes to arbitrary  $p$  with at most  $n(2^p - 1)$  basis functions included.

Due to the construction of the basis functions, the  $L_1$ -norm of the  $\hat{\beta}$  coefficients equals the variation norm of the estimate. Thus, HAL estimates can be achieved by making use of regularized risk minimization algorithms that have been made popular by the lasso algorithm [47]. Benkeser and Van Der Laan [8] showed that HAL estimates converge in terms of regret at a rate of  $o_p(n^{-[1/4+1/\{8p+1\}]})$ , where  $p$  is the dimension of the feature vector. The authors also demonstrated competitive performance of HAL with other commonly used machine learning frameworks across a variety of empirical experiments.

### 3.3.2 Highly Adaptive Regression Trees

Previous work has shown that regression fits generated by the Highly Adaptive Lasso algorithm can be represented as trees [37] referred to as Highly Adaptive Regression Trees (HART). HAL itself results in a non-recursive partitioning of the feature space. The HART algorithm involves generating a set of candidate values implied by the HAL fit and using those values to recursively partition the feature space. In each sub-region of the feature space, the set of candidate values is appropriately pruned to minimize the amount of redundancy in the final tree representation.

These tree representations make HAL an ideal candidate for the estimation of CATEs and, by extension, treatment policies.

## 3.4 CATE Estimation with HAL

### 3.4.1 Theoretical Guarantees for S and DR learners

Let  $\hat{\mu}$  and  $\hat{\pi}$  be given estimates of  $\mu_0$  and  $\pi_0$ . Further, let  $\hat{R}_p^* := o_p(n^{-[1/4+1/\{8p+1\}]})$ , which is the regret rate of the HAL estimator when estimating a function of bounded variation. Finally, for general  $P_0$ -measurable function  $f$  of  $O$ , we define the  $L_m(P_0)$  norm of  $f$  to be  $\|f\|_{m,P_0} := (\int f^m dP_0)^{1/m}$ .

**Theorem 3.4.1.** *Let  $\hat{\psi}_S(v)$  be an estimate of  $\psi_0(v)$  constructed with the S-learner algorithm. If  $V = W$  and HAL is used to construct the estimator, then*

$$\int \left[ L_S(\hat{\psi}_S) - L_S(\psi_0) \right] dP_0 = \hat{R}_p^{*2}.$$

*If  $V \subset W$ , and HAL is used to perform the second stage regression, then*

$$\begin{aligned} & \int \left[ L_{S,\mu_0}(\hat{\psi}_S) - L_{S,\mu_0}(\psi_0) \right] dP_0 \\ &= \sum_{a \in \mathcal{A}_0} o_p(\|\hat{\mu}(a, \cdot) - \mu_0(a, \cdot)\|_{2,P_0}) \cdot \hat{R}_q^* + \hat{R}_q^* \end{aligned}$$

A proof of Theorem 3.4.1 can be found in Appendix section B.2. Recall that when  $V = W$ , the S-Learner only involves estimation of the outcome regression  $\mu_0(a, v)$ . Intuitively then, when HAL is used to estimate  $\mu_0(a, v)$ , the S-Learner CATE estimate converges in terms of regret to the true CATE at a rate of  $R_{n,p}^*$ . When  $V \subset W$ , the S-Learner requires estimation of  $\mu_0$  as a nuisance parameter. We consider the rate at which the S-Learner CATE estimate constructed with the true outcome regression converges in terms of regret to the true CATE. We see that when HAL is used for

the second stage regression this rate depends both on the regret rate of the outcome regression estimator and that of the HAL pseudo-outcome regression estimator.

We have the following result for the DR learner.

**Theorem 3.4.2.** *Let  $\hat{\psi}_{DR}(v)$  be an estimate of  $\psi_0(a | v)$  constructed with the DR-learner algorithm. If HAL is used for the final stage regression, then*

$$\begin{aligned} & \int \left[ L_{DR, \mu_0, \pi_0}(\hat{\psi}_{DR}) - L_{DR, \mu_0, \pi_0}(\psi_0) \right] dP_0 \\ &= \hat{R}_q^* + \sum_{a \in \mathcal{A}_0} [o_p(\|\hat{\pi}(a | \cdot) - \pi_0(a | \cdot)\|_{4, P_0}) \\ & \quad \cdot o_p(\|\hat{\mu}(a, \cdot) - \mu_0(a, \cdot)\|_{4, P_0})] \cdot \hat{R}_q^* . \end{aligned}$$

Theorem 3.4.2 gives a similar result for the DR-Learner as with the S learner, but where the rate now depends on the regret rates of the estimates of  $\pi_0(a | w)$  and  $\mu_0(a, w)$ . A proof can be found in Appendix section B.3. The theorem indicates that the DR-Learner may enjoy advantages over the S-Learner. Consider a scenario in which the CATEs and propensity score are easier to learn than the outcome regression. When using the S-Learner, the overall regret rate may be slowed by poor estimation of the outcome regression. However, the rate of the DR-Learner involves the *product* of the regret rates of the propensity score *and* the outcome regression. Thus, the ability to quickly learn the propensity score may mitigate the impact of poor outcome regression estimation, thereby achieving an overall rate closer or equal to  $\hat{R}_q^*$ . This scenario is plausible in several settings including when the data are generated from randomized experiments.

Moreover, the pseudo-outcome regression may be easier to learn than the outcome regression itself simply because the contrast function  $\mu_0(a, \cdot) - \mu_0(0, \cdot)$  may be simpler or smoother function than  $\mu_0$  itself. In the case of HAL, the most relevant measure of smoothness is the variation norm of the respective functions. In the case when  $V \subset W$ , we may also expect that  $\psi_0$  will have lower variation norm than  $\mu_0(a, w) - \mu_0(0, w)$ , as

---

**Algorithm 4:** CATE Tree Construction
 

---

**Input:** CATE Estimate  $\hat{\psi}(a | v)$   
**if**  $|\mathcal{A}| == 2$  **then**  
   Apply HART to  $\hat{\psi}(1 | v)$   
   Collapse regions with identical terminal nodes  
**else**  
   Apply HART to  $\hat{\psi}(a | v)$  enforcing  $A$  appears first  
   Collapse regions with identical terminal nodes  
**end if**

---

it conditions on a subset of variables, thereby averaging over covariates across which there may be substantial variation in  $\mu_0$ .

### 3.4.2 Tree Representations

When HAL is used as the final stage regression estimator for either the S or DR-learning procedures, the resulting CATE and OTP estimates can be represented using trees. Both of those procedures rely on extensions of the original HART algorithm [37]. In the case when there are two treatments, we can simply apply the original HART algorithm to build a CATE representation and apply a threshold of 0 to the terminal nodes in that tree in order to obtain a OTP estimate representation. In the case when there are more than two treatments, we must account for the treatment indicator variables in the tree. The HART algorithm allows the user to select the order that variables appear in the tree. Typically, we aim to find an ordering that minimizes the size of the tree. However, in this case, it is convenient to enforce that the treatment indicator variables appear first in the tree so that the effects for each treatment arm relative to the reference treatment are represented by distinct regions of the tree. Full details for representing the CATE estimate as a tree are shown in algorithm 4.

When representing the OTP in the case of more than two treatments, we can enforce that the treatment indicator variables appear last in the tree, just before the



---

**Algorithm 5:** Treatment Policy Tree Construction
 

---

**Input:** CATE Estimate  $\hat{\psi}(a | v)$   
**if**  $|\mathcal{A}| == 2$  **then**  
   Apply HART to  $\hat{\psi}(1 | v)$   
   Set terminal node values to 1 if  $\hat{\psi}(1 | v) > 0$ , 0 else  
   Collapse regions with identical terminal nodes  
**else**  
   Apply HART to  $\hat{\psi}(a | v)$  enforcing  $A$  appears last  
   Replace treatment nodes and terminal nodes with  $\operatorname{argmax}_{a \in \mathcal{A}} \hat{\psi}(a | v)$  in each region  
   Collapse regions with identical terminal nodes  
**end if**

---

terminal nodes in the regions where they are necessary to split on. We can then replace the treatment indicator variable nodes and the terminal nodes with terminal nodes displaying the treatment that has the highest expected counterfactual outcome. Full details for representing the OTP estimate can be found in algorithm 5.

## 3.5 Simulations

We performed simulations to evaluate the performances of both the S and DR-Learner versions of HATT. We also compared their performances to those of Causal Trees, Causal Forests, and the S and DR-Learners built with other machine learning algorithms.

### 3.5.1 Data generating process

Simulations were conducted under three settings, *linear*, *polynomial*, and *sinusoidal*, so named for the chosen functional form of the true CATEs. In each setting we considered a continuous outcome  $Y$  and a binary treatment  $A \in \{0, 1\}$ . Additional simulations with  $|\mathcal{A}| = 3$  treatment arms are included in Appendix section ???. Counterfactual outcome values were calculated by adding standard normal random noise to the specified true outcome regression. Treatment values were assigned by simulating from

a Bernoulli distribution with probability of  $A = 1$  set by a chosen propensity score function. We simulated data at sample sizes  $n = 50, 100, 200, 500,$  and  $1000$ . We generated data, fit models to estimate the CATE and OTP and evaluated performances using metrics described in 3.5.2. This process was replicated 1000 times at each sample size, and results were averaged. Below are descriptions of each setting.

*Linear setting*

Here we considered  $p = 5$  covariates, and set the task of estimating the treatment effect conditional on all five. All covariates were simulated independently from standard normal distributions. The true outcome regression was  $\mu_0(a, w) = 4a + 5aw_1 + w_2 - 3w_3 + 5w_3 * w_4$ . Consequently, the true CATE was  $\psi_0(1 | w) = 4 + 5w_1$ . We specified the true propensity score as

$$\pi(a | w) = \frac{1}{1 + \exp(-.5 * w_1 + .2w_2 - .3w_3 + w_5)} .$$

*Polynomial setting*

Here we again considered  $p = 5$  covariates and aimed to estimate the CATE conditional on all 5. This time we specified a polynomial function of covariates for the outcome regression:  $\mu_0(a, w) = 5a + 2aw_1 + w_2 - 3aw_3w_5^2 + 2aw_4^3$ . This resulted in a CATE of  $\psi_0(1 | w) = 5 + 2w_1 - 3w_3w_5^2 + 2w_4^3$ . We set the propensity score to be the same as the logistic regression specified in the linear setting.

*Sinusoidal setting*

Here we considered  $p = 1$  covariate,  $W_1$  simulated from a standard normal distribution. We set the true outcome regression as  $\mu_0(a, w) = \sin\{(2a - 1)cw_1\}$ , for  $c = 1, 10$ . This resulted in CATEs of the form  $\psi_0(1 | w) = \sin(cw_1) - \sin(-cw_1)$ . The propensity score was set to be constant  $\pi(a | w) = 0.5$  for all  $w$ .

### 3.5.2 Evaluation

To evaluate each estimation method, we generated independent test datasets of size  $N = 1e5$ . These data sets were first used to approximate the  $L_2$  norm of each estimated CATE:  $E[(\hat{\psi}(1 | w) - \psi_0(1 | w))^2]$ , a measure of global fit of  $\hat{\psi}$  to  $\psi_0$ . Next, we used the test data to estimate the *value* of the learned policy. That is, we can simulate counterfactuals  $Y(a)$  for all  $a \in \mathcal{A}$  for observations in the test datasets. We then evaluate the learned policy recommendation for each test dataset observation. The value of the policy is the average value of the counterfactual treatment recommended by the learned policy.

### 3.5.3 Linear CATE Results

DR-Learning of the CATE with HAL as the second stage algorithm and S-Learning with a Super Learner [51, 57, 11] ensemble performed best both in terms of policy value and  $L_2$  norm (Figure 3.1). Both methods yielded high initial policy values, showed fast convergence to the optimal policy value, and yielded the smallest  $L_2$  norms at all sample sizes. The performance of S-Learning with HAL is lower, which may be explained by the fact that the DR-Learner with HAL uses Super Learner, which may be providing a more accurate outcome regression estimate than HAL itself. Causal Tree, the only other interpretable method of estimation, showed significantly lower performance at all sample sizes than HAL-based methods.

### 3.5.4 Polynomial CATE Results

In this setting, DR-Learning with HAL and S-Learning with Super Learner still tended to have the highest performers, though the relative improvements over other methods was less marked than in the linear setting (Figure 3.2). As with the linear setting, the benefit of Doubly-Robust estimation is still evident, though less pronounced.

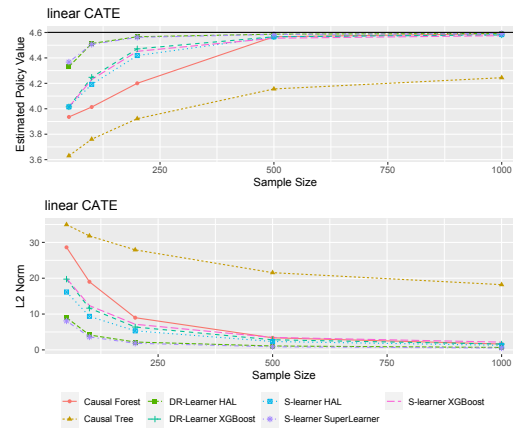


Figure 3.1: Simulation results under the Linear setting. The DR-Learner with HAL used Super Learning to construct nuisance parameter estimates. Results at each sample size are averaged over 1000 replications.

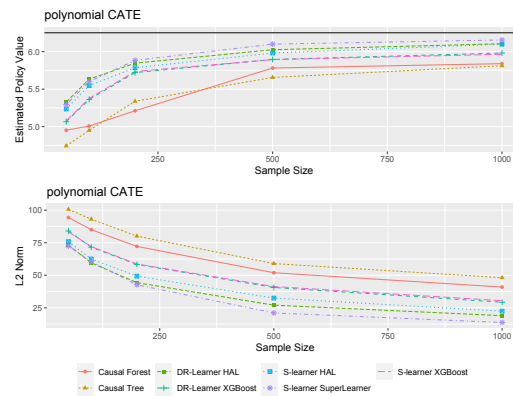


Figure 3.2: Simulation results under the Polynomial setting. The DR-Learner with HAL used Super Learning to construct nuisance parameter estimates. Results at each sample size are averaged over 1000 replications.

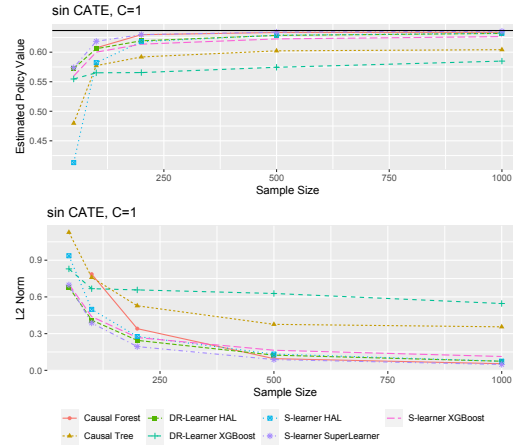


Figure 3.3: Simulation results under the Sinusoidal setting with lower variation norm. The DR-Learner with HAL used Super Learning to construct outcome regression estimates. All propensity score estimates were constructed with simple means. At  $n = 50$ , Causal Forest showed abnormally low performance and is omitted from the plot to keep other learners distinguishable. Results at each sample size are averaged over 1000 replications.

### 3.5.5 Sinusoidal CATE Results

When  $C = 1$  (indicating a lower variation norm of the outcome regression), Causal Forest, and S-Learner with Super Learner were the highest performers, while the DR-Learner with HAL shows marginally lower performance (Figure 3.3). When  $C$  is increased to 10, DR-Learning with HAL and DR-Learning with XGBoost tended to perform best (Figure 3.4). At the lower sample sizes, S-learning with HAL shows poor performance. This drop is likely due to the increased variation norm of the true outcome regression. This reinforces the idea that, in cases when HAL is not powerful enough to estimate the the CATE in single stage regression, it can be paired with other learning techniques, such as Super Learner to achieve superior performance while preserving interpretability of the final CATE estimate.

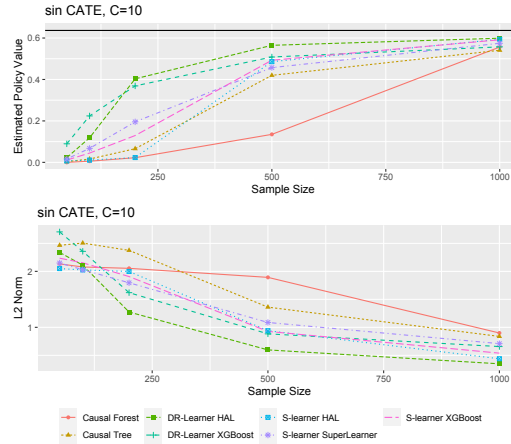


Figure 3.4: Simulation results under the Sinusoidal setting with higher variation norm. The DR-Learner with HAL used Super Learning to construct outcome regression estimates. All propensity score estimates were constructed with simple means. Results at each sample size are averaged over 1000 replications.

### 3.6 Data Analysis

We analyzed experimental data to identify the conditional effect of sending mailers with messages addressing ballot secrecy concerns on voter turnout in the 2010 Connecticut congressional general election [24]. Researchers worked with the Connecticut Secretary of State to send written messages to  $n = 3744$  registered voters who had not voted in the 2008 presidential election. There were two “treatment” arms. Some registrants received a letter containing information about voting and a message intending to assuage concerns about ballot secrecy. The other registrants received a letter containing only voting information. Several covariates were recorded for each subject including age, gender, political party (Republican or Democrat), number of members of household, and the town in which they resided. The researchers then recorded whether each individual had cast a vote in the 2010 election.

We employed the DR-Learner with the estimated propensity score given by the sample proportion in each treatment condition and the estimated outcome regression produced using a Super Learner ensemble. HAL was used to carry out the second stage regression, and we employed Algorithm 4 to build a tree representation (Figure

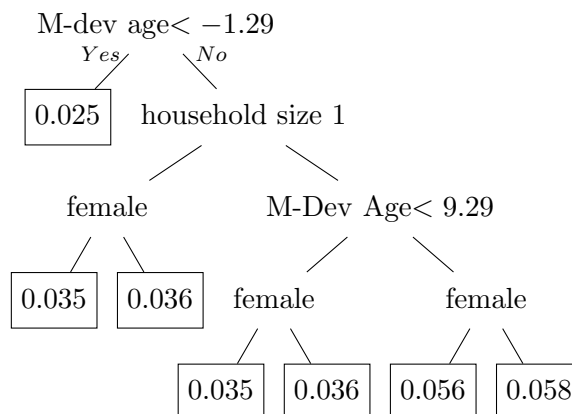


Figure 3.5: Tree representing the effect of receiving a mailer with a message meant to assuage fears about ballot secrecy. M-dev age represent the number of years above the median study age. The terminal nodes represent differences in estimated counterfactual probability of voting under treatment versus control.

3.5). As we can see there is a small amount of heterogeneity in the estimated effect of the treatment across different strata of the covariates. In general, the effect of a secrecy message seems lower for females than males. The effect also appears negatively correlated with age. Furthermore, there is an interaction between the subjects age and household size. Whether or not the subject was the sole member of their household only changed the treatment effect for subjects greater than 9.29 years above the median study age. All strata have positive estimated effects, and the resulting optimal treatment policy would be to send messages to assuage fears about ballot secrecy to everyone. However, such a tree could be useful in prioritizing segments of the population in a, resource-limited setting. For example, if an organization aiming to increase turnout could only send mailers to a fraction of potential voters, they could prioritize older registrants living in households with larger numbers of people.

### 3.7 Discussion

In this paper we have introduced a method for CATE and OTP estimation that empirically has strong performance and has desirable interpretability properties. The

empirical results seen in our experiments align with the regret rate theory that we establish. When paired with our adaptations of the tree representation algorithms in Nizam and Benkeser [37], the method provides an appealing framework for developing data-driven, interpretable decision making systems. Nevertheless, a potential downside of this work is that, while learned OTP may be interpretable, they may not satisfy fundamental notions of fairness. A concern is that the interpretability of an OTP may provide a false sense of security in terms of fairness. However, it is not clear that human observers will be adequate judges of the fairness of an OTP simply by studying the policy alone. This may be particularly true if the tree for the OTP is complex. In the future, we would like enhance our work by making use of the literature on algorithmic fairness, such as in Nabi et al. [35]. Unifying such work with our own could lead to the development of policies that are interpretable, optimal, and fair.



# Appendix A

## Chapter 1 Supplementary Material

### A.1 Notation

Here we define new notation to be used throughout Appendix A.

- Let  $P$  be a probability measure and  $f$  be a measurable function. We define  $Pf := \int f dP$ . Further, for two measures  $P_1$  and  $P_2$ , we can define  $(P_1 - P_2)f := \int f d(P_1 - P_2) := \int f dP_1 - \int f dP_2$ . For two measurable functions  $f_1$  and  $f_2$ , we define  $P(f_1 - f_2) := Pf_1 - Pf_2$ .
- In the proof of Theorem 1.3.1 we suppress the dependence of the AUPRC on the function  $\psi$  and the data  $X$  in notation by letting  $Z := \psi(X)$ .
- In the proof of Theorem 1.3.1, we write the AUPRC in a new form. Let  $Q_P(z, y) := \frac{y}{P(Y=1)} \cdot \frac{P(Z > z, Y=1)}{P(Z > z)}$ . Then we can define the AUPRC as  $\Phi(P) := \int \int Q_P(z, y) P(dz, dy) := PQ_P$ .
- In the proof of Theorem 1.3.1, we let  $F_{0,Z|Y=1}(z) := P(Z \leq z | Y = 1)$ .

## A.2 Theorem 1.3.1 Proof

Under the new notation defined in which the dependence on  $\psi$  is suppressed, we can refer to our NPML estimator using  $\Phi(P_n)$  and the true value using  $\Phi(P_0)$ . We begin by decomposing the difference between the estimate and the truth into several terms:

$$\begin{aligned}
\Phi(P_n) - \Phi(P_0) &= P_n Q_{P_n} - P_0 Q_{P_0} \\
&= P_n Q_{P_n} - P_0 Q_{P_0} \pm P_n Q_{P_0} \pm P_0 Q_{P_n} \pm P_0 Q_{P_0} \pm (P_n - P_0)(Q_{P_n} - Q_{P_0}) \\
&= (P_n - P_0)Q_{P_0} + P_0(Q_{P_n} - Q_{P_0}) + (P_n - P_0)(Q_{P_n} - Q_{P_0}) \\
&= (P_n - P_0)Q_{P_0} + P_0(Q_{P_n} - Q_{P_0}) + R_{n,1}
\end{aligned}$$

where  $R_{n,1} := (P_n - P_0)(Q_{P_n} - Q_{P_0})$ . We break this proof into two parts. In part (i) we will show that  $(P_n - P_0)Q_{P_0} + P_0(Q_{P_n} - Q_{P_0}) = \frac{1}{n} \sum_{i=1}^n D_{\Phi}(P_0)(O_i) + o_p(n^{-1/2})$ . In part (ii) we will show that  $R_{n,1} = o_p(n^{-1/2})$ , and the proof will be complete.

(i) Consider  $Q_{P_n} - Q_{P_0} = Q_{P_n}(z, y) - Q_{P_0}(z, y)$ . Plugging in the definitions of both terms, we have

$$Q_{P_n}(z, y) - Q_{P_0}(z, y) = y \left[ \frac{P_n(Z > z | Y = 1)}{P_n(Z > z)} - \frac{P_0(Z > z | Y = 1)}{P_0(Z > z)} \right].$$

We can rewrite this as

$$\begin{aligned}
&y \left[ \frac{P_n(Z > z | Y = 1)}{P_n(Z > z)} - \frac{P_0(Z > z | Y = 1)}{P_0(Z > z)} \right. \\
&\quad \pm \frac{P_n(Z > z, Y = 1)}{P_0(Y = 1)P_0(Z > z)} \pm \frac{P_0(Z > z = 1)P_n(Y = 1)}{P_0(Y = 1)P_0(Z > z)} \\
&\quad \left. \pm \frac{P_0(Z > z | Y = 1)}{P_0(Z > z)} \pm \frac{P_0(Z > z | Y = 1)P_n(Z > z)}{P_0(Z > z)} \right], \tag{A.1}
\end{aligned}$$

which can be rearranged to

$$\begin{aligned}
Q_{P_n}(z, y) - Q_{P_0}(z, y) &= \frac{1}{n} \sum_{i=1}^n \frac{1}{P_0(Z > z)} \frac{Y_i}{P(Y = 1)} [I(Z_i > z) - P_0(Z > z | Y = 1)] \\
&\quad - \frac{1}{n} \sum_{i=1}^n \frac{P_0(Z > z | Y = 1)}{P_0(Z > z)^2} [I(Z_i > z) - P_0(Z > z)] \\
&\quad + R_{n,2}.
\end{aligned} \tag{A.2}$$

The first term above comes from combining the positive portion of the third term and the negative portion of the fourth term in [ref]. The second term above comes from combining the positive portion of the fifth term and the negative portion of the sixth term of [ref].  $R_{n,2}$  is comprised of the remaining terms and is defined below:

$$\begin{aligned}
R_{n,2} &= \frac{P_n(Y = 1, Z > z)}{P_n(Y = 1)P_n(Z > z)} - \frac{P_0(Y = 1, Z > z)}{P_0(Y = 1)P_0(Z > z)} \\
&\quad - \frac{P_n(Y = 1, Z > z)}{P_0(Y = 1)P_0(Z > z)} + \frac{P_n(Z > z | Y = 1)P_n(Y = 1)}{P_0(Y = 1)P_0(Z > z)} \\
&\quad - \frac{P_0(Z > z | Y = 1)}{P_n(Z > z)} + \frac{P_0(Z > z | Y = 1)P_n(Z > z)}{P_0(Z > z)^2}
\end{aligned}$$

We can show by repeatedly adding, subtracting, and rearranging terms that  $R_{n,2} = o_p(n^{-1/2})$ . Our goal is to produce terms of the form  $\frac{A_n B_n}{C_n}$  where  $A_n = o_p(1)$ ,  $B_n = O_p(n^{-1/2})$ , and  $C_n \xrightarrow{p} c$  for some constant  $c$ . Then, by the Continuous Mapping Theorem,  $\frac{A_n}{C_n} = o_p(1)$ , and we can conclude that  $\frac{A_n B_n}{C_n} = o_p(1)O_p(n^{-1/2}) = o_p(n^{-1/2})$ . We begin by considering the first and third terms of  $R_{n,2}$  above:

$$\begin{aligned}
& \frac{P_n(Y = 1, Z > z)}{P_n(Y = 1)P_n(Z > z)} - \frac{P_n(Y = 1, Z > z)}{P_0(Y = 1)P_0(Z > z)} \\
= & \frac{P_n(Y = 1, Z > z)}{P_n(Y = 1)P_n(Z > z)} - \frac{P_n(Y = 1, Z > z)}{P_0(Y = 1)P_0(Z > z)} \pm \frac{P_n(Y = 1, Z > z)}{P_0(Y = 1)P_n(Z > z)} \\
= & P_n(Y = 1, Z > z) \left[ \frac{P_0(Y = 1) - P_n(Y = 1)}{P_0(Y = 1)P_n(Y = 1)P_n(Z > z)} \right] \\
& + P_n(Y = 1, Z > z) \left[ \frac{P_0(Z > z) - P_n(Z > z)}{P_0(Y = 1)P_0(Z > z)P_n(Z > z)} \right] \\
& \pm P_0(Y = 1, Z > z) \left[ \frac{P_0(Y = 1) - P_n(Y = 1)}{P_0(Y = 1)P_n(Y = 1)P_n(Z > z)} \right] \\
& \pm P_0(Y = 1, Z > z) \left[ \frac{P_0(Z > z) - P_n(Z > z)}{P_0(Y = 1)P_0(Z > z)P_n(Z > z)} \right] \\
= & \frac{[P_n(Y = 1, Z > z) - P_0(Y = 1, Z > z)][P_0(Y = 1) - P_n(Y = 1)]}{P_0(Y = 1)P_n(Y = 1)P_n(Z > z)} \\
& + \frac{[P_n(Y = 1, Z > z) - P_0(Y = 1, Z > z)][P_0(Z > z) - P_n(Z > z)]}{P_0(Y = 1)P_0(Z > z)P_n(Z > z)} \\
& + \frac{P_0(Y = 1, Z > z)[P_0(Y = 1) - P_n(Y = 1)]}{P_0(Y = 1)P_n(Y = 1)P_n(Z > z)} \\
& + \frac{P_0(Y = 1, Z > z)[P_0(Z > z) - P_n(Z > z)]}{P_0(Y = 1)P_0(Z > z)P_n(Z > z)} \\
= & o_p(n^{-1/2}) + \frac{P_0(Y = 1, Z > z)[P_0(Y = 1) - P_n(Y = 1)]}{P_0(Y = 1)P_n(Y = 1)P_n(Z > z)} \\
& + \frac{P_0(Y = 1, Z > z)[P_0(Z > z) - P_n(Z > z)]}{P_0(Y = 1)P_0(Z > z)P_n(Z > z)}
\end{aligned}$$

Note that the first two terms after the third equality take the form  $\frac{A_n B_n}{C_n}$  as discussed.

Therefore the final equality follows. Next, consider the second, fourth, fifth, and sixth terms of  $R_{n,2}$ . Letting  $d_0 := \frac{P_0(Y=1, Z>z)}{P_0(Y=1)P_0(Z>z)}$ , we can factor and rewrite those terms as

$$d_0 \left[ \frac{P_n(Z > z) - P_0(Z > z)}{P_0(Z > z)} \right] + d_0 \left[ \frac{P_n(Y = 1) - P_0(Y = 1)}{P_0(Y = 1)} \right].$$

Putting all six terms of  $R_{n,2}$  back together, we can now write

$$\begin{aligned}
R_{n,2} &= d_0 \left[ \frac{P_0(Z > z) - P_n(Z > z)}{P_n(Z > z)} + \frac{P_n(Z > z) - P_0(Z > z)}{P_0(Z > z)} \right] \\
&\quad + \frac{P_0(Y = 1, Z > z) [P_0(Y = 1) - P_n(Y = 1)]}{P_0(Y = 1)P_n(Y = 1)P_n(Z > z)} \\
&\quad + d_0 \left[ \frac{P_n(Y = 1) - P_0(Y = 1)}{P_0(Y = 1)} \right] + o_p(n^{-1/2}) \\
&= d_0 \left[ \frac{\{P_0(Z > z) - P_n(Z > z)\} \{P_n(Z > z) - P_0(Z > z)\}}{P_n(Z > z)P_0(Z > z)} \right] \\
&\quad + \frac{P_0(Y = 1, Z > z) [P_0(Y = 1) - P_n(Y = 1)]}{P_0(Y = 1)P_n(Y = 1)P_n(Z > z)} \\
&\quad + d_0 \left[ \frac{P_n(Y = 1) - P_0(Y = 1)}{P_0(Y = 1)} \right] + o_p(n^{-1/2}) \\
&= \frac{P_0(Y = 1, Z > z) [P_0(Y = 1) - P_n(Y = 1)]}{P_0(Y = 1)P_n(Y = 1)P_n(Z > z)} \\
&\quad + d_0 \left[ \frac{P_n(Y = 1) - P_0(Y = 1)}{P_0(Y = 1)} \right] + o_p(n^{-1/2})
\end{aligned}$$

where the second equality results from finding a common denominator for the two fractions in the first term of the first equality. We then note that this term also takes the form  $\frac{A_n B_n}{C_n}$  and is therefore  $o_p(n^{-1/2})$ . The final equality follows since  $o_p(n^{-1/2}) + o_p(n^{-1/2}) = o_p(n^{-1/2})$ . The remaining two terms may be shown to be  $o_p(n^{-1/2})$  using similar techniques, beginning with adding and subtracting the term  $d_0 \frac{P_0(Y=1) - P_n(Y=1)}{P_n(Y=1)}$ . We omit the details in the interest of space. Thus, having shown that  $R_{n,2} = o_p(n^{-1/2})$ , applying  $P_0$  to A.2 yields

$$\begin{aligned}
& P_0(Q_{P_n} - Q_{P_0}) \\
&= \int_z P_0(Y = 1) \frac{1}{n} \sum_{i=1}^n \left\{ \frac{Y_i}{P_0(Z > z) P_0(Y = 1)} [I(Z_i > z) - P_0(Z > z | Y = 1)] \right\} dF_{0,Z|Y=1}(z) \\
&\quad - \int_z P_0(Y = 1) \frac{1}{n} \sum_{i=1}^n \left\{ \frac{P_0(Z > z | Y = 1)}{P_0(Z > z)^2} [I(Z_i > z) - P_0(Z > z)] \right\} dF_{0,Z|Y=1}(z) \\
&\quad + o_p(n^{-1/2}) \\
&= \frac{1}{n} \sum_{i=1}^n Y_i \int_z \left\{ \frac{I(Z_i > z) - P_0(Z > z | Y = 1)}{P_0(Z > z)} \right. \\
&\quad \quad \left. - \frac{P_0(Y = 1, Z > z)}{P_0(Z > z)^2} [I(Z_i > z) - P_0(Z > z)] \right\} dF_{0,Z|Y=1}(z) \\
&\quad + o_p(n^{-1/2}) \\
&= \frac{1}{n} \sum_{i=1}^n \left\{ Y_i \int_{z < Z_i} \frac{1}{P_0(Z > z)} dF_{0,Z|Y=1}(z) - \frac{Y_i}{P_0(Y = 1)} \Phi(P_0) \right. \\
&\quad \quad \left. - \int_{z < Z_i} \frac{P_0(Z > z, Y = 1)}{P_0(Z > z)^2} dF_{0,Z|Y=1}(z) + \Phi(P_0) \right\} + o_p(n^{-1/2})
\end{aligned}$$

Where the second and third equalities are the result of rearranging terms. Using this result, we may return to the difference between the NPMLE and the truth and write

$$\begin{aligned}
\Phi(P_n) - \Phi(P_0) &= P_0(Q_{P_n} - Q_{P_0}) + (P_n - P_0)Q_{P_0} + R_{n,1} \\
&= \frac{1}{n} \sum_{i=1}^n \left\{ Y_i \int_{z < Z_i} \frac{1}{P_0(Z > z)} dF_{0,Z|Y=1}(z) - \frac{Y_i}{P_0(Y = 1)} \Phi(P_0) \right. \\
&\quad \quad \left. - \int_{z < Z_i} \frac{P_0(Z > z, Y = 1)}{P_0(Z > z)^2} dF_{0,Z|Y=1}(z) + \Phi(P_0) \right\} \\
&\quad + \frac{1}{n} \sum_{i=1}^n \left\{ \frac{Y_i}{P_0(Y = 1)} \cdot \frac{P_0(Z > Z_i, Y = 1)}{P_0(Z > Z_i)} - \Phi(P_0) \right\} + o_p(n^{-1/2}) + R_{n,1}.
\end{aligned}$$

Bringing all but the remainder terms into the same sum and rearranging gives

$$\begin{aligned}
\Phi(P_n) - \Phi(P_0) &= \frac{1}{n} \sum_{i=1}^n \left\{ \int_{z < Z_i} \left[ Y_i - \frac{P_0(Z > z, Y = 1)}{P_0(Z > z)} \right] \left( \frac{1}{P_0(Z > z)} \right) dF_{0,Z|Y=1}(z) \right. \\
&\quad \left. + \frac{Y_i}{P_0(Y = 1)} \left[ \frac{P_0(Z > Z_i, Y = 1)}{P_0(Z > Z_i)} - \Phi_\psi(P_0) \right] \right\} + o_p(n^{-1/2}) + R_{n,1} \\
&= \frac{1}{n} \sum_{i=1}^n D_\Phi(P_0)(Z_i, Y_i) + o_p(n^{-1/2}) + R_{n,1}.
\end{aligned}$$

Where  $D_\Phi(P_0)(Z_i, Y_i)$  is exactly as defined in the statement of Theorem 1.3.1 using reduced notation. The only remaining task is to show that  $R_{n,1} = o_p(n^{-1/2})$ .

(ii) Recall that  $R_{n,1} = (P_n - P_0)(Q_{P_n} - Q_{P_0})$ . Using a key result from Empirical Process Theory [54], we can state that  $(P_n - P_0)(Q_{P_n} - Q_{P_0}) = o_p(n^{-1/2})$  if:

1.  $P_0(Q_{P_n} - Q_{P_0})^2 = o_p(1)$
2.  $Q_{P_n} - Q_{P_0}$  falls in a  $P_0$ -Donsker Class.

We introduce the following notation to prove that the two conditions above are met:

- $Q_p(z, y) := \frac{y}{P(Y=1)} \cdot \frac{P(Z > z, Y=1)}{P(Z > z)}$
- $PQ_p := \int Q_p(z, y) dP(y, z)$
- $h_0 := P_0(Y = 1)$
- $f_z(\tilde{o}) := I(\tilde{z} > z)$
- $f_{1,z}(\tilde{o}) := I(\tilde{y} = 1, \tilde{z} > z)$
- $g_n(z) := P_n(Y = 1 | Z > z) = \frac{P_n f_z}{P_n f_{1,z}}$
- $h_n := P_n(Y = 1)$
- Similarly for  $g_0(z)$  and  $h_0$

Beginning with condition 1, we can rewrite

$$\begin{aligned}
P_0(Q_{P_n} - Q_{P_0})^2 &= \int y \left( \frac{g_n(z)}{h_n} - \frac{g_0(z)}{h_0} \right)^2 dP_0(y, z) \\
&= \int y \left( \frac{g_n(z)}{h_n} \pm \frac{g_n(z)}{h_0} \pm \frac{g_0(z)}{h_n} \pm \frac{g_0(z)}{h_0} - \frac{g_0(z)}{h_0} \right)^2 \\
&= \frac{(h_n - h_0)^2}{h_n^2 h_0^2} \int y g_0^2(z) dP_0(y, z) \tag{A.3}
\end{aligned}$$

$$+ \frac{h_n^2 + (h_n - h_0)^2}{h_n^2 h_0^2} \int y (g_n(z) - g_0(z))^2 dP_0(y, z) \tag{A.4}$$

By the Continuous Mapping Theorem and Slutsky's Theorem, term A.3 and the portion of A.4 outside of the integral are both  $o_p(1)$ . Thus, to prove that condition 1 holds, we must only show that  $\int y (g_n(z) - g_0(z))^2 dP_0(y, z) = o_p(1)$ .

We can decompose that term and subsequently bound the resulting summands in the following way:

$$\begin{aligned}
&\int y (g_n(z) - g_0(z))^2 dP_0(y, z) \\
&= \int y \left[ \frac{(P_n - P_0)f_z}{P_0 f_{1,z}} \right]^2 dP_0(y, z) \\
&\quad + \int y P_0^2 f_z \left[ \frac{(P_n - P_0)f_{1,z}}{P_n f_{1,z} P_0 f_{1,z}} \right]^2 dP_0(y, z) \\
&\quad + \int y \left[ \frac{\{(P_n - P_0)f_z\} \{(P_n - P_0)f_{1,z}\}}{P_n f_{1,z} P_0 f_{1,z}} \right]^2 dP_0(y, z) \\
&< \int \frac{y}{P_0^2 f_{1,z}} \left( \text{Sup}_{z \in (0,1)} |(P_n - P_0)f_z| \right)^2 dP_0(y, z) \\
&\quad + \int \frac{y P_0^2 f_z}{P_n^2 f_{1,z} P_0^2 f_{1,z}} \left( \text{Sup}_{z \in (0,1)} |(P_n - P_0)f_{1,z}| \right)^2 dP_0(y, z) \\
&\quad + \int \frac{y}{P_n^2 f_{1,z} P_0^2 f_{1,z}} \left( \text{Sup}_{z \in (0,1)} |(P_n - P_0)f_z| \right)^2 \left( \text{Sup}_{z \in (0,1)} |(P_n - P_0)f_{1,z}| \right)^2 dP_0(y, z).
\end{aligned}$$

We can then pull each of the Sup terms out of the integrals to obtain



$$\left(\text{Sup}_{z \in (0,1)} |(P_n - P_0)f_z|\right)^2 \int \frac{y}{P_0^2 f_{1,z}} dP_0(y, z) \quad (\text{A.5})$$

$$+ \left(\text{Sup}_{z \in (0,1)} |(P_n - P_0)f_{1,z}|\right)^2 \int \frac{y P_0^2 f_z}{P_n^2 f_{1,z} P_0^2 f_{1,z}} dP_0(y, z) \quad (\text{A.6})$$

$$+ \left(\text{Sup}_{z \in (0,1)} |(P_n - P_0)f_z|\right)^2 \left(\text{Sup}_{z \in (0,1)} |(P_n - P_0)f_{1,z}|\right)^2 \int \frac{y}{P_n^2 f_{1,z} P_0^2 f_{1,z}} dP_0(y, z). \quad (\text{A.7})$$

$\text{Sup}_{z \in (0,1)} |(P_n - P_0)f_z|$  can be equivalently written as  $\text{Sup}_{f \in \mathcal{F}} |(P_n - P_0)f|$  where  $\mathcal{F} = \{f_z : 0 \leq z \leq 1\}$ . Note that  $\mathcal{F}$  is a Donsker class [18] and thus  $\text{Sup}_{z \in (0,1)} |(P_n - P_0)f_z| = \text{Sup}_{f \in \mathcal{F}} |(P_n - P_0)f| = O_p(n^{-1/2})$ . Similarly,  $\text{Sup}_{z \in (0,1)} |(P_n - P_0)f_{1,z}| = O_p(n^{-1/2})$ . Thus, the terms outside of the integrals in A.5, A.6, A.7 are  $O_p(n^{-1})$ ,  $O_p(n^{-1})$ , and  $O_p(n^{-2})$  respectively. Each of those terms is then certainly  $O_p(1)$  allowing us to write

$$\begin{aligned} & \int y(g_n(z) - g_0(z))^2 dP_0(y, z) \\ &= O_p(1) \int \frac{y}{P_0^2 f_{1,z}} dP_0(y, z) \\ &+ O_p(1) \int \frac{y P_0^2 f_z}{P_n^2 f_{1,z} P_0^2 f_{1,z}} dP_0(y, z) \\ &+ O_p(1) \int \frac{y}{P_n^2 f_{1,z} P_0^2 f_{1,z}} dP_0(y, z). \end{aligned} \quad (\text{A.8})$$

The stochastic order of the sum above is dictated by A.8. Plugging in the definition of the  $f_{1,z}$ , we can rewrite that term as

$$O_p(1) \int \frac{y}{P_n(Y=1, Z \geq z)^2 P_0(Y=1, Z \geq z)^2} P_0(dz | Y=1).$$

By the Continuous Mapping Theorem and the assumption given in Theorem 1.3.1,

$$\int \frac{y}{P_n(Y = 1, Z \geq)^2 P_0(Y = 1, Z \geq z)^2} P_0(dz | Y = 1) = O_p(1).$$

Thus, we can conclude that  $\int y(g_n(z) - g_0(z))^2 dP_0(y, z) = o_p(1)O_p(1)O_p(1) = o_p(1)$ , and that condition 1 is met.

We next turn to condition 2. We will show that  $Q_{P_n} - Q_{P_0}$  has bounded uniform sectional variation norm (USVN) with probability tending to 1 which is a sufficient condition to show that it falls in a  $P_0$ -Donsker Class. Throughout the remainder of the proof, we assume that  $P_0(Y = 1) > \delta$  for some  $\delta > 0$ . The USVN of  $Q_{P_n} - Q_{P_0}$  is

$$\|Q_{P_n} - Q_{P_0}\|_V^* = \max \left\{ \text{Sup}_{y,z} |Q_{P_n}(y, z) - Q_{P_0}(y, z)|, \right. \quad (\text{A.9})$$

$$\text{Sup}_y \int |Q_{P_n}(y, dz) - Q_{P_0}(y, dz)|, \quad (\text{A.10})$$

$$\text{Sup}_z \int |Q_{P_n}(dy, z) - Q_{P_0}(dy, z)|, \quad (\text{A.11})$$

$$\left. \int |Q_{P_n}(dy, dz) - Q_{P_0}(dy, dz)| \right\} \quad (\text{A.12})$$

Term A.9 can be written

$$\begin{aligned} & \text{Sup}_z |Q_{P_n}(1, z) - Q_{P_0}(1, z)| \\ &= \text{Sup}_z \left| \frac{g_n(z)}{h_n} - \frac{g_0(z)}{h_0} \right| \leq \frac{2}{\delta} + o_p(1). \end{aligned}$$

Term A.10 can be written

$$\begin{aligned}
& \int |Q_{P_n}(1, dz) - Q_{P_0}(1, dz)| \\
&= \int \left| \frac{g_n(dz)}{h_n} - \frac{g_0(dz)}{h_0} \right| \\
&= \int \frac{g_n(dz) - g_0(dz)}{h_0} + o_p(1) \\
&\leq \frac{\text{Sup}_z |g_n(z) - g_0(z)|}{\delta} \leq \frac{2}{\delta} + o_p(1).
\end{aligned}$$

Term A.11 can be written

$$\text{Sup}_z |Q_{P_n}(1, z) - Q_{P_0}(1, z)| \leq \frac{2}{\delta} + o_p(1).$$

Term A.12 can be written the in the same way as term A.10:

$$\int |Q_{P_n}(1, dz) - Q_{P_0}(1, dz)| \leq \frac{2}{\delta} + o_p(1).$$

Thus,  $P_0(\|Q_{P_n} - Q_{P_0}\|_V^* \leq \frac{2}{\delta}) \rightarrow 1$  as  $n \rightarrow \infty$  and so  $Q_{P_n} - Q_{P_0}$  has bounded USVN, with probability tending to 1. The class of functions with bounded USVN is a Donkser class [48]. Thus condition 2 is met and the proof is complete.

### A.3 Theorem 1.3.1 Condition Examination

Here we present evidence that, the condition in Theorem 1.3.1 that  $\int \frac{dP_0(Z \leq z | Y=1)}{P_0(Z \geq z | Y=1)^4} < \infty$  is likely too stringent and that there exists lighter conditions under which the parameter is asymptotically linear with the given influence function.

Consider the scenario in which  $P_0(Y = 1) = 0.5$  and  $Z \sim \text{Unif}(0, 1)$  independent of  $Y$ . In the context of a binary classification problem, this would indicate an uninformative machine learning model which generates a score at random. Here, the

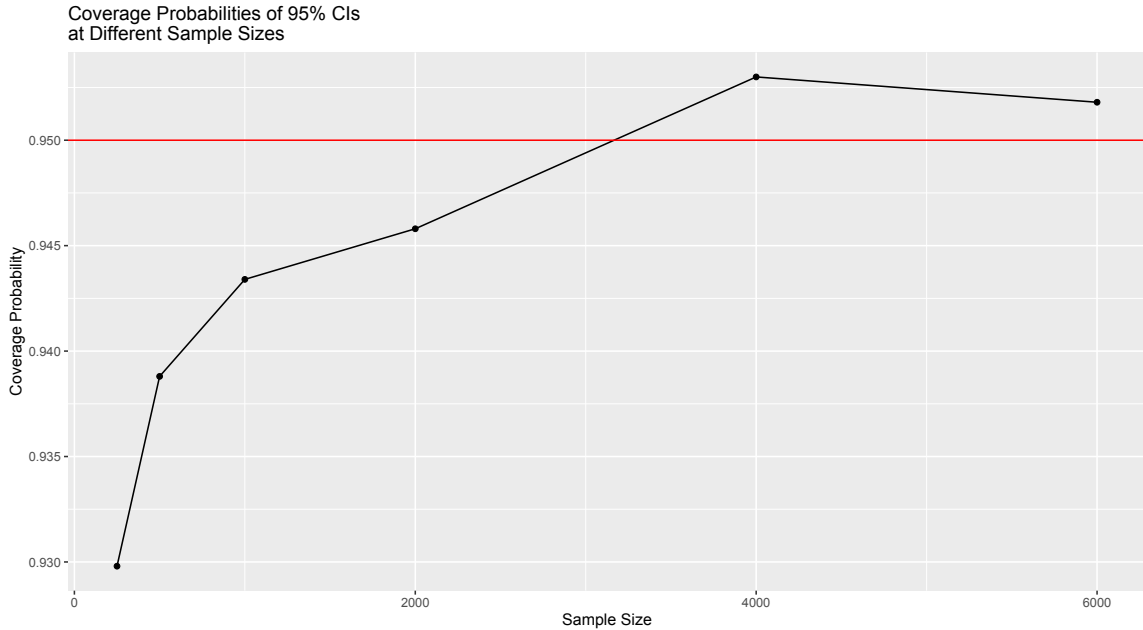


Figure A.1: Coverage probabilities of 95% confidence intervals for the AUPRC. Probabilities are calculated as the proportion of times the interval covered the true parameter out of 5000 iterations.

true AUPRC is 0.5. The term of interest is

$$\int \frac{16}{[P_0(Z \geq z | Y = 1)P(Y = 1)]^4} dz = \int \frac{16}{(1-z)^4} dz > \infty$$

Thus the condition is not satisfied. We conducted simulations under this data generating distribution at a variety of sample sizes. For each sample size, we generated data according to  $P_0(Y = 1) = 0.5$  and  $Z \sim \text{Unif}(0, 1)$  and calculated the NPMLE of the AUPRC and constructed confidence intervals using the variance of the influence function given in Theorem 1.3.1. This process was repeated 5000 times per sample size.

Coverage probabilities of 95% confidence intervals converge to 95% (A.1), and bias scaled by  $\sqrt{n}$  shrinks to 0 as sample size increases. Thus, we can still do valid estimation and inference even while violating the condition.

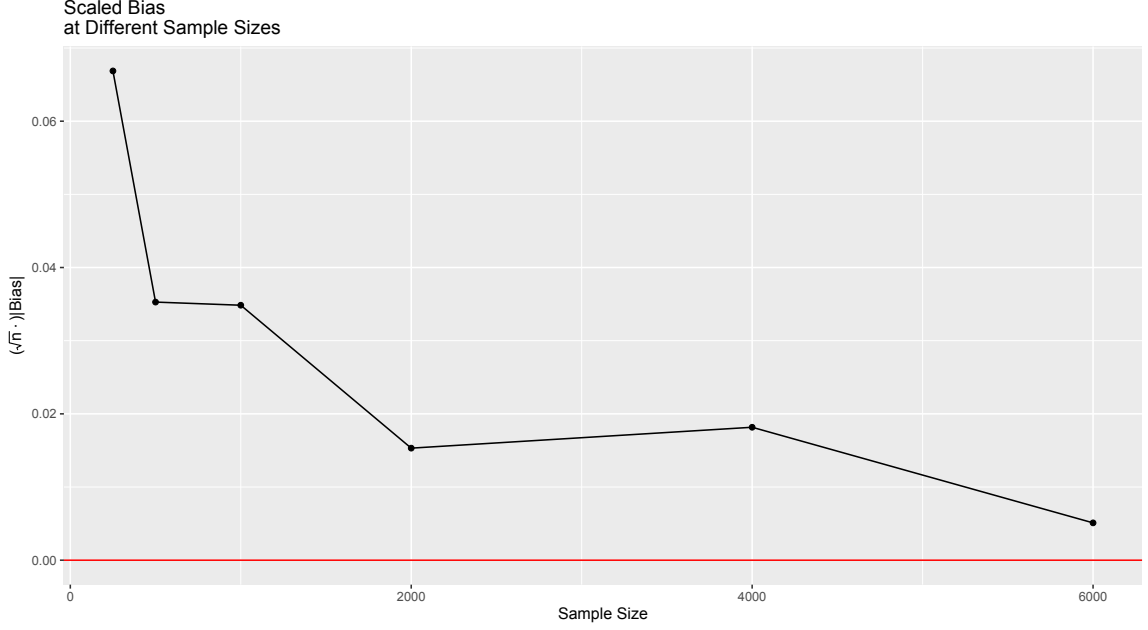


Figure A.2: Scaled absolute bias of the NPMLE of the AUPRC at several sample sizes. Results at a given sample size are averaged over 5000 iterations.

## A.4 Theorem 1.3.2 Proof

By Theorem 1.3.1, each AUPRC plug-in estimator  $\Phi_{\psi_{n,k}}(P_{n,k}^1)$  of  $\Phi_{\psi_{n,k}}(P_0)$  is asymptotically linear, and we can write

$$\begin{aligned} \phi_{n,cv} - \phi_{0,cv} &= \frac{1}{K} \sum_{k=1}^K [\Phi_{\psi_{n,k}}(P_{n,k}^1)] - \frac{1}{K} \sum_{k=1}^K [\Phi_{\psi_{n,k}}(P_0)] \\ &= \frac{1}{K} \sum_{k=1}^K (P_{n,k}^1 - P_0) D_{\Phi_{\psi_{n,k}}}(P_0) + o_p(n^{-1/2}). \end{aligned}$$

We can decompose the first term in the sum above into two pieces:

$$\begin{aligned} \frac{1}{K} \sum_{k=1}^K (P_{n,k}^1 - P_0) D_{\Phi_{\psi_{n,k}}}(P_0) &= \frac{1}{K} \sum_{k=1}^K (P_{n,k}^1 - P_0) D_{\Phi_{\psi_k}}(P_0) \\ &\quad + \frac{1}{K} \sum_{k=1}^K (P_{n,k}^1 - P_0) \{D_{\Phi_{\psi_{n,k}}}(P_0) - D_{\Phi_{\psi_k}}(P_0)\}. \end{aligned}$$

By the assumption that  $P_0\{D_{\Phi_{\psi_{n,k}}}(P_0) - D_{\Phi_{\psi_k}}(P_0)\}^2 \xrightarrow{p} 0$  and Theorem 2.14.1 in [53], we can write for the second piece that

$$\frac{1}{K} \sum_{k=1}^K (P_{n,k}^1 - P_0) \{D_{\Phi_{\psi_{n,k}}}(P_0) - D_{\Phi_{\psi_k}}(P_0)\} = o_p(n^{-1/2}).$$

Finally, we note that the first piece can be rewritten

$$\frac{1}{K} \sum_{k=1}^K (P_{n,k}^1 - P_0) D_{\Phi_{\psi_k}}(P_0) = (P_n - P_0) D_{\Phi_{\psi_k}}(P_0).$$

Thus, we have that

$$\begin{aligned} \phi_{n,cv} - \phi_{0,cv} &= (P_n - P_0) D_{\Phi_{\psi_k}}(P_0) + o_p(n^{-1/2}) \\ &= \frac{1}{n} \sum_{i=1}^n D_{\Phi_{\psi_k}}(P_0)(O_i) + o_p(n^{-1/2}). \end{aligned}$$

The CV estimator is therefore asymptotically linear and has the follows the asymptotic distribution stated in the theorem.

## A.5 Supplementary Figures

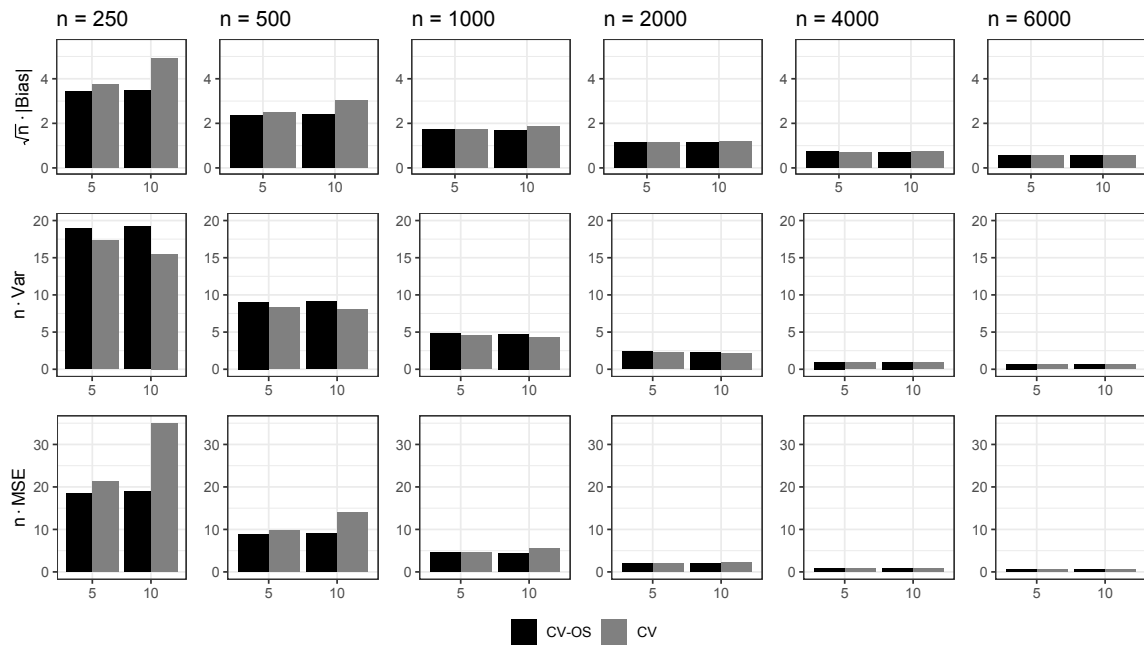


Figure A.3: Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Random Forest on data with **no imbalance**. 5 and 10 on the x-axis indicate number of outer folds  $K$ . In each case results are based on 1000 simulations.

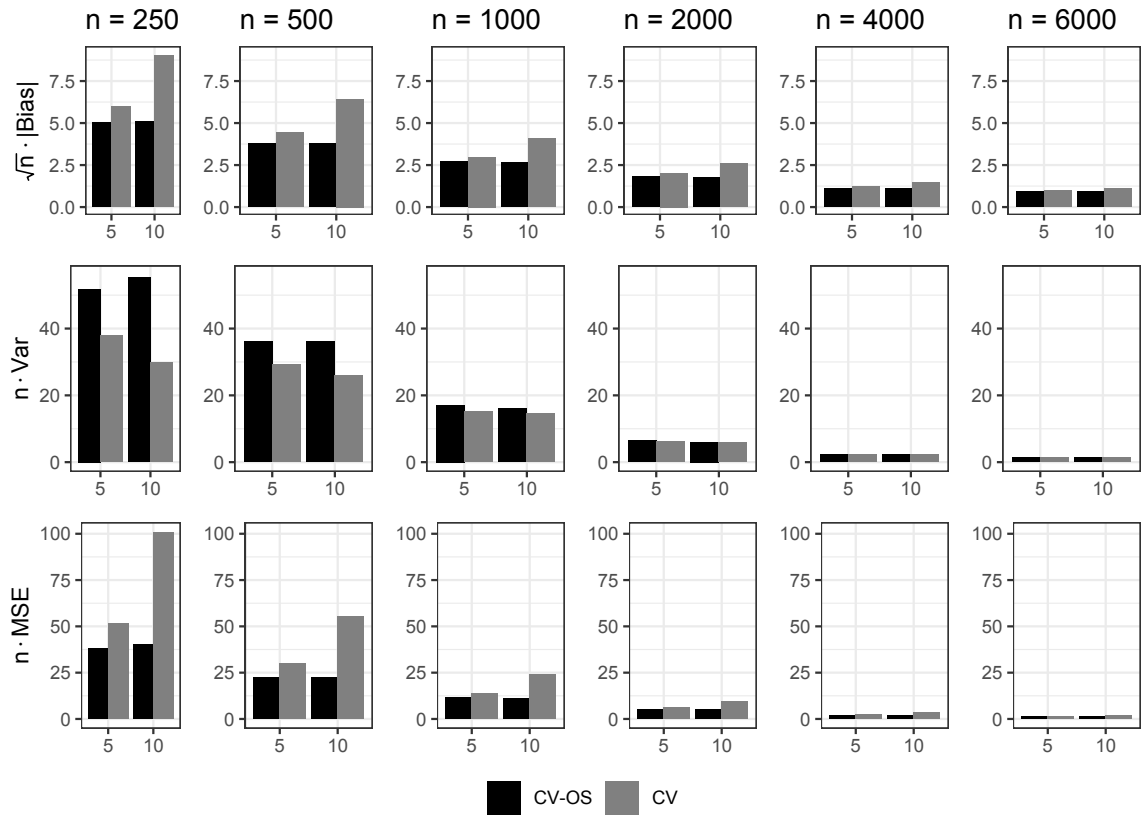


Figure A.4: Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Random Forest on data with **imbalance**. 5 and 10 on the x-axis indicate number of outer folds  $K$ . In each case results are based on 1000 simulations.



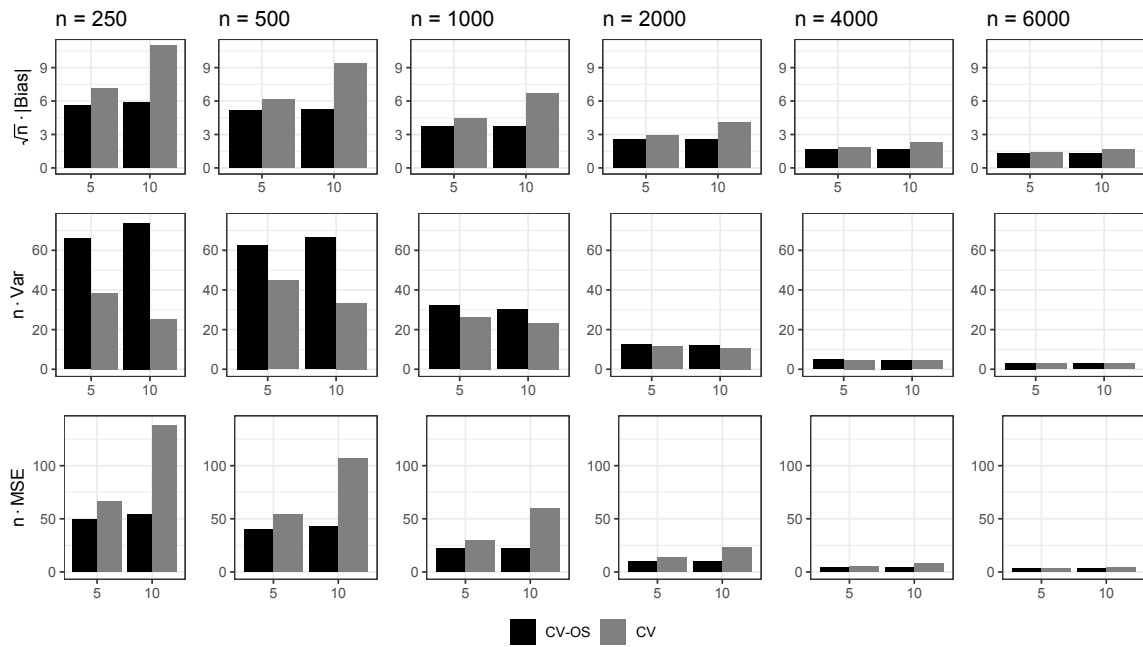


Figure A.5: Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Random Forest on data with **large imbalance**. 5 and 10 on the x-axis indicate number of outer folds  $K$ . In each case results are based on 1000 simulations.

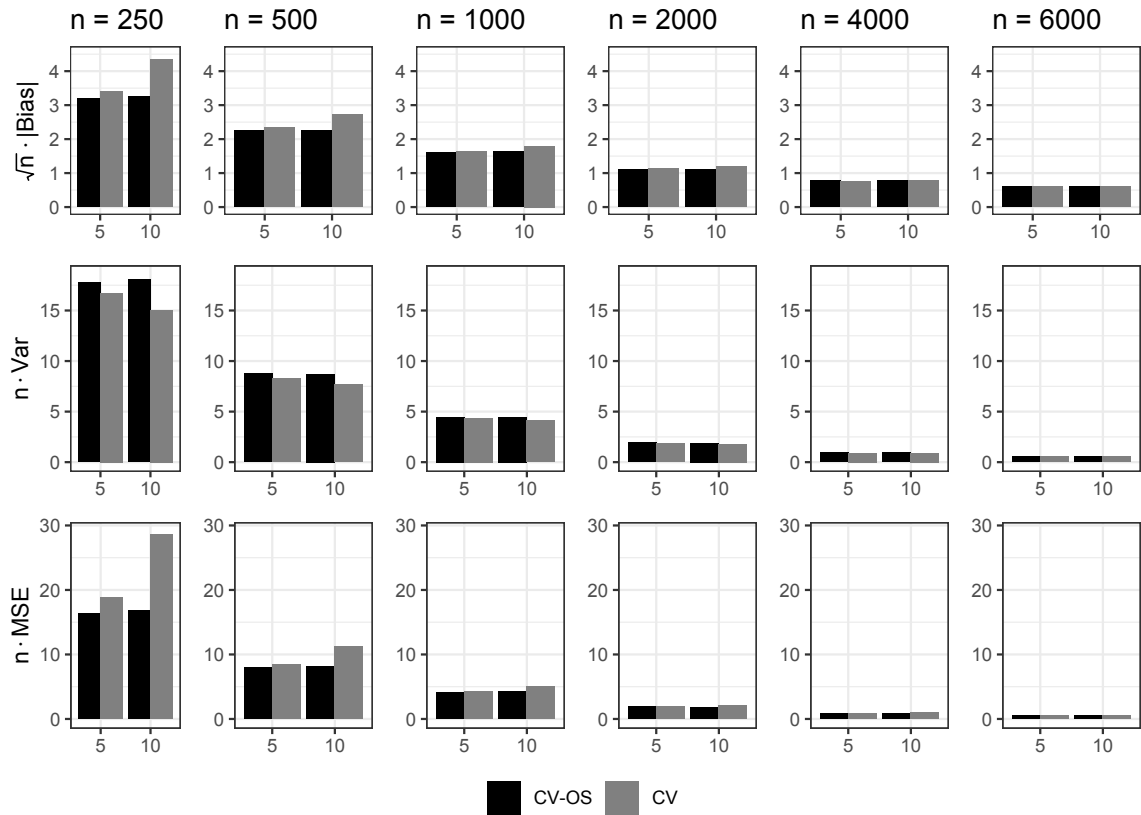


Figure A.6: Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Logistic Regression on data with **no imbalance**. 5 and 10 on the x-axis indicate number of outer folds  $K$ . In each case results are based on 1000 simulations.

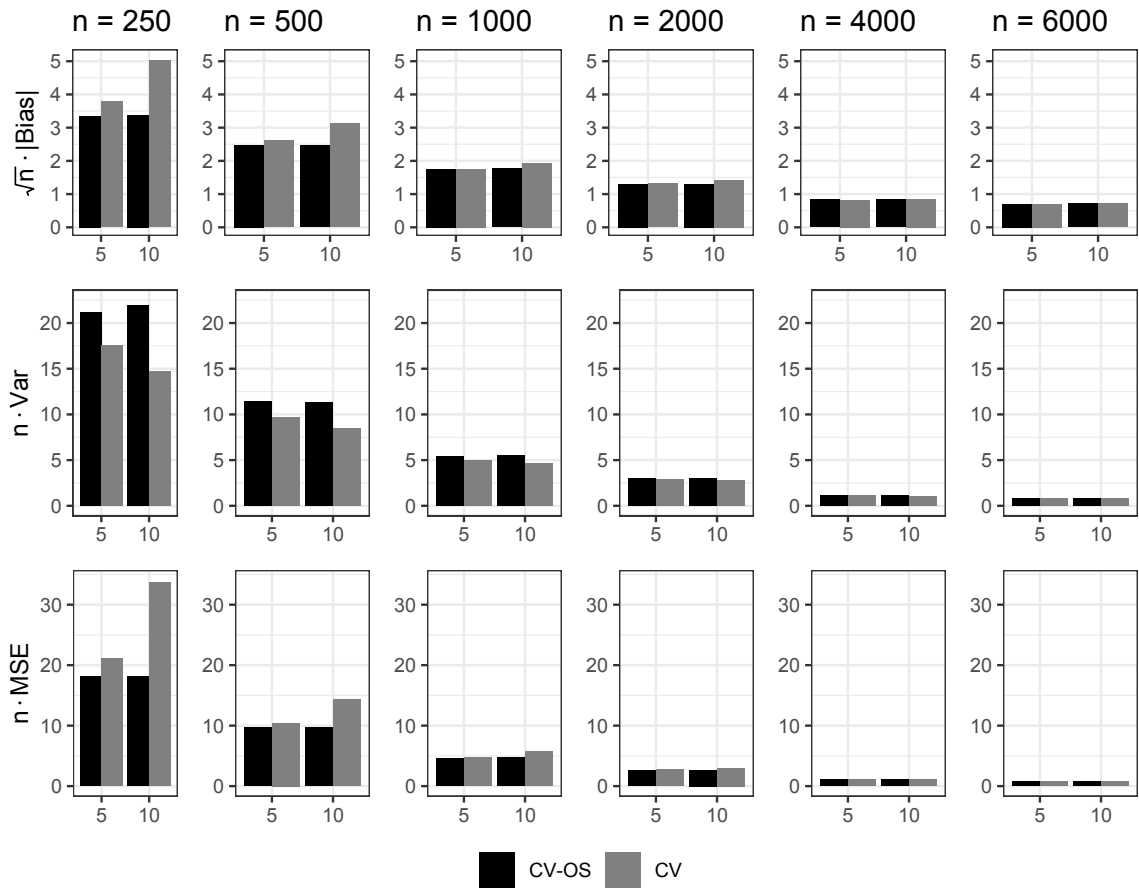


Figure A.7: Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Logistic Regression on data with **imbalance**. 5 and 10 on the x-axis indicate number of outer folds  $K$ . In each case results are based on 1000 simulations.

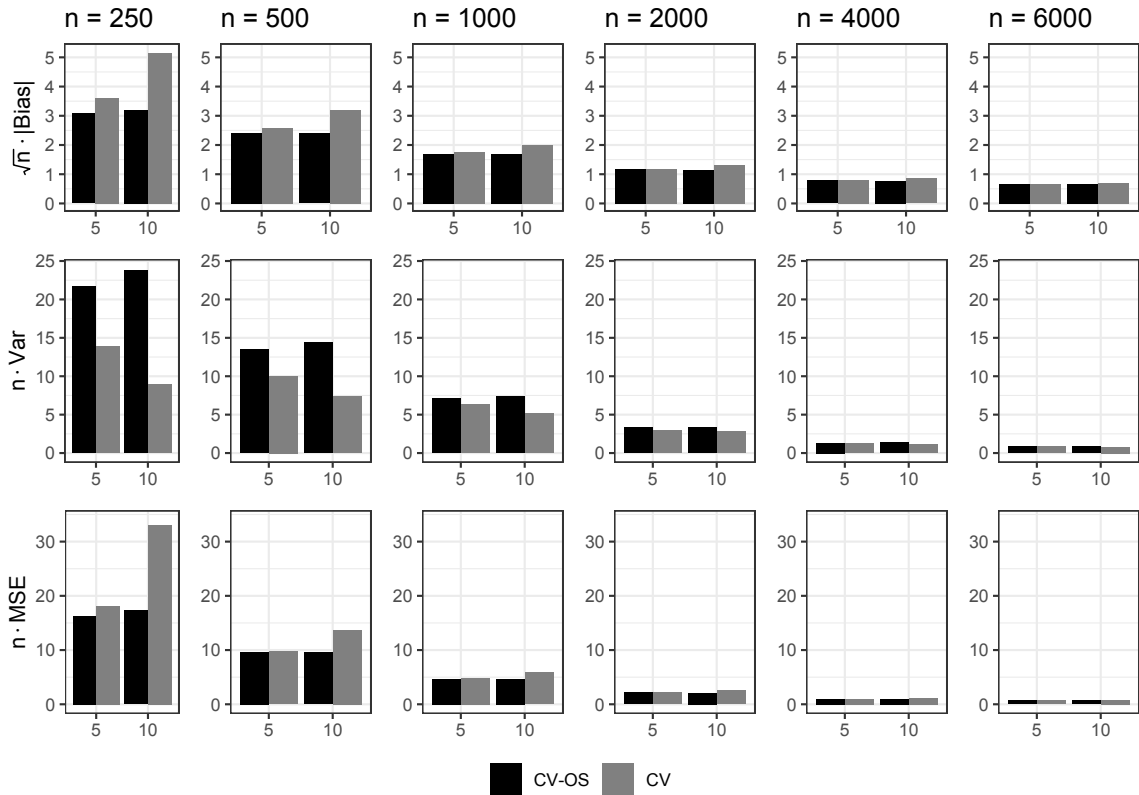


Figure A.8: Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Logistic Regression on data with **large imbalance**. 5 and 10 on the x-axis indicate number of outer folds  $K$ . In each case results are based on 1000 simulations.

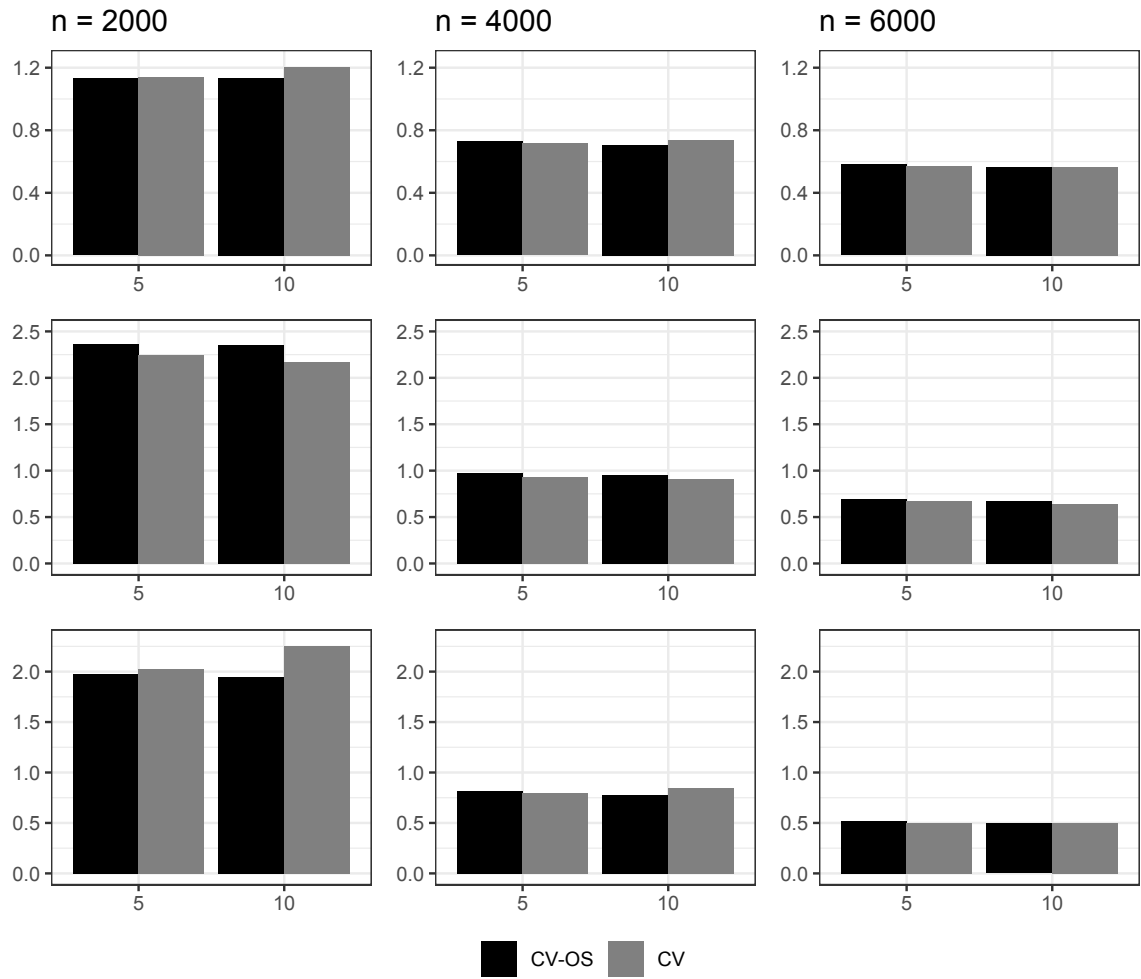


Figure A.9: Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Random Forest on data with **no imbalance**. 5 and 10 on the x-axis indicate number of outer folds  $K$ . In each case results are based on 1000 simulations. Restricted to  $n = 200, 4000, 6000$  for easier viewing.

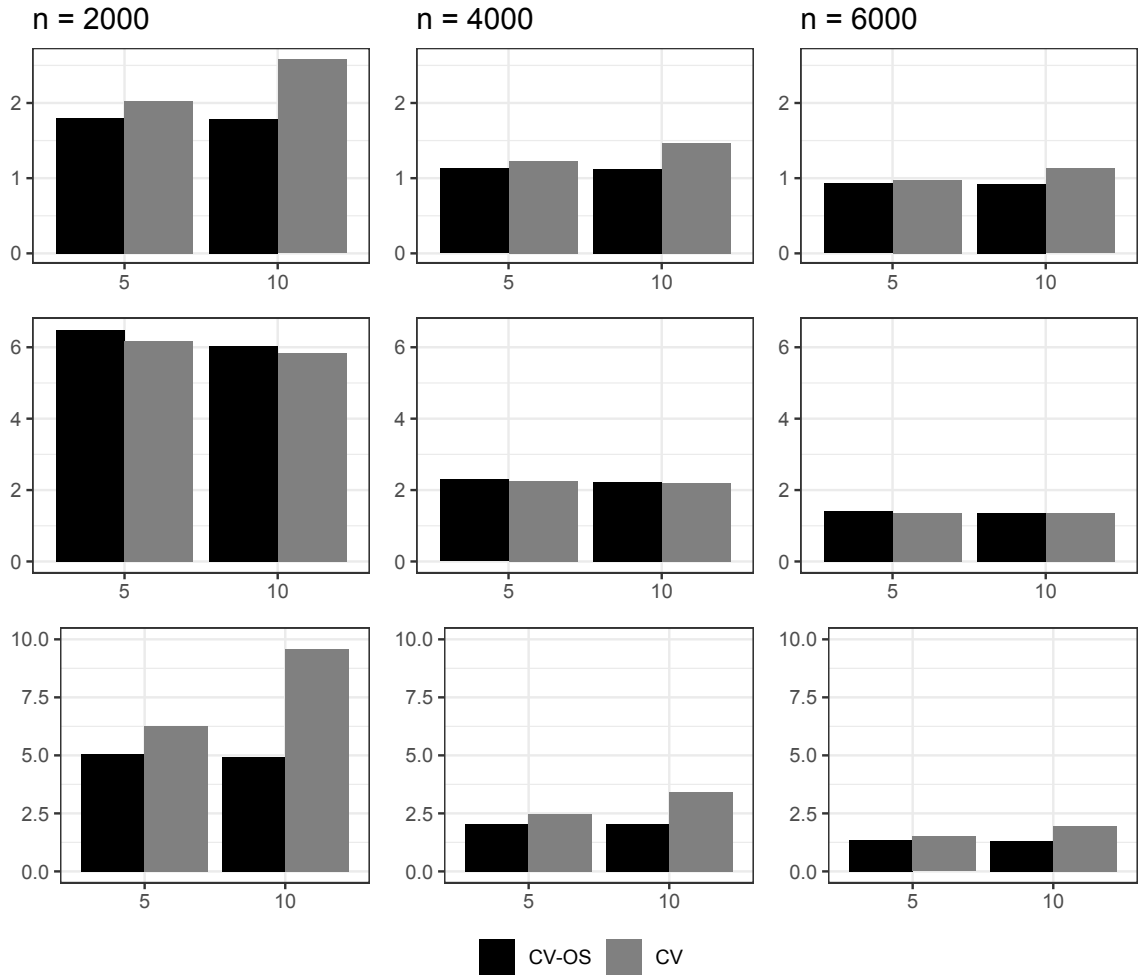


Figure A.10: Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Random Forest on data with **imbalance**. 5 and 10 on the x-axis indicate number of outer folds  $K$ . In each case results are based on 1000 simulations. Restricted to  $n = 200, 4000, 6000$  for easier viewing.

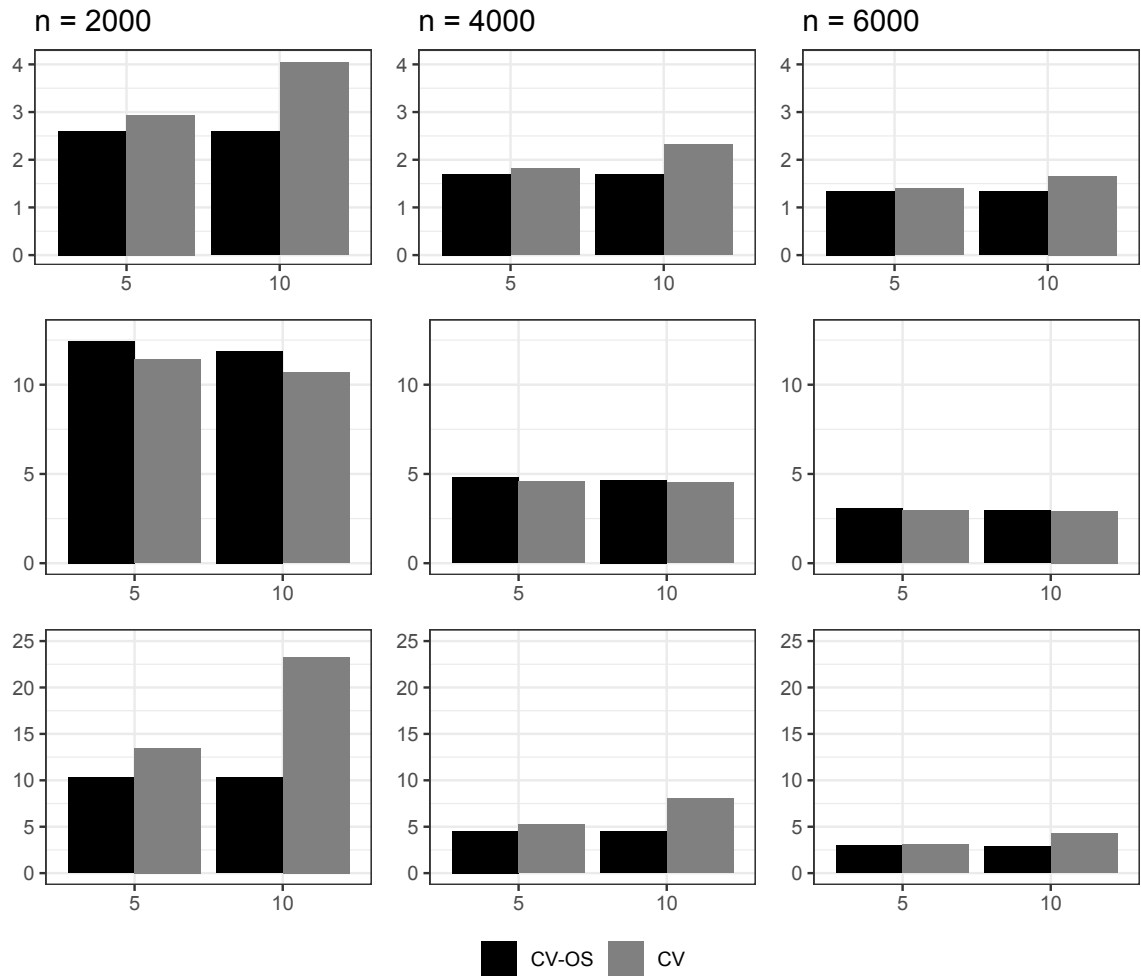


Figure A.11: Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Random Forest on data with **large imbalance**. 5 and 10 on the x-axis indicate number of outer folds  $K$ . In each case results are based on 1000 simulations. Restricted to  $n = 200, 4000, 6000$  for easier viewing.

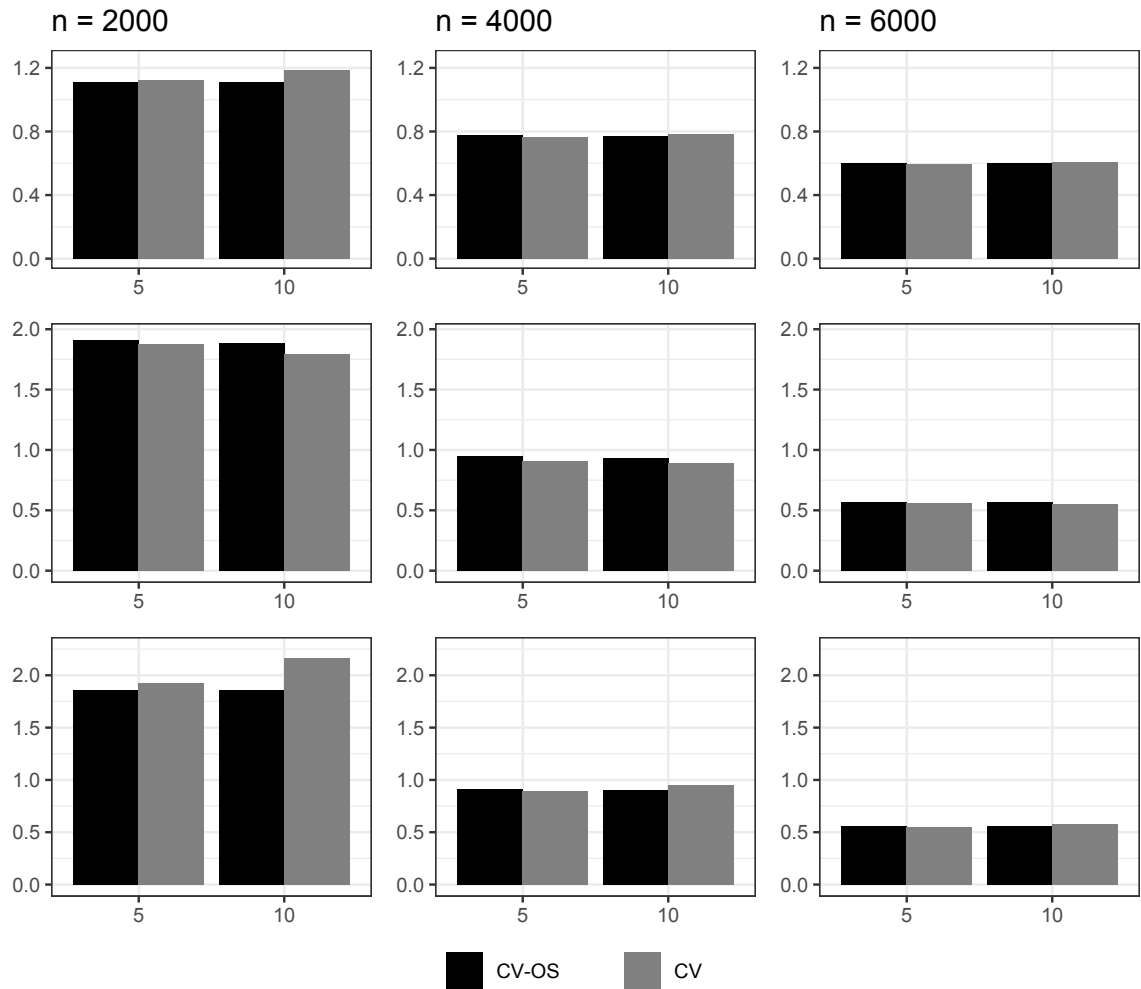


Figure A.12: Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Logistic Regression on data with **no imbalance**. 5 and 10 on the x-axis indicate number of outer folds  $K$ . In each case results are based on 1000 simulations. Restricted to  $n = 200, 4000, 6000$  for easier viewing.



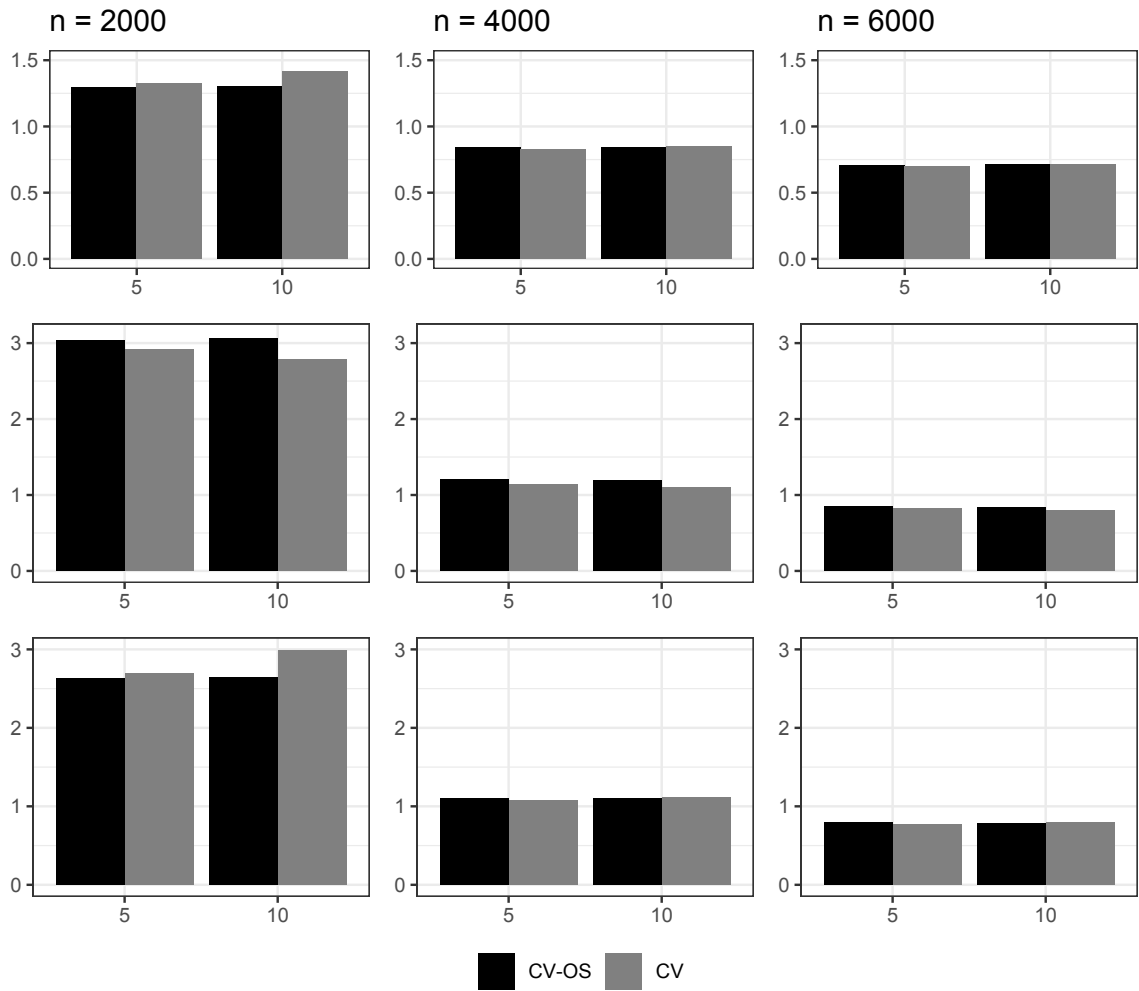


Figure A.13: Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Logistic Regression on data with **imbalance**. 5 and 10 on the x-axis indicate number of outer folds  $K$ . In each case results are based on 1000 simulations. Restricted to  $n = 200, 4000, 6000$  for easier viewing.

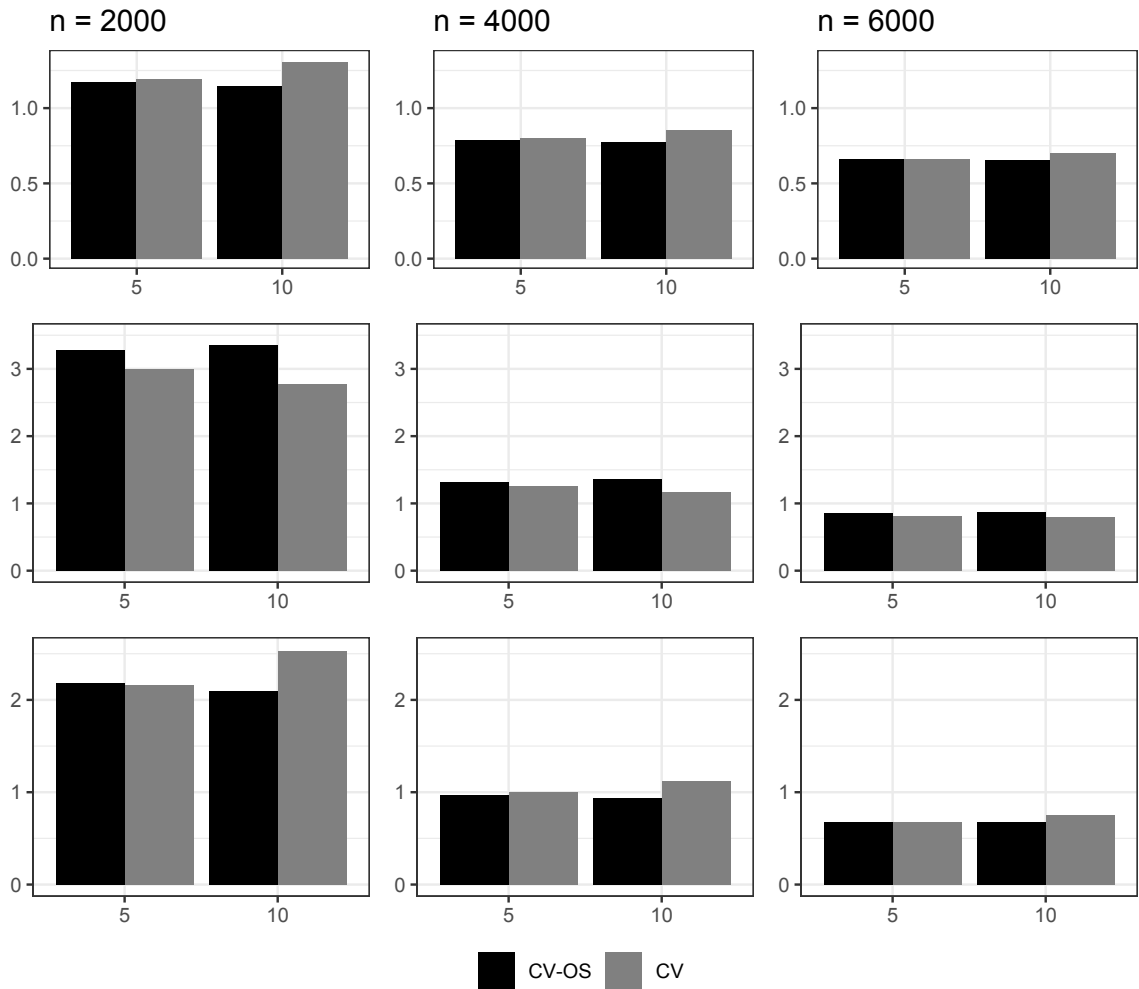


Figure A.14: Scaled absolute bias, variance, and mean squared error (MSE) for CV and CV-OS estimators of the performance of Logistic Regression on data with **large imbalance**. 5 and 10 on the x-axis indicate number of outer folds  $K$ . In each case results are based on 1000 simulations. Restricted to  $n = 200, 4000, 6000$  for easier viewing.

# Appendix B

## Chapter 3 Supplementary Material

### B.1 Theorem 3.2.1 Proof

Throughout the appendix we adopt the notation that for a  $P$ -measurable function  $f$ ,  $Pf := \int f dP$ .

Consider the pseudo-outcome for learning  $\mu_0(a, W) - \mu_0(0, W)$  defined for an observation  $O_i$  as

$$D_{\mu, \pi}^a(O_i) = \left[ \frac{\mathbb{1}_a(A_i)}{\pi(a | W_i)} - \frac{\mathbb{1}_0(A_i)}{\pi(0 | W_i)} \right] \{Y_i - \mu(A_i, W_i)\} + \mu(a, W_i) - \mu(0, W_i)$$

Let  $\mathcal{A}$  denote the set of possible treatments. Consider that

$$\begin{aligned}
E_0[D_{\mu,\pi}^a(O) | W] &= E_0 \left( \left[ \frac{\mathbb{1}_a(A)}{\pi(a | W)} - \frac{\mathbb{1}_0(A)}{\pi(0 | W)} \right] \{Y - \mu(A, W)\} + \mu(a, W) - \mu(0, W) | W \right) \\
&= E_0 \left( \left[ \frac{\mathbb{1}_a(A)}{\pi(a | W)} - \frac{\mathbb{1}_0(A)}{\pi(0 | W)} \right] \{Y - \mu(A, W)\} | W \right) + \mu(a, W) - \mu(0, W) \\
&= E_0 \left\{ E_0 \left( \left[ \frac{\mathbb{1}_a(A)}{\pi(a | W)} - \frac{\mathbb{1}_0(A)}{\pi(0 | W)} \right] \{Y - \mu(A, W)\} | A, W \right) | W \right\} + \mu(a, W) - \mu(0, W) \\
&= E_0 \left( \left[ \frac{\mathbb{1}_a(A)}{\pi(a | W)} - \frac{\mathbb{1}_0(A)}{\pi(0 | W)} \right] \{\mu_0(A, W) - \mu(A, W)\} | W \right) + \mu(a, W) - \mu(0, W) \\
&= \sum_{u \in \mathcal{A}} \left( \left[ \frac{I(u = a)}{\pi(a | W)} - \frac{I(u = 0)}{\pi(0 | W)} \right] \{\mu_0(u, W) - \mu(u, W)\} \right) \pi_0(u | W) + \mu(a, W) - \mu(0, W) \\
&= \frac{\pi_0(a | W)}{\pi(a | W)} \{\mu_0(a, W) - \mu(a, W)\} - \frac{\pi_0(0 | W)}{\pi(0 | W)} \{\mu_0(0, W) - \mu(0, W)\} + \mu(a, W) - \mu(0, W) .
\end{aligned}$$

Thus,  $E_0[D_{\mu,\pi}^a(O) | W] = \mu_0(a, W) - \mu_0(0, W)$  if either  $\mu = \mu_0$  or  $\pi = \pi_0$ .  $\square$

## B.2 Theorem 3.4.1 Proof

Consider the oracle regret  $P_0 \left[ L_{S,\mu_0}^a(\hat{\psi}_S) - L_{S,\mu_0}^a(\psi_0) \right]$  for learning the effect of treatment  $a$ .

(i) When  $V = W$ ,  $\hat{\psi}_S(a | v) = \hat{\mu}(a | v) - \hat{\mu}(0 | v)$ . Thus,

$$\begin{aligned}
P_0 \left[ L_{S,\mu_0}^a(\hat{\psi}_S) - L_{S,\mu_0}^a(\psi_0) \right] &= P_0 \left[ (\hat{\mu}_a - \mu_{a,0} + \mu_{0,0} - \hat{\mu}_0)^2 \right] \\
&= \|\hat{\mu}_a - \mu_{a,0} + \mu_{0,0} - \hat{\mu}_0\|_{2,P_0}^2 \\
&\leq (\|\hat{\mu}_a - \mu_{a,0}\|_{2,P_0} + \|\mu_{0,0} - \hat{\mu}_0\|_{2,P_0})^2 \\
&= \hat{R}_p^{*2}
\end{aligned}$$

where the third line follows from the Triangle Inequality, and the fourth line follows from two applications of the regret rate established for the Highly Adaptive Lasso in Nizam and Benkeser [37].

We can also consider simultaneously learning  $\psi_0(a | v)$  for multiple  $a$  by considering

a summed loss function

$$L_{S,\mu}(\psi) = \sum_{a \in \mathcal{A}_0} \omega(a) L_{S,\mu}^a(\psi) ,$$

where  $\omega(a)$  is any positive weight. Then the oracle regret is as follows:

$$P_0 \left[ L_{S,\mu_0}(\hat{\psi}_S) - L_{S,\mu_0}(\psi_0) \right] = \sum_{a \in \mathcal{A}_0} \omega(a) P_0 \left[ L_{S,\mu_0}^a(\hat{\psi}_S) - L_{S,\mu_0}^a(\psi_0) \right]$$

where  $P_0 \left[ L_{S,\mu_0}^a(\hat{\psi}_S) - L_{S,\mu_0}^a(\psi_0) \right]$  achieves the rate established above.  $\square$

(ii) Again, begin by considering the effect of a single treatment  $a$ . When  $V \subset W$ , recall that we first estimate the outcome regression, then regress pseudo-outcome  $D_{\hat{\mu}}^a(O) = \hat{\mu}(a, W) - \hat{\mu}(0, W)$  on covariates  $V$ . It is convenient to introduce the following shorthand:  $\hat{\psi}(v) := \hat{\psi}_S(a | v)$ ,  $\psi_0(v) := \psi_0(a | v)$ ,  $\mu_{0,a} := \mu_0(a, \cdot)$ ,  $\hat{\mu}_a := \hat{\mu}(a, \cdot)$ ,  $L_{0,a} := L_{S,\mu_0,a}$ ,  $\hat{L}_a := \hat{L}_{a,\hat{\mu}}$ ,  $\hat{D}_a := D_{\hat{\mu}}^a$ ,  $D_0 := D_{\mu_0}^a$ .

The oracle regret can be written as

$$\begin{aligned} P_0[L_{0,a}(\hat{\psi}) - L_{0,a}(\psi_0)] &= P_0[L_{0,a}(\hat{\psi}) - L_{0,a}(\psi_0)] + P_0 \left[ L_{0,a}(\hat{\psi}) - L_{0,a}(\psi_0) - \left( \hat{L}_a(\hat{\psi}) - \hat{L}_a(\psi_0) \right) \right] \\ &= P_0 \left[ \hat{L}_a(\hat{\psi}) - \hat{L}_a(\psi_0) \right] \end{aligned} \tag{B.1}$$

$$+ P_0 \left[ \hat{L}_a(\psi_0) - L_{0,a}(\psi_0) \right] \tag{B.2}$$

$$- P_0 \left[ \hat{L}_a(\hat{\psi}) - L_{0,a}(\hat{\psi}) \right] . \tag{B.3}$$

We expect due to the theory of the HAL estimator [8], that (B.1) achieves rate  $\hat{R}_q^*$ .

We can rewrite (B.2) as:

$$\begin{aligned}
P_0 \left[ \hat{L}_a(\psi_0) - L_{0,a}(\psi_0) \right] &= P_0 \left[ (\psi_0 - \hat{D}_a)^2 - (\psi_0 - D_{0,a}) \right] \\
&= P_0 \left[ 2\psi_0(D_{0,a} - \hat{D}_a) + \hat{D}_a^2 - D_{0,a}^2 \right].
\end{aligned}$$

(B.3) can be rewritten similarly:

$$\begin{aligned}
P_0 \left[ \hat{L}_a(\hat{\psi}) - L_{0,a}(\hat{\psi}) \right] &= P_0 \left[ (\hat{\psi} - \hat{D}_a)^2 - (\hat{\psi} - D_{0,a}) \right] \\
&= P_0 \left[ 2\hat{\psi}(D_{0,a} - \hat{D}_a) + \hat{D}_a^2 - D_{0,a}^2 \right].
\end{aligned}$$

Putting all three results together, we can write our oracle regret as

$$\begin{aligned}
&P_0[L_{0,a}(\hat{\psi}) - L_{0,a}(\psi_0)] \\
&= 2P_0 \left[ (\psi_0 - \hat{\psi})(D_{0,a} - \hat{D}_a) \right] + \hat{R}_q^* \\
&= 2P_0 \left[ \{\mu_{0,a} - \hat{\mu}_{0,a} + \hat{\mu}_0 - \mu_{0,0}\} \{\psi_0 - \hat{\psi}\} \right] + \hat{R}_q^* \\
&= 2P_0 \left[ \{\mu_{0,a} - \hat{\mu}_{0,a}\} \{\psi_0 - \hat{\psi}\} \right] + 2P_0 \left[ \{\hat{\mu}_0 - \mu_{0,0}\} \{\psi_0 - \hat{\psi}\} \right] + \hat{R}_q^* \\
&\leq 2P_0 \left[ \psi_0 - \hat{\psi} \right]^{1/2} \left\{ P_0 \left[ (\mu_{0,a} - \hat{\mu}_a)^2 \right]^{1/2} + P_0 \left[ (\hat{\mu}_0 - \mu_{0,0})^2 \right]^{1/2} \right\} + \hat{R}_q^* \\
&= 2\hat{R}_q^* o_p \left( \|\mu_{0,a} - \hat{\mu}_a\|_{2,P_0} + \|\hat{\mu}_0 - \mu_{0,0}\|_{2,P_0} \right) + \hat{R}_q^*
\end{aligned}$$

where the third line results from plugging in the definition of the pseudo-outcome, the fifth line is a result of the Cauchy-Schwartz Inequality, and the sixth line invokes the regret rate established for the Highly Adaptive Lasso in Nizam and Benkeser [37].

Again, we can then consider simultaneously learning  $\psi_0(a | V)$  for multiple  $a$  by

considering a summed loss function

$$L(\psi) = \sum_{a \in \mathcal{A}_0} \omega(a) L^a(\psi) ,$$

where  $\omega(a)$  is any positive weight. Then the oracle regret is as follows:

$$P_0 \left[ L_0(\hat{\psi}) - L_0(\psi_0) \right] = \sum_{a \in \mathcal{A} \setminus \{0\}} \omega(a) P_0 \left[ L_{0,a}(\hat{\psi}) - L_{0,a}(\psi_0) \right]$$

where  $P_0 \left[ L_{0,a}(\hat{\psi}) - L_{0,a}(\psi_0) \right]$  achieves the rate established above.  $\square$

### B.3 Theorem 3.4.2 Proof

Again, we begin by simplifying notation with the following shorthand:  $\hat{\psi} := \hat{\psi}_{\text{DR}}(a \mid v)$ ,  $\psi_0 := \psi_0(a \mid v)$ ,  $\mu_{0,a} := \mu_0(a, \cdot)$ ,  $\hat{\mu}_a := \hat{\mu}(a, \cdot)$ ,  $L_{0,a} := L_{\text{DR}, \mu_0, \pi_0, a}$ ,  $\hat{L}_a := \hat{L}_{a, \hat{\mu}, \hat{\pi}}$ ,  $\hat{D}_a := D_{\hat{\mu}, \hat{\pi}}^a$ ,  $D_0 := D_{\mu_0, \pi_0}^a$ .

We start by considering the regret for learning the effect of a single treatment  $a$ . We use the work from the proof of Theorem 3.4.1, shown in B.2 and immediately note that

$$P_0[L_{0,a}(\hat{\psi}) - L_{0,a}(\psi_0)] = 2P_0 \left[ \left( D_{0,a} - \hat{D}_a \right) \left( \psi_0 - \hat{\psi} \right) \right] + \hat{R}_q^*.$$

Applying the Law of Total Expectation, we can rewrite the expectation in that

expression as

$$\begin{aligned}
& P_0 \left[ \left( D_0(O) - \hat{D}(O) \right) \left( \psi_0 - \hat{\psi} \right) \right] \\
&= E_0 \left[ E_0 \left[ \left( D_0(O) - \hat{D}(O) \right) \left( \psi_0(a | V) - \hat{\psi}(a | V) \right) \mid A, W \right] \right] \\
&= E_0 \left[ E_0 \left[ D_0(O) - \hat{D}(O) \mid A, W \right] \left( \psi_0(a | V) - \hat{\psi}(a | V) \right) \right] \\
&= E_0 \left[ E_0 \left[ E_0 \left[ D_0(O) - \hat{D}(O) \mid A, W \right] \left( \psi_0(a | V) - \hat{\psi}(a | V) \right) \mid W \right] \right] \\
&= E_0 \left[ E_0 \left[ E_0 \left[ D_0(O) - \hat{D}(O) \mid A, W \right] \mid W \right] \left( \psi_0(a | V) - \hat{\psi}(a | V) \right) \right]
\end{aligned}$$

Now consider the inner-most expectation in the term above.

$$\begin{aligned}
& E_0 \left[ D_0(O) - \hat{D}(O) \mid A, W \right] \\
&= \left[ \frac{\mathbb{1}_a(A)}{\pi_0(a | W)} - \frac{\mathbb{1}_0(A)}{\pi_0(0 | W)} \right] \{ \mu_0(A, W) - \mu_0(a, W) \} + \mu_0(a, W) - \mu_0(0, W) \\
&\quad - \left[ \frac{I(A_i = a)}{\hat{\pi}(a | W)} - \frac{\mathbb{1}_0(A)}{\hat{\pi}(0 | W)} \right] \{ \mu_0(A, W) - \hat{\mu}(A, W) \} + \hat{\mu}(a, W) - \hat{\mu}(0, W) \\
&= \left[ \frac{\mathbb{1}_a(A)}{\hat{\pi}(a | W)} - \frac{\mathbb{1}_0(A)}{\hat{\pi}(0 | W)} \right] \{ \hat{\mu}(A, W) - \mu_0(A, W) \} + \{ \mu_0(a, W) - \hat{\mu}(a, W) + \hat{\mu}(0, W) - \mu_0(0, W) \}
\end{aligned} \tag{B.4}$$



Next, we write

$$\begin{aligned}
& E_0 \left[ E_0 \left[ D_0(O) - \hat{D}(O) \mid A, W \right] \mid W \right] \\
&= E_0 \left[ \left\{ \frac{\mathbb{1}_a(A)}{\hat{\pi}(a \mid W)} - \frac{\mathbb{1}_0(A)}{\hat{\pi}(0 \mid W)} \right\} \{ \hat{\mu}(A, W) - \mu_0(A, W) \} \mid W \right] \\
&\quad + \{ \mu_0(a, W) - \hat{\mu}(a, W) + \hat{\mu}(0, W) - \mu_0(0, W) \} \\
&= \frac{\pi_0(a \mid W)}{\hat{\pi}(a \mid W)} \{ \hat{\mu}(a, W) - \mu_0(a, W) \} - \frac{\pi_0(0 \mid W)}{\hat{\pi}(0 \mid W)} \{ \hat{\mu}(0, W) - \mu_0(0, W) \} \\
&\quad + \{ \mu_0(a, W) - \hat{\mu}(a, W) \} + \{ \hat{\mu}(0, W) - \mu_0(0, W) \} \\
&= \frac{\pi_0(a \mid W) \{ \hat{\mu}(a, W) - \mu_0(a, W) \} - \hat{\pi}(a \mid W) \{ \hat{\mu}(a, W) - \mu_0(a, W) \}}{\hat{\pi}(a \mid W)} \\
&\quad + \frac{\hat{\pi}(0 \mid W) \{ \hat{\mu}(0, W) - \mu_0(0, W) \} - \pi_0(0 \mid W) \{ \hat{\mu}(0, W) - \mu_0(0, W) \}}{\hat{\pi}(0 \mid W)} \\
&= \frac{\{ \pi_0(a \mid W) - \hat{\pi}(a \mid W) \} \{ \hat{\mu}(a, W) - \mu_0(a, W) \}}{\hat{\pi}(a \mid W)} \\
&\quad - \frac{\{ \pi_0(0 \mid W) - \hat{\pi}(0 \mid W) \} \{ \hat{\mu}(0, W) - \mu_0(0, W) \}}{\hat{\pi}(0 \mid W)}.
\end{aligned}$$

Thus, returning to the original regret rate, we can say that

$$\begin{aligned}
& P_0[L_0(\hat{\psi}) - L_0(\psi_0)] \\
&= 2E_0 \left[ (\psi_0(a \mid V) - \hat{\psi}(a \mid V))(D_0(O) - \hat{D}(O)) \right] + \hat{R}_q^* \\
&= 2E_0 \left[ \left( \frac{\{ \pi_0(a \mid W) - \hat{\pi}(a \mid W) \} \{ \hat{\mu}(a, W) - \mu_0(a, W) \}}{\hat{\pi}(a \mid W)} \right) (\psi_0(a \mid V) - \hat{\psi}(a \mid V)) \right] \\
&\quad - 2E_0 \left[ \left( \frac{\{ \pi_0(0 \mid W) - \hat{\pi}(0 \mid W) \} \{ \hat{\mu}(0, W) - \mu_0(0, W) \}}{\hat{\pi}(0 \mid W)} \right) (\psi_0(a \mid V) - \hat{\psi}(a \mid V)) \right].
\end{aligned}$$

We can then apply the Cauchy-Schwartz inequality two times:

$$\begin{aligned}
& E_0 \left[ \left( \frac{\{\pi_0(a | W) - \hat{\pi}(a | W)\} \{\hat{\mu}(a, W) - \mu_0(a, W)\}}{\hat{\pi}(a | W)} \right) (\psi_0(a | V) - \hat{\psi}(a | V)) \right] \\
& \leq E_0 \left[ \frac{\{\pi_0(a | W) - \hat{\pi}(a | W)\}^4}{\hat{\pi}(a | W)^4} \right]^{1/4} E_0 [\{\hat{\mu}(a, W) - \mu_0(a, W)\}^4]^{1/4} \\
& \quad \cdot E_0 \left[ \{\psi_0(a | V) - \hat{\psi}(a | V)\}^2 \right]^{1/2}
\end{aligned}$$

We can apply the regret rate established for HAL in Nizam and Benkeser [37] to the third term in the product above. Assuming that  $\hat{\pi}(a | w)$  is bounded away from 0 and 1, we can establish the following rate:

$$\begin{aligned}
& P_0 [L_{\hat{\mu}, \hat{\pi}}(\psi_n) - L_{\hat{\mu}, \hat{\pi}}(\psi_0)] \\
& = o_p(\|\pi_{a,0} - \hat{\pi}_a\|_{4, P_0} \cdot \|\hat{\mu}_a - \mu_{a,0}\|_{4, P_0}) \hat{R}_q^* \\
& \quad + o_p(\|\pi_{0,0} - \hat{\pi}_0\|_{4, P_0} \cdot \|\hat{\mu}_0 - \mu_{0,0}\|_{4, P_0}) \hat{R}_q^* \\
& \quad + \hat{R}_q^*.
\end{aligned}$$

We can also consider simultaneously learning  $\psi_0(a | V)$  for multiple  $a$  by considering a summed loss function

$$L(\psi) = \sum_{a \in \mathcal{A}_0} \omega(a) L^a(\psi) ,$$

where  $\omega(a)$  is any positive weight. Then the oracle regret is as follows:

$$P_0 [L_0(\hat{\psi}) - L_0(\psi_0)] = \sum_{a \in \mathcal{A}_0} \omega(a) P_0 [L_{0,a}(\hat{\psi}) - L_{0,a}(\psi_0)]$$

where  $P_0 [L_{0,a}(\hat{\psi}) - L_{0,a}(\psi_0)]$  achieves the rate established above.  $\square$

## B.4 Further 2-Arm Simulation Results

Additional evaluation metrics for the simulation study discussed in the main body of the paper are included below (Figures B.1, B.2, and B.3). Treating assignment of the treatment rule as a binary classification problem, we calculated the Accuracy, Sensitivity, and Specificity of each estimation method. Again we see that Doubly-Robust estimation with HAL is at or among the top performers in each setting.

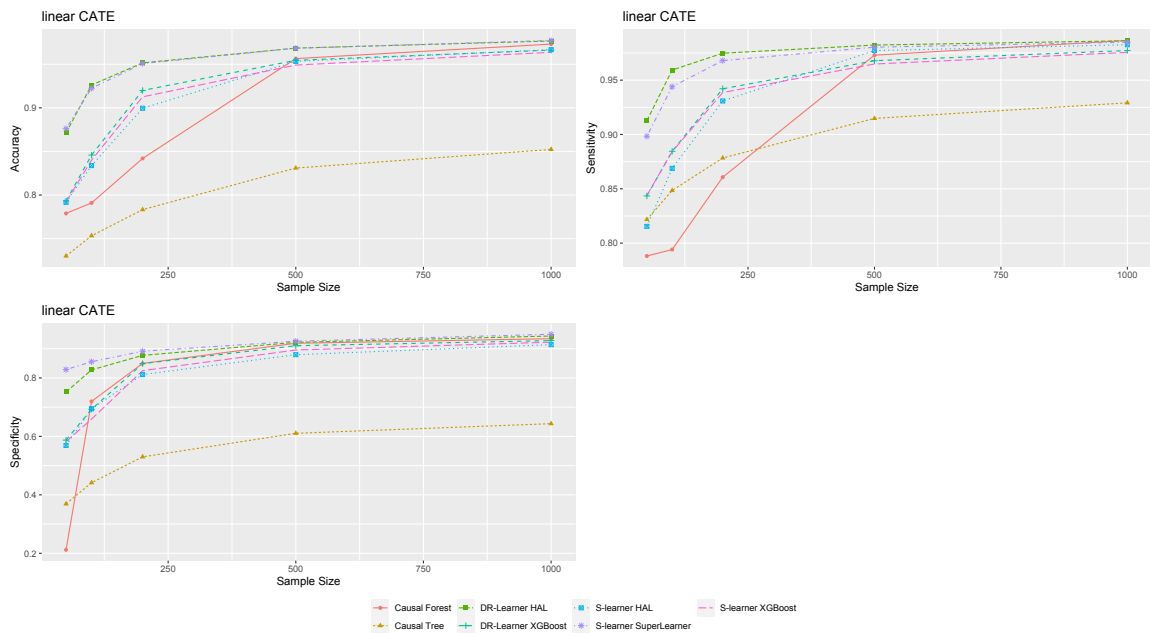


Figure B.1: Accuracy, Sensitivity, and Specificity results for learning the learning the binary optimal treatment rule in the Linear CATE setting.

## B.5 Multi-Arm Simulation Results

We conducted additional simulations to test our methodology in a scenario with three treatment arms. Simulations were conducted under two settings, *linear* and *polynomial* so named for the chosen functional form of the true CATEs. In each setting we considered a continuous outcome  $Y$  and treatment with three arms,  $A \in \{0, 1, 2\}$ . Counterfactual outcome values were calculated by adding standard normal random

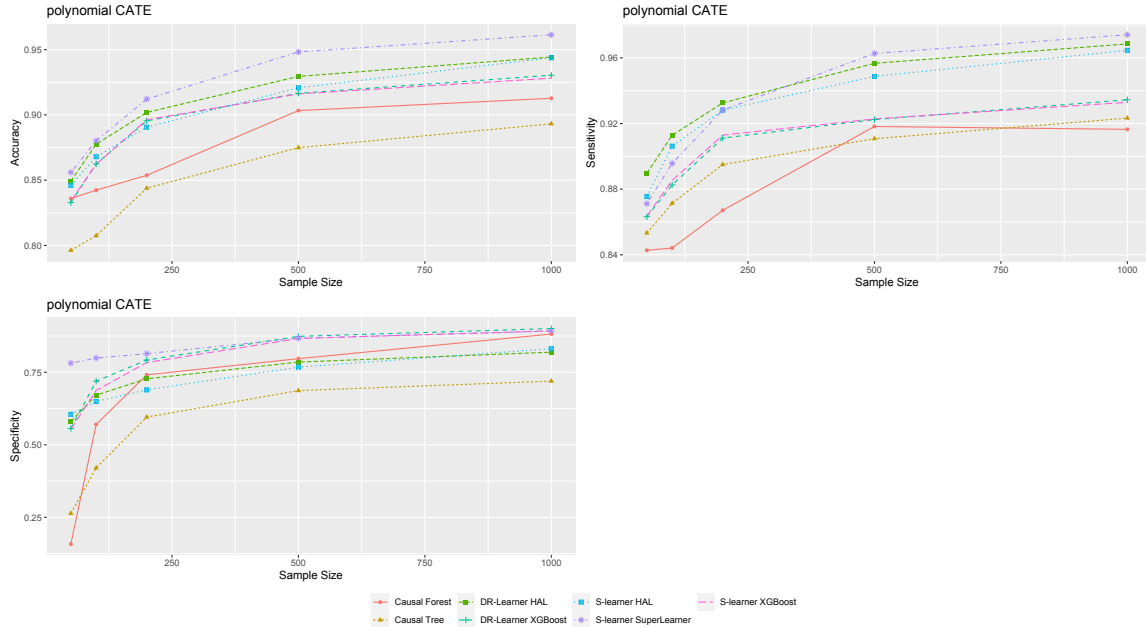


Figure B.2: Accuracy, Sensitivity, and Specificity results for learning the learning the binary optimal treatment rule in the Polynomial CATE setting.

noise to the specified true outcome regression. Treatment values were assigned by randomly sampling from  $\{0, 1, 2\}$ . In each setting, we specified that each treatment have equal probability ( $P(A = a) = 1/3$  for  $a = 1, 2, 3$ ).

We simulated data at sample sizes  $n = 50, 100, 200$ , and  $500$ . We generated data, fit models to estimate the CATEs and OTP and evaluated performances using CATE estimate  $L_2$  norms, estimated policy values, and accuracy of treatment assignment. This process was replicated 1000 times at each sample size, and results were averaged. Below are descriptions of each setting.

### *Linear setting*

Here we considered  $p = 5$  covariates, and set the task of estimating the treatment effect conditional on all five. All covariates were simulated independently from standard normal distributions. The true outcome regression was  $\mu_0(a, w) = \mathbb{1}_1(a)(4 + 5 * w_1) + \mathbb{1}_2(a)(3 + 4 * w_2) + w_2 - 3 * w_3 + 5 * w_3 w_4$ .

### *Polynomial setting*

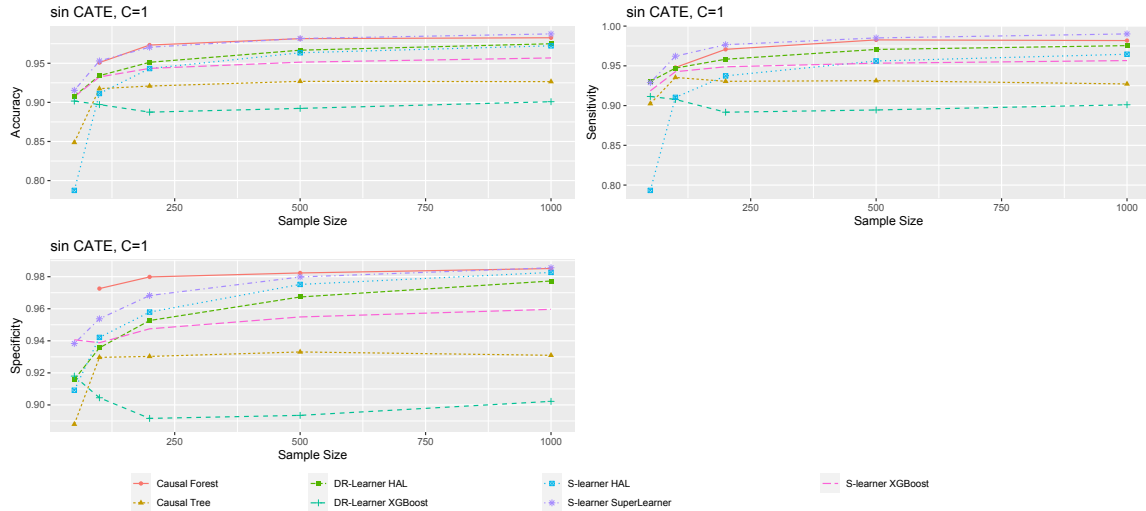


Figure B.3: Accuracy, Sensitivity, and Specificity results for learning the learning the binary optimal treatment rule in the Sinusoidal CATE setting with  $C = 1$ . At  $n = 50$ , Causal Forest showed abnormally low performance and is omitted from the plot to keep other learners distinguishable.

Here we again considered  $p = 5$  covariates and aimed to estimate the CATEs conditional on all 5. This time we specified a polynomial function of covariates for the outcome regression:  $\mu_0(a, w) = \mathbb{1}_1(a)(5 + 2w_1 - 3w_3w_5^2 + 2w_4^3) + \mathbb{1}_2(a)(w_1 + 2w_3w_4^2) + w_2$ .

#### *Interpretation*

Results in these simulations were consistent with the two treatment arm settings. Both S and DR-Learners with HAL consistently show performance superior to Causal Forest (Figure B.5 and Figure B.6). In some cases, the S-Learner with Super Learner shows some marginal improvement over HAL based methods.

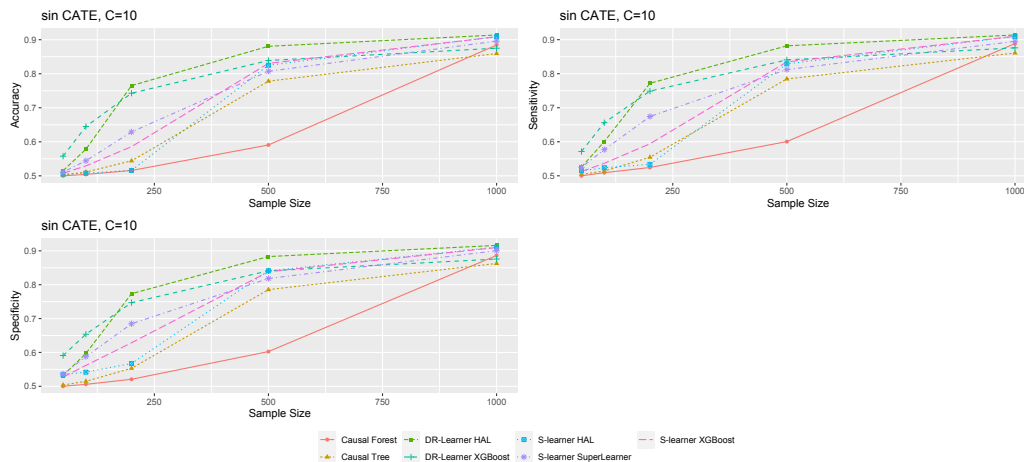


Figure B.4: Accuracy, Sensitivity, and Specificity results for learning the learning the binary optimal treatment rule in the Sinusoidal CATE setting with  $C = 10$ .

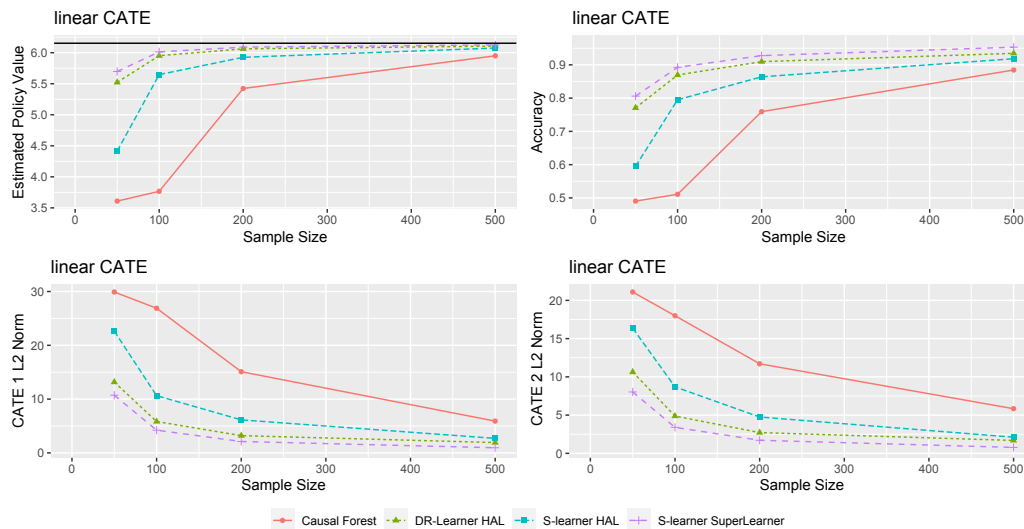


Figure B.5: Multi-arm simulation results in the Linear setting. A simple mean estimator was used to estimate the propensity score for the DR-Learner with HAL. The bottom left and bottom right panels show  $L_2$  norms for  $\hat{\psi}(1 | v)$  and  $\hat{\psi}(2 | v)$  respectively.

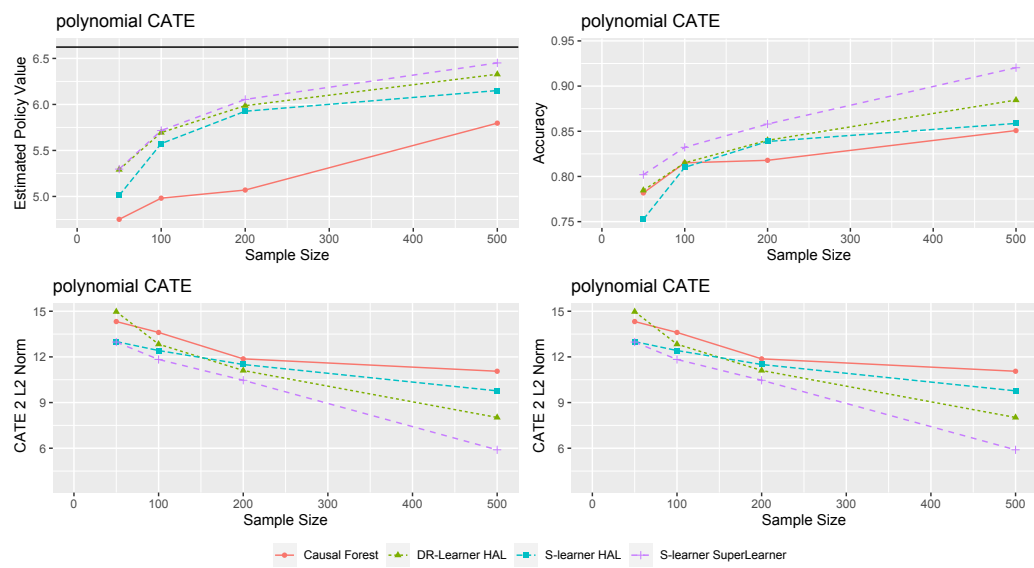


Figure B.6: Multi-arm simulation results in the polynomial setting. A simple mean estimator was used to estimate the propensity score for the DR-Learner with HAL. The bottom left and bottom right panels show  $L_2$  norms for  $\hat{\psi}(1 | v)$  and  $\hat{\psi}(2 | v)$  respectively.

# Bibliography

- [1] Laura Acion, Diana Kelmansky, Mark van der Laan, Ethan Sahker, DeShauna Jones, and Stephan Arndt. Use of a machine learning framework to predict substance use disorder treatment success. *PLOS One*, 12(4):e0175383, 2017. doi: <https://doi.org/10.1371/journal.pone.0175383>.
- [2] S Aeberhard, D Coomans, and O De Vel. Comparison of classifiers in high dimensional settings. *Dept. Math. Statist., James Cook Univ., North Queensland, Australia, Tech. Rep*, 92(02), 1992. doi: [https://doi.org/10.1016/0031-3203\(94\)90145-7](https://doi.org/10.1016/0031-3203(94)90145-7).
- [3] David M Allen. The relationship between variable selection and data agumentation and a method for prediction. *technometrics*, 16(1):125–127, 1974.
- [4] Christophe Amat, Tomasz Michalski, and Gilles Stoltz. Fundamentals and exchange rate forecastability with simple machine learning methods. *Journal of International Money and Finance*, 88:1–24, 2018.
- [5] Susan Athey and Guido Imbens. Recursive partitioning for heterogeneous causal effects. *Proceedings of the National Academy of Sciences*, 113(27):7353–7360, 2016.
- [6] Diogo Ayres-de Campos, Joao Bernardes, Antonio Garrido, Joaquim Marques-de Sa, and Luis Pereira-Leite. Sisporto 2.0: a program for automated analysis of



- cardiotocograms. *Journal of Maternal-Fetal Medicine*, 9(5):311–318, 2000. doi: 10.1002/1520-6661(200009/10)9:5<311::AID-MFM12>3.0.CO;2-9.
- [7] Patrick Bajari, Denis Nekipelov, Stephen P Ryan, and Miaoyu Yang. Machine learning methods for demand estimation. *American Economic Review*, 105(5): 481–85, 2015.
- [8] David Benkeser and Mark Van Der Laan. The highly adaptive lasso estimator. In *2016 IEEE international conference on data science and advanced analytics (DSAA)*, pages 689–696. IEEE, 2016.
- [9] David Benkeser, Maya Petersen, and Mark J van der Laan. Improved small-sample estimation of nonlinear cross-validated prediction metrics. *Journal of the American Statistical Association*, 115(532):1917–1932, 2020.
- [10] Beate Bollig and Ingo Wegener. Improving the variable ordering of obdds is np-complete. *IEEE Transactions on computers*, 45(9):993–1002, 1996. doi: 10.1109/12.537122.
- [11] Leo Breiman. Stacked regressions. *Machine learning*, 24:49–64, 1996.
- [12] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.
- [13] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and Regression Trees*. CRC press, 1984.
- [14] Ching-Chin Chern, Yu-Jen Chen, and Bo Hsiao. Decision tree-based classifier in providing telehealth service. *BMC medical informatics and decision making*, 19(1):1–15, 2019. doi: 10.1186/s12911-019-0825-9.
- [15] Matthew M Churpek, Trevor C Yuen, Christopher Winslow, David O Meltzer, Michael W Kattan, and Dana P Edelson. Multicenter comparison of machine learning methods and conventional regression for predicting clinical deterioration

- on the wards. *Critical Care Medicine*, 44(2):368, 2016. doi: 10.1097/CCM.0000000000001571.
- [16] Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, pages 233–240, 2006.
- [17] David L Donoho et al. High-dimensional data analysis: The curses and blessings of dimensionality. *AMS Math Challenges Lecture*, 1(2000):32, 2000.
- [18] Monroe D Donsker. Justification and extension of doob’s heuristic approach to the kolmogorov-smirnov theorems. *The Annals of mathematical statistics*, pages 277–281, 1952.
- [19] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL <http://archive.ics.uci.edu/ml>.
- [20] Elaine Fehrman, Awaz K Muhammad, Evgeny M Mirkes, Vincent Egan, and Alexander N Gorban. The five factor model of personality and evaluation of drug consumption risk. In *Data science*, pages 231–242. Springer, 2017. doi: [https://doi.org/10.1007/978-3-319-55723-6\\_18](https://doi.org/10.1007/978-3-319-55723-6_18).
- [21] Jerome Friedman, Trevor Hastie, Robert Tibshirani, et al. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.
- [22] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22, 2010. doi: 10.18637/jss.v033.i01. URL <https://www.jstatsoft.org/v33/i01/>.
- [23] Seymour Geisser. The predictive sample reuse method with applications. *Journal of the American statistical Association*, 70(350):320–328, 1975.

- [24] Alan S Gerber, Gregory A Huber, Daniel R Biggers, and David J Hendry. Ballot secrecy concerns and voter mobilization: New experimental evidence about message source, context, and the duration of mobilization effects. *American Politics Research*, 42(5):896–923, 2014.
- [25] Nima S Hejazi, Jeremy R Coyle, and Mark J van der Laan. hal9001: Scalable highly adaptive lasso regression inr. *Journal of Open Source Software*, 5(53):2526, 2020.
- [26] Anning Hu. Heterogeneous treatment effects analysis for social scientists: A review. *Social Science Research*, page 102810, 2022.
- [27] Xin-Lin Huang, Xiaomin Ma, and Fei Hu. Machine learning and intelligent communications. *Mobile Networks and Applications*, 23(1):68–70, 2018.
- [28] Daniel Jacob. Cate meets ml: Conditional average treatment effect and machine learning. *Digital Finance*, 3(2):99–148, 2021.
- [29] Gordon V Kass. An exploratory technique for investigating large quantities of categorical data. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 29(2):119–127, 1980. doi: <https://doi.org/10.2307/2986296>.
- [30] Edward H Kennedy. Towards optimal doubly robust estimation of heterogeneous causal effects. *arXiv preprint arXiv:2004.14497*, 2020.
- [31] Sören R Künnel, Jasjeet S Sekhon, Peter J Bickel, and Bin Yu. Metalearners for estimating heterogeneous treatment effects using machine learning. *Proceedings of the national academy of sciences*, 116(10):4156–4165, 2019.
- [32] M. J. Kusner, J. R. Loftus, C. Russell, and R. Silva. Counterfactual fairness. *ArXiv e-prints*, March 2017. doi: <https://doi.org/10.48550/arXiv.1703.06856>.

- [33] Chunhuan Lao, Mark Elwood, Marion Kuper-Hommel, Ian Campbell, and Ross Lawrenson. Impact of menopausal status on risk of metastatic recurrence of breast cancer. *Menopause*, 28(10):1085–1092, 2021. doi: 10.1097/GME.0000000000001817.
- [34] Charles X Ling, Jin Huang, Harry Zhang, et al. Auc: a statistically consistent and more discriminating measure than accuracy. In *Ijcai*, volume 3, pages 519–524, 2003.
- [35] Razieh Nabi, Daniel Malinsky, and Ilya Shpitser. Learning optimal fair policies. In *International Conference on Machine Learning*, pages 4674–4682. PMLR, 2019.
- [36] Xinkun Nie, Stefan Wager, et al. Learning objectives for treatment effect estimation. *arXiv preprint arXiv:1712.04912*, 2017.
- [37] Sohail Nizam and David Benkeser. Highly adaptive regression trees. *Journal of Evolutionary Intelligence*, 2023+.
- [38] JONATHAN J Oliver, DAVID L Dowe, and CS Wallace. Inferring decision graphs using the minimum message length principle. In *Proceedings of the 5th Australian Joint Conference on Artificial Intelligence*, pages 361–367. World Scientific, 1992.
- [39] Romain Pirracchio, Maya L Petersen, Marco Carone, Matthieu Resche Rigon, Sylvie Chevret, and Mark J van der Laan. Mortality prediction in intensive care units with the super icu learner algorithm (sacula): a population-based study. *The Lancet Respiratory Medicine*, 3(1):42–52, 2015. doi: 10.1016/S2213-2600(14)70239-5.
- [40] Vili Podgorelec, Peter Kokol, Bruno Stiglic, and Ivan Rozman. Decision trees: an overview and their use in medicine. *Journal of medical systems*, 26(5):445–463, 2002. doi: 10.1023/a:1016409317640.

- [41] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2020. URL <https://www.R-project.org/>.
- [42] Alexandros Rekkas, Jessica K Paulus, Gowri Raman, John B Wong, Ewout W Steyerberg, Peter R Rijnbeek, David M Kent, and David van Klaveren. Predictive approaches to heterogeneous treatment effects: a scoping review. *BMC Medical Research Methodology*, 20(1):1–12, 2020.
- [43] Anthony J Rosellini, Francisca Dussailant, José R Zubizarreta, Ronald C Kessler, and Sherri Rose. Predicting posttraumatic stress disorder following a natural disaster. *Journal of Psychiatric Research*, 96:15–22, 2018. doi: 10.1016/j.jpsychires.2017.09.010.
- [44] Steven L Salzberg. C4. 5: Programs for machine learning by j. ross quinlan. morgan kaufmann publishers, inc., 1993, 1994.
- [45] Jun Shao. Linear model selection by cross-validation. *Journal of the American statistical Association*, 88(422):486–494, 1993.
- [46] Mervyn Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the royal statistical society: Series B (Methodological)*, 36(2):111–133, 1974.
- [47] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.
- [48] MJ van deLaan. Efficient and inefficient estimation in semiparametric models. *CWI Tracts*, 1995.
- [49] Mark van der Laan. A generally efficient targeted minimum loss based estimator

- based on the highly adaptive lasso. *The international journal of biostatistics*, 13(2), 2017. doi: 10.1515/ijb-2015-0097.
- [50] Mark J van der Laan. Statistical inference for variable importance. *The International Journal of Biostatistics*, 2(1), 2006.
- [51] Mark J van der Laan, Eric C Polley, and Alan E Hubbard. Super learner. *Statistical applications in genetics and molecular biology*, 6(1), 2007.
- [52] Mark J van der Laan, Alan E Hubbard, and Sara Kherad Pajouh. Statistical inference for data adaptive target parameters. 2013.
- [53] Aad W Van der Vaart. *Asymptotic statistics*, volume 3. Cambridge university press, 2000.
- [54] Aad W Van Der Vaart, Jon A Wellner, Aad W van der Vaart, and Jon A Wellner. *Weak convergence*. Springer, 1996.
- [55] Ashwini Venkatasubramaniam, Julian Wolfson, Nathan Mitchell, Timothy Barnes, Meghan JaKa, and Simone French. Decision trees in epidemiological research. *Emerging themes in epidemiology*, 14(1):1–12, 2017. doi: 10.1186/s12982-017-0064-4.
- [56] Stefan Wager and Susan Athey. Estimation and inference of heterogeneous treatment effects using random forests. *Journal of the American Statistical Association*, 113(523):1228–1242, 2018.
- [57] David H Wolpert. Stacked generalization. *Neural networks*, 5(2):241–259, 1992.
- [58] Marvin N. Wright and Andreas Ziegler. ranger: A fast implementation of random forests for high dimensional data in C++ and R. *Journal of Statistical Software*, 77(1):1–17, 2017. doi: 10.18637/jss.v077.i01.

- [59] Hassan Zeineddine, Udo Braendle, and Assaad Farah. Enhancing prediction of student success: Automated machine learning approach. *Computers & Electrical Engineering*, 89:106903, 2021.
- [60] Heping Zhang, Richard S Legro, Jeffrey Zhang, Leon Zhang, Xiang Chen, Hao Huang, Peter R Casson, William D Schlaff, Michael P Diamond, Stephen A Krawetz, et al. Decision trees for identifying predictors of treatment effectiveness in clinical trials and its application to ovulation in a study of women with polycystic ovary syndrome. *Human Reproduction*, 25(10):2612–2621, 2010. doi: 10.1093/humrep/deq210.
- [61] Milan Zwitter, Matjaz & Soklic. Breast Cancer. UCI Machine Learning Repository, 1988.