Flexibly Estimating Outcome Model Plug-In, IPW Plug-In, and Doubly Robust Estimator Using Gradient-Boosted Decision Trees (GBDTs)

Peter Wu

May 9, 2022

Goal/Purpose/Introduction

This study aims at flexibly estimating the the outcome and propensity scores using specifically the non-parametric machine learning model gradient-boosted decision trees. In many applications, it is common to utilize parametric models such as linear regression and logistic regression to predict the outcome and propensity scores. Are we able to get more accurate predictions for both the outcome in terms of mean-squared error and propensity score in terms of accuracy? Do we necessarily want to always utilize the most accurate model to predict these quantities? What are some of the trade-offs that we have to make when deciding between different models to choose to estimate those quantities?

It is very plausible that when we use a flexible method like gradient-boosted decision trees, we will get a better accuracy in propensity scores. But in doing so, we may get extreme propensity score values, like values that are very close to 0 or 1, which will cause our estimators to in turn have extremely high variances and essentially "blow up". This is the specific quantity of interest we explore in this study.

Furthermore, this problem is challenging from a statistical perspective because it is unclear knowing nothing about a dataset which model would be the most accurate. In statistics, there is "no free lunch". There are cases where fitting a non-parametric model would be preferred while cases where a parametric model may be more practical, but really it all depends on the dataset. From a causal perspective, if the variance of the estimators is slightly higher using the flexible models but the propensity scores are much more accurate, which model should we use? What is the largest magnitude, practically speaking, we can observe these variances grow to while utilizing these flexible methods?

By exploring some concrete datasets related to the above observations, we intend to understand the variance of flexible models better when estimating three specific estimators: outcome model plug-in, IPW plug-in, and the doubly robust estimator.

Data Description

In our simulation study, we simulated four different datasets, each with three features of length 1000, 200 times. The main parameter we change across the datasets is the amount of right-skewed variables in the dataset. The datasets are as follows:

- Dataset 1: All three variables are normally distributed.
- Dataset 2: Two variables are normally distributed. One is Exponentially distributed.
- Dataset 3: One variable is normally distributed. One is Exponentially distributed and the other is Poisson distributed.
- Dataset 4: No variables are normally distributed. One is Exponentially distributed, one is Poisson distributed, and one is Gamma distributed.

More formally, the process below describes the exact data generation process for Dataset 1.

$$
\begin{aligned}
&X_{i1} \sim N(0, 1) \\
&X_{i2} \sim N(0, 10) \\
&X_{i3} \sim N(0, 100) \\
&Y_i(0) \sim N(25 + X_{i1} \cdot X_{i3} + X_{i2}^3 + |X_{i1}^3 \cdot X_{i3}|, 1) \\
&Y_i(1) = Y_i(0) - 100 \\
&e(X_i) = \frac{\exp(\frac{X_{i1}}{10} + \frac{X_{i2}}{100} + \frac{X_{i3}}{100})}{1 + \exp(\frac{X_{i1}}{10} + \frac{X_{i2}}{100} + \frac{X_{i3}}{100})} \\
&Z_i = \text{Bern}(e(X_i)) \\
&Y_i = Z_i Y_i(1) + (1 - Z_i) Y_i(0)
\end{aligned}
$$

Across our creation of the four datasets, the main parameter we change is the first three lines above, specifically the part where we are drawing the three features from a specified random variable distribution.

In Dataset 2, the first three lines are:

$$
\begin{aligned}
&X_{i1} \sim \text{Exponential}(\frac{1}{5}) \\
&X_{i2} \sim N(0, 10) \\
&X_{i3} \sim N(0, 100)
\end{aligned}
$$

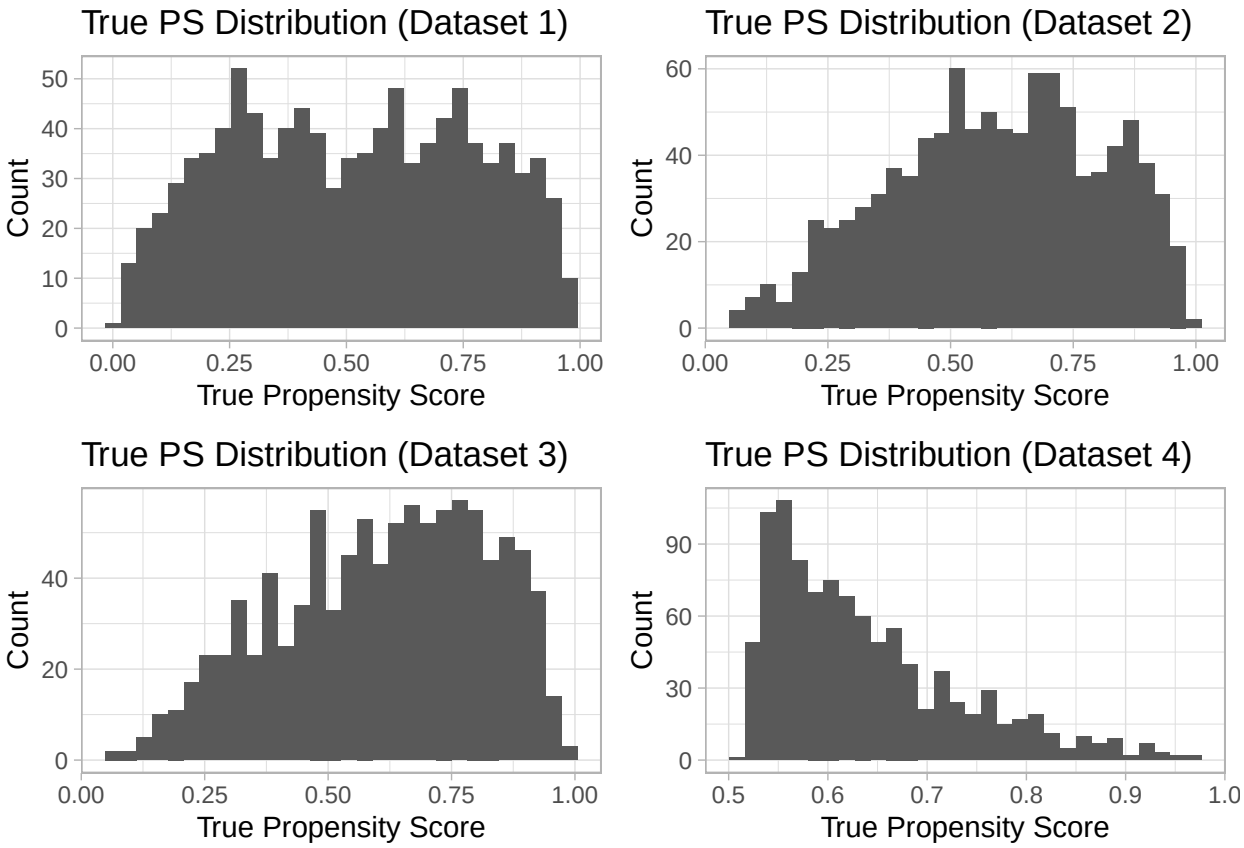
In Dataset 3, the first three lines are:

$$
\begin{aligned}
&X_{i1} \sim \text{Exponential}(\frac{1}{5}) \\
&X_{i2} \sim \text{Poisson}(10) \\
&X_{i3} \sim N(0, 100)
\end{aligned}
$$

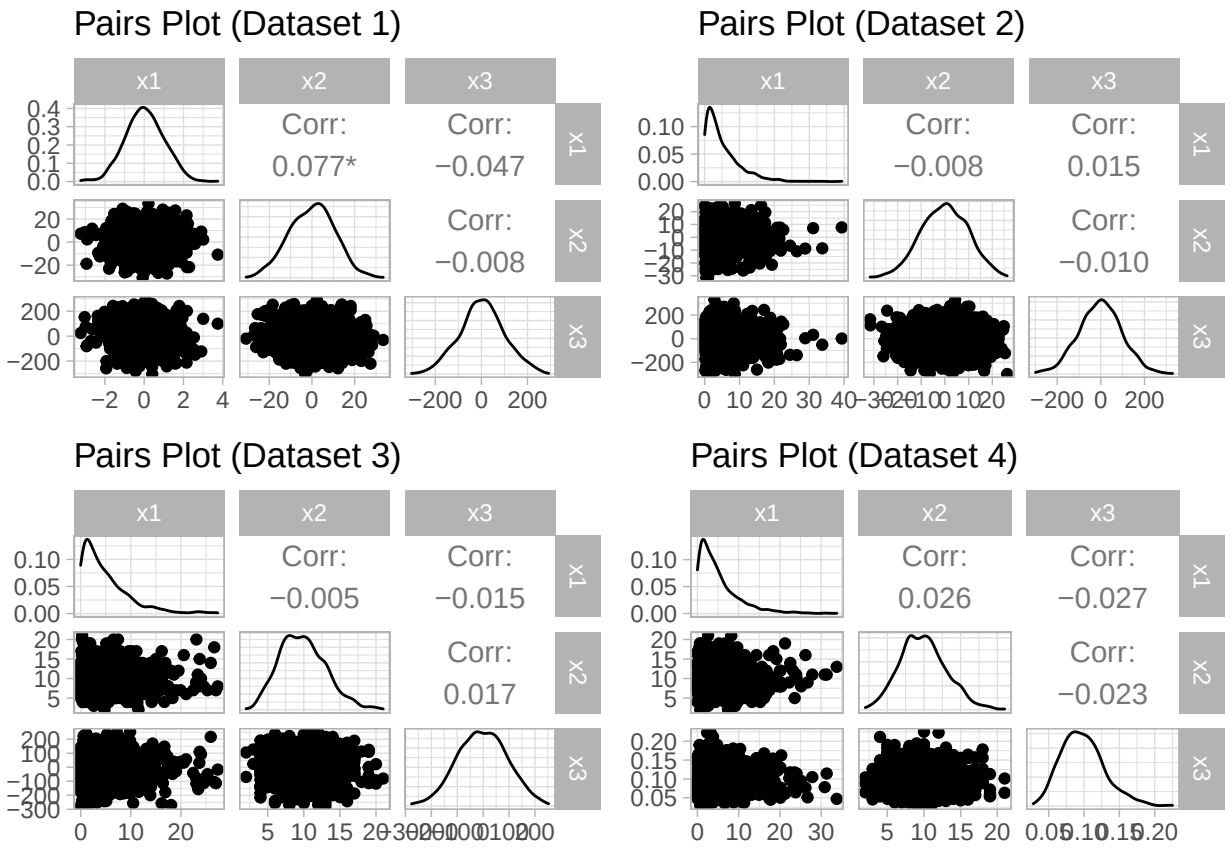
In Dataset 4, the first three lines are:

$$X_{i1} \sim \text{Exponential}(\frac{1}{5})$$
$$X_{i2} \sim \text{Poisson}(10)$$
$$X_{i3} \sim \text{Gamma}(10, 100)$$

We now provide the relevant exploratory data analyses for one instance of each of the four datasets. We provide the true propensity score distribution for each of the four datasets.



Next, we provide the pairs plot for each of the three variables in each dataset.



Methods

We list the three estimators we are interested in computing in this study:

- Outcome Plug-In Model Estimator

$$\hat{\tau}_{\text{reg}} = \frac{1}{N}\sum_{i=1}^{N} \hat{\mu}_1(\mathbf{X}_i) - \hat{\mu}_0(\mathbf{X}_i)$$

- IPW Estimator

$$\hat{\tau}_{\text{IPW}} = \frac{1}{N}\sum_{i=1}^{N} \left[\frac{Z_i Y_i}{e(\mathbf{X}_i)} - \frac{(1-Z_i)Y_i}{1-e(\mathbf{X}_i)}\right]$$

- Doubly Robust Estimator

$$\hat{\tau}_{\text{DR}} = \frac{1}{N}\sum_{i=1}^{N} \left(\left[\hat{\mu}_1(\mathbf{X}_i) + \frac{Z_i\left(Y_i - \hat{\mu}_1(\mathbf{X}_i)\right)}{\hat{e}(\mathbf{X}_i)}\right] - \left[\hat{\mu}_0(\mathbf{X}_i) + \frac{(1-Z_i)\left(Y_i - \hat{\mu}_0(\mathbf{X}_i)\right)}{1-\hat{e}(\mathbf{X}_i)}\right]\right)$$

Our methodology centers around estimating $\hat{\mu}(\mathbf{X}_i)$ and $\hat{e}(\mathbf{X}_i)$ using the flexible machine learning method gradient-boosted decision trees.

A **decision tree** is essentially nested if-else statements based on the features. It is created using measures that quantify how disordered (or ordered) a particular split is. The two most commonly used measures for creating a decision tree are gini impurity and entropy. For example, a split with a low gini impurity is desirable because this would mean the split was very pure (i.e. 100% one class and 0% the other class), meaning that we obtained a lot of information from the split. On the other hand, a high gini impurity would mean something close to a 50%–50% split meaning we did not obtain much information from the split. Given the desired depth of a decision tree, the tree can be constructed by trying to "gain as much information as possible" from the data.

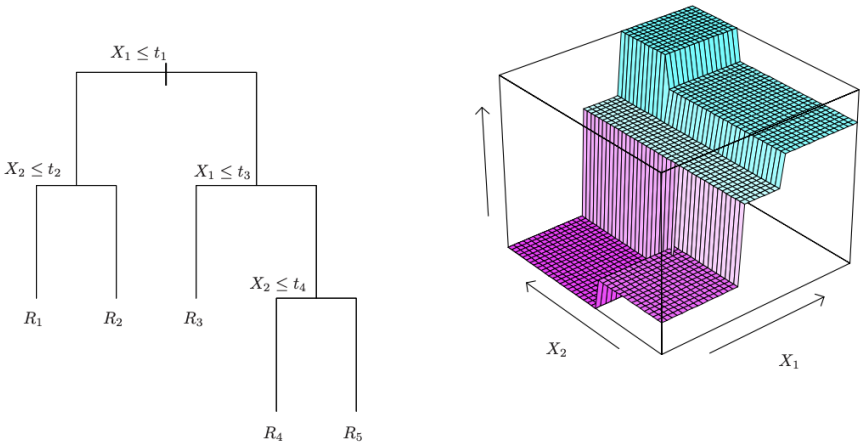


Figure 1: Decision Trees from An Introduction to Statistical Learning: With Applications in R Book by James et al.

The above image on the left is an example of a decision tree. On the image on the right, we can visualize the decision tree in n-dimensional space. The splits from a decision tree amount to essentially partitioning the n-dimensional space into many different hypercuboids.

A **gradient-boosted decision tree** is essentially many decision trees generated one after another and combined in an additive process. We generate many high bias and low variance decision trees, calculate the residuals on the dataset using the trees, and then update our model with those trees. We iteratively do this process of fitting to the new residuals at each step. Intuitively, we can see that this process increases the variance as we add more and more trees, which helps to lessen the high individual biases of each of the trees. We will see later that one the hyperparameters when fitting this particular model is the number of base learners (or decision trees in this instance) that one should specify to be genreated in the final model. When there are a lot of trees fit, we can think of these trees essentially squeezing every ounce of information there can be obtained from the residuals, thus lending itself to overfitting. This is something we have to monitor carefully, and we cannot simply use the default parameters for this given in regular software packages because of this issue. For the purposes of this report, we use `n.trees = 1000` for regression tasks and `n.trees = 100` for classification tasks.

Algorithm 10.3 *Gradient Tree Boosting Algorithm.*

1. Initialize $f_0(x) = \arg\min_\gamma \sum_{i=1}^{N} L(y_i, \gamma)$.
2. For $m = 1$ to M:
   (a) For $i = 1, 2, \ldots, N$ compute
   $$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}.$$
   (b) Fit a regression tree to the targets r_{im} giving terminal regions $R_{jm},\ j = 1, 2, \ldots, J_m$.
   (c) For $j = 1, 2, \ldots, J_m$ compute
   $$\gamma_{jm} = \arg\min_\gamma \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \gamma\right).$$
   (d) Update $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$.
3. Output $\hat{f}(x) = f_M(x)$.

Figure 2: Gradient-Boosting Algorithm from Elements of Statistical Learning (ESLR) Book by Hastie et al.

The above image is the exact algorithm for gradient-boosting, where the base learner we use are the decision trees. Note that because the derivative of $\frac{1}{2} \cdot$ (Squared Error) is equal to the residual, we can simply calculate what is known as the pseudo-residual on the loss function of our choice.

Furthermore, in estimating the doubly robust estimator we utilize sample-splitting where we first randomly split the dataset in half (D_1), fit our models to the first half, and then run the same exercise of fitting the estimators on the second half (D_2) of the data. We repeat the process using the second half as train and then the first half as test, and then our result will be the average of the two quantities we obtain from the first run and the second run.

Results/Interpretations

Dataset 1 results for Outcome Model Estimator, IPW Estimator, and Doubly Robust Estimator:

Table 1: Dataset 1 Outcome, IPW, and DR Estimator 95% Confidence Interval and MSE Using Linear + Logistic Regression and GBDTs

	2.5%	50%	97.5%	MSE
Outcome LR	-493.3830	-110.74849	232.16848	34017.718
Outcome GBDT	-194.5849	-29.28249	23.92506	6957.882
IPW LogReg	-590.8801	-100.83616	492.69017	91611.448
IPW GBDT	-347.5488	-60.12781	360.02384	51391.697
DR LR + LogReg	-648.8715	-105.34817	335.87347	77303.616
DR GBDT	-467.1845	-84.39248	245.88297	46614.320

Dataset 2 results for Outcome Model Estimator, IPW Estimator, and Doubly Robust Estimator:

Table 2: Dataset 2 Outcome, IPW, and DR Estimator 95% Confidence Interval and MSE Using Linear + Logistic Regression and GBDTs

	2.5%	50%	97.5%	MSE
Outcome LR	-70707.137	-18745.459	8413.360	861074053
Outcome GBDT	-2603.241	0.000	3450.403	7573542
IPW LogReg	-63929.073	11287.561	41281.953	1239926469
IPW GBDT	-9255.096	21775.536	56042.217	781372765
DR LR + LogReg	-140817.179	7415.836	44295.427	2046294674
DR GBDT	-28622.699	6277.099	43563.260	429848414

Dataset 3 results for Outcome Model Estimator, IPW Estimator, and Doubly Robust Estimator:

Table 3: Dataset 3 Outcome, IPW, and DR Estimator 95% Confidence Interval and MSE Using Linear + Logistic Regression and GBDTs

	2.5%	50%	97.5%	MSE
Outcome LR	-51680.67	-16239.492	6924.707	593599388
Outcome GBDT	-3963.10	0.000	4253.550	5642791
IPW LogReg	-66535.22	14318.771	41174.195	3278809408
IPW GBDT	-12057.19	22575.894	51937.147	831717615
DR LR + LogReg	-80389.64	11989.689	44475.467	3124168336
DR GBDT	-26062.41	7986.131	45691.466	397719123

Dataset 4 results for Outcome Model Estimator, IPW Estimator, and Doubly Robust Estimator:

Table 4: Dataset 4 Outcome, IPW, and DR Estimator 95% Confidence Interval and MSE Using Linear + Logistic Regression and GBDTs

	2.5%	50%	97.5%	MSE
Outcome LR	-181.2201	-115.88514	-38.769155	1606.1474
Outcome GBDT	-111.9685	-81.47113	-34.987764	812.3996
IPW LogReg	-264.8131	-92.96386	13.210968	6535.8215
IPW GBDT	-116.1799	-37.39737	53.505014	5821.2130
DR LR + LogReg	-270.6235	-105.15385	-1.931346	7372.5093
DR GBDT	-150.9438	-91.90315	-46.271083	796.0100

In Table 1, we can see that the GBDT MSE's were generally better than their Linear Regression and Logistic Regression counterparts in estimating the three estimators. We note that the 50% point estimate from Linear Regression and Logistic Regression was closer to the true average treatment effect of -100 than GBDT. The GBDT variance was also smaller than that of Linear Regression and Logistic Regression. The MSE on a whole was better for GBDT, but the point estimates it was giving were not in the ballpark. We witness these same general trends across Table 2, 3, and 4.

The intuition behind this is likely that the GBDT models are overfitting, which means the variance is higher but overall it is doing better in terms of prediction accuracy averaged over all the training datasets using MSE. Furthermore, it is also interesting to note that in Dataset 1 and Dataset 4, the doubly robust estimator performs much better than the Outcome and IPW estimator. Because only `n.trees = 100` were used for classification tasks, it may appear that the IPW estimations are a little more biased, and perhaps more trees are needed.

The wild fluctuations we are observing in Datasets 2 and 3 indicate that the nature of the dataset will dictate a large portion of which model will do better. When there are a combination of features with varying features (such as symmetricness or skewness), the models exhibit extremely large MSE's which is not ideal. This usually signals the need for techniques like data standardization or normalization where we put all the features on the same scale. In general, doing that will yield more consistent results when we feed them into models and is a good procedure to do for prediction.

We would postulate that in Datasets 2 and 3 perhaps bias is more of a problem as these estimates are completely missing the target while in Datasets 1 and 4 reveal variance could be more of an issue. GBDT is able to generate more accurate predictions, but at the cost of being slightly more biased in Datasets 1 and 4. Perhaps adding more trees in this instance will help alleviate this issue. It truly does seem that there are both strengths and weaknesses to how GBDT can predict. It definitely has the potential to predict far better than normal parametric techniques, but it has the potential to open up the can of worms in terms of other

problems it has to deal with like the bias-variance tradeoff and finding the right number of trees to fit for both classication and regression tasks. Too little trees equals high bias while too many trees will equate to high variance.

Conclusions/Future Work

Overall, we have learned that even though it is conceivable that we can improve the accuracy of propensity scores algorithm with more flexible non-parametric methods like gradient-boosting decision trees, it is prone to other issues like the bias-variance tradeoff in determining the number of trees to fit. Fit too little trees, and one has to deal with a high bias problem. Fit too many trees, and one has to deal with a high bias problem. Thus, one would really have to perform a grid search (trying all possible combinations of the number of trees) to find the most optimal one for this bias-variance tradeoff. Because the high bias or high variance issue varied across the different datasets with a differing number of right-skewed distributions, this also illustrates that we cannot tell which problem we will have (whether it is high bias or high variance) initially. Different datasets will lead to different problems related to the GBDTs, and one has to really look into the nuts and bolts of the algorithm on the dataset itself to understand how the model is performing.

Nonetheless, we have shown that it is great to have another tool right up our toolbelt in being able to apply a non-parametric method like gradient-boosting decision trees for both the classification and regression tasks. The non-parametric methods possess much more scope and power than the parametric methods, but "with great power comes great responsibility". One must be extremely mindful of not just squeezing out every ounce of predictive power from the dataset. Picking the right number of trees is an art form and should be treated as a problem in it of itself because it has a great effect on the bias-variance tradeoff and the actual predictions the model will yield.

Some limitations of the project are as follows. We only looked at four datasets in this study and specifically altered the number of right-skewed distributions in the dataset, but in reality there are far more complex and more "messy" datasets that the methods we used to predict will interact with. We could tweak the way we simulated our data, and perhaps make it more complex or more random. We could also try another different flexible non-parametric model like random forests to see if the same results we got here will still hold. As mentioned previously, we can perform grid search on `n.trees` to optimize the bias-variance tradeoff issues. I would be interested in also using a train-test process for predicting actual propensity score outcomes using metrics like ROC-AUC and confusion matrices.

Furthermore, we can somehow take into account the fact that gradient-boosting decision trees can perform both classification and regression tasks. It seems like the outcome and propensity scores tasks were two separate tasks. Perhaps there is some conceptual way we can combine them and then only do the task once but have some result that is able to use that information and have "applied it twice" in a causal setting.