Applying machine learning

Lucia Gomez Llactahuamani

Introduction

Mullainathan and Spiess (2017) note that machine learning revolves around the problem of prediction $\hat{Y}$ while many economic applications instead revolve around parameter estimation ($\hat{\beta}$). Therefore, applying machine learning algorithms to economics requires finding relevant prediction tasks. One category of such applications appears when the key object of interest is the parameter β, but the inference procedures contain a prediction task. For example, the first stage of instrumental variables (IV) regression is effectively a prediction. [1]

Since most machine learning methods are built for prediction and typically outperform ordinary least squares at this task, using machine learning methods for estimation of the first stage in an IV setting with very many potential instrumental variables can lead to more accurate first-stage predictions and more precise second-stage estimates. Mogstad, Torgovitsky, and Walters (2021) report a survey of empirical papers using instrumental variables (IVs) that were published in leading economics journals since 2000. More than half of these papers report results from a specification with multiple IVs for a single treatment, typically combined using two-stages least squares (2SLS). The authors classify these papers into three types by the relationship between their multiple IVs. A 67% of these studies use multiple economically distinct instruments, a 13% includes studies that use interaction of covariates with a single instrument, the remaining 19% of the studies use multiple functions of a single instruments. Currently, several papers adapt machine learning techniques for selecting a subset of large number of instruments to predict the first stage of a linear regression (e.g. Belloni et al. 2012). However, the risk of applying out-of-the-box machine learning methods for 2SLS type applications are less clear (Lennon, Rubin, and Waddell 2021).

In this article I study the performance of three-based methods (random forest and boosting) in an IV setting with a binary endogenous explanatory variable. In particular, I apply these machine learning methods to estimate the returns to college attendance, simulating the data used by Carneiro, Heckman, and Vytlacil (2011).

Methodology

In this section I provide a brief introduction to instrumental variables and two tree-based machine learning techniques: gradient boosting and random forest.

Instrumental variables

Consider a linear population model

$$(1)\ Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_K x_K + u$$
$$E(u) =, Cov(x_K, u) = 0, k = 1, 2, ..., K-1$$

but were x_K might be correlated with u. In other words, the explanatory variables $x_1, x_2, ..., x_{K-1}$ are exogenous but x_K is potentially endogenous in equation (1). In applied econometrics endogeneity usually arises in one of three ways: omitted variables, measurement error and simultaneity. Ordinary least squares

[1] This point has been recognized in the literature [@belloni2012sparse;@mullainathan2017machine;@angrist2022machine].

(OLS) estimation of equation (1) generally results in inconsistent estimators of all the β_i if $Cov(x_K, u) \neq 0$. In the simulation

The method of IV provides a general solution to the problem of an endogenous explanatory variable. To use the IV approach with x_K endogenous, we need an observable variable z_1 not in equation (1), that satisfies two conditions.

- First, exclusion restriction, z_1 must be uncorrelated with u: $Cov(z_1, u) = 0$, therefore the excluded instruments have no effect on outcome except through the treatment variable.
- Second, instrument relevance, z_1 is correlated with the regressor x_K.

The IV method is a two step procedure. Consider the familiar setting of the returns to schooling that we are going to use throughout this study. An extensive literature argues that OLS estimates of the return to schooling may be biased by unobserved differences in ability (e.g., Card 2001). The two-equation system describing schooling (S_i) and log wages (Y_i) for individual i is the following:

$$(2)\ S_i = 1[\psi + Z_i\pi + X_i\delta + v_i > 0]$$

$$(3)\ Y_i = \alpha + \beta S_i + X_i\gamma + u_i$$

In general, taking care of endogeneity of non-linear models like the one with binary endogenous explanatory variable is more challenging than in linear models. For instance, applying 2SLS reasoning directly to nonlinear models leads to the so-called forbidden regression and inconsistent estimates (Hausman 1975). In the case of endogenous explanatory variable that happens to be a dummy variable we might use a nonlinear first stage such as logit or probit model.

Random Forest

In standard classification or regression forests as proposed by Breiman (2001), the prediction for a particular test point x is determined by averaging predictions across an ensemble of different de-correlated trees. Individual trees are grown by greedy recursive partitioning, that is, we recursively add axis-aligned splits to the tree, where each split is chosen to maximize the improvement to model fit. The trees are randomized using boostrap (or subsample) agregation, whereby each tree is grown in a different subset of the training data, and random split selection that restricts the variables available at each step of the algorithms.

A tree is represented as $f(x)$ and a random forest (regression) predictor with B number of bootstrapped tress can be written as:

$$(4)\ F(x) = \frac{1}{B}\sum_{b=1}^{B} f_b x$$

In detail, random forest bootstraps the training data B number of times by randomly selecting m variables form p input variables each time before another split is made. Selecting only a subset of variables decorrelates the trees and prevents over fitting by decreasing the variance of the predicted values. This limitation of cutting only in p variables prevents strong predictors from dominating all the trees.

Boosting

Friedman (2001) introduces gradient boosting as an additive ensemble model where a weighted sum of base tree models are added together. The trees are grown sequentially, unlike random forest where each tree are grown separately and then averaged. At each iteration, a new tree is added to a weighted sum of all previous trees.

The model is initialized at the average of the outcomes, $\bar{y}$ for N observations. At each subsequent iteration, the model parameters are chosen to correct the error, or loss function [2], from the previous iteration with

[2]There is a variety of different loss functions, e.g. sum of squares loss function $L(y, F(x)) = -\frac{1}{2}(y - F(x))^2$

an additional tree and a subsample of the original data. Suppose there are M iterations, then the model is defined

$$(5)\ F_M(x) = \sum_{m=1}^{M} \rho f_m(x)$$

where $f_m(x)$ is the tree at iteration m and ρ is a shrinkage parameter to prevent overfit and to slow down the learning process. By setting ρ to be a smaller number, usually between 0.01 and 0.1, this put small weight on each additional tree and makes the final prediction less sensitive. $f_m(x)$ is usually chosen to be a stump where the input space is cut by one variable; the deep of the trees can be increased but performs best with few terminal nodes.

Simulation Studies

In this section the results of the simulations studies are presented to assess the performance of the different estimators introduced in the last section in the setting of a model with a binary endogenous explanatory variable. The baseline data generating process (DGP) for all the cases we consider in this section resembles to the DGP we get when we simulate the data used by Carneiro, Heckman, and Vytlacil (2011) to estimate the returns to college attendance. The results of different scenarios are discussed. I start with a very simple low-complexity scenario with a single binary endogenous regressor and a single instrument. Later I move to a high-complexity scenario where there are many available instruments but there exits only a small set of strong instruments. Next I consider another high complexity case, a linear model with many instruments, all of which are weak. Finally, I fully consider the model employed by Carneiro, Heckman, and Vytlacil (2011) to estimate the returns to college attendance.

Case 1: One instrument no covariates

In order to examine the performance of non-linear three-based methods in 2SLS I start with a very simple low-complexity scenario with a single binary endogenous regressor and a single instrument. This case allow us to test how replacing first stage OLS with either boosted trees or random forests perform when there is little to be gained from variable selection.

The data $D = \{(y_i, x_i, z_i)_i | i = 1, 2, ..., N\}$ is generated using the following GDP:

(6);

$$y = \beta x + u$$

(7);$x = 1[\alpha z + v > 0]$ (8);

$$\begin{pmatrix} u \\ v \end{pmatrix} = N(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma_u & \sigma_{uv} \\ \sigma_{vu} & \sigma_v \end{bmatrix})$$

(9);

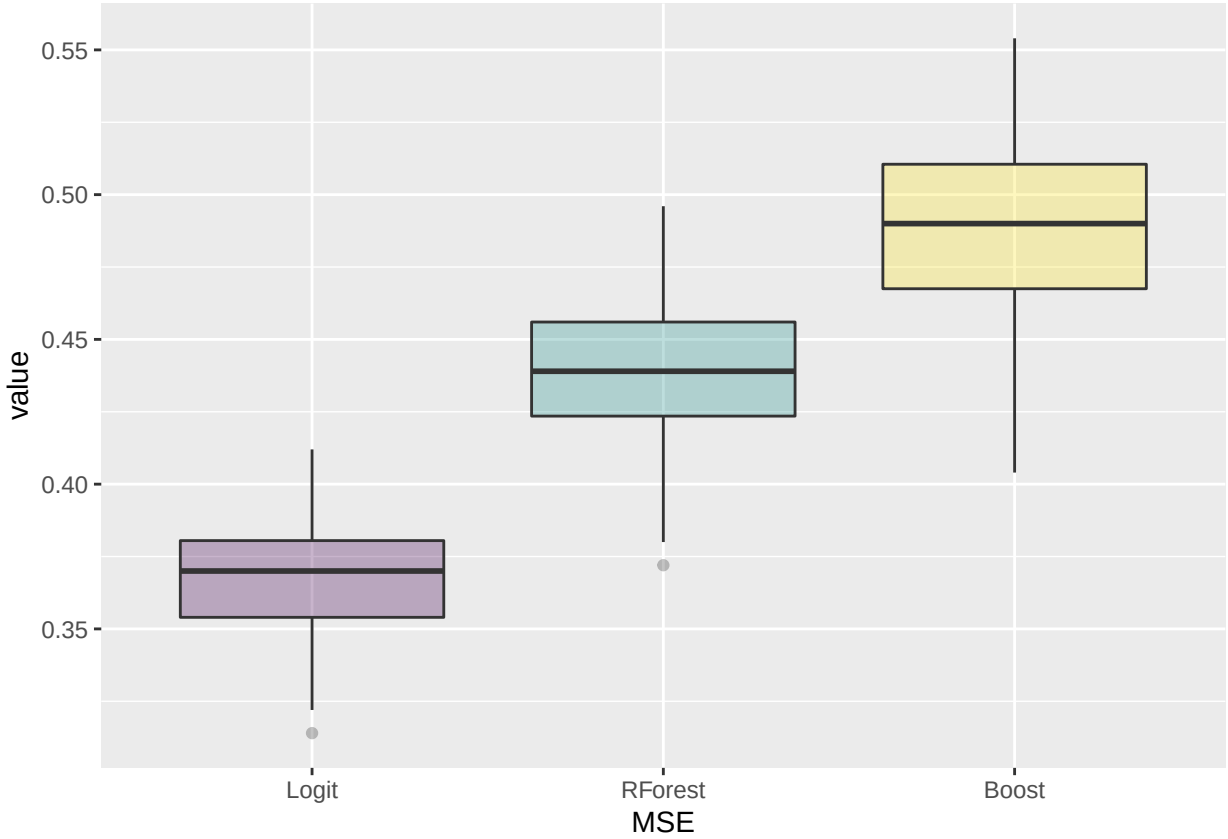
$$z \sim N(0, 1)$$

Where $\beta = (\beta_0, \beta_1) = (-0.9, 0.75)$ are the parameters of interest. We also let $\sigma_u = 1$,$\sigma_v = 1$. The correlation structure in (8), leads to $E(x.u) \neq 0$ implying an endogeneity issue. We consider a sample size n of 1000; we consider a value of sigma of σ_{uv}: 0.3. We assume $\alpha = (\alpha_0, \alpha_1) = (0.3, 0.3)$.

For the first-stage prediction I expect logit outperforms non-linear models like random forest or boosted tree since there is no need of variable selection. For the second stage, I can expect that 2SLS, which uses logit in the first-stage and OLS in the second-stage, yields consistent and unbiased estimates of the true parameter. I can also expect that the OLS estimator that entirely ignores endogeneity generates a biased beta. Regarding the tree-based methods, I can expect they perform poorly given a previous simulation studies performed in low-complexity scenarios (Lennon, Rubin, and Waddell 2021; Angrist and Frandsen 2022).

Table 1: Test MSE in the first-stage prediction of a 2SLS

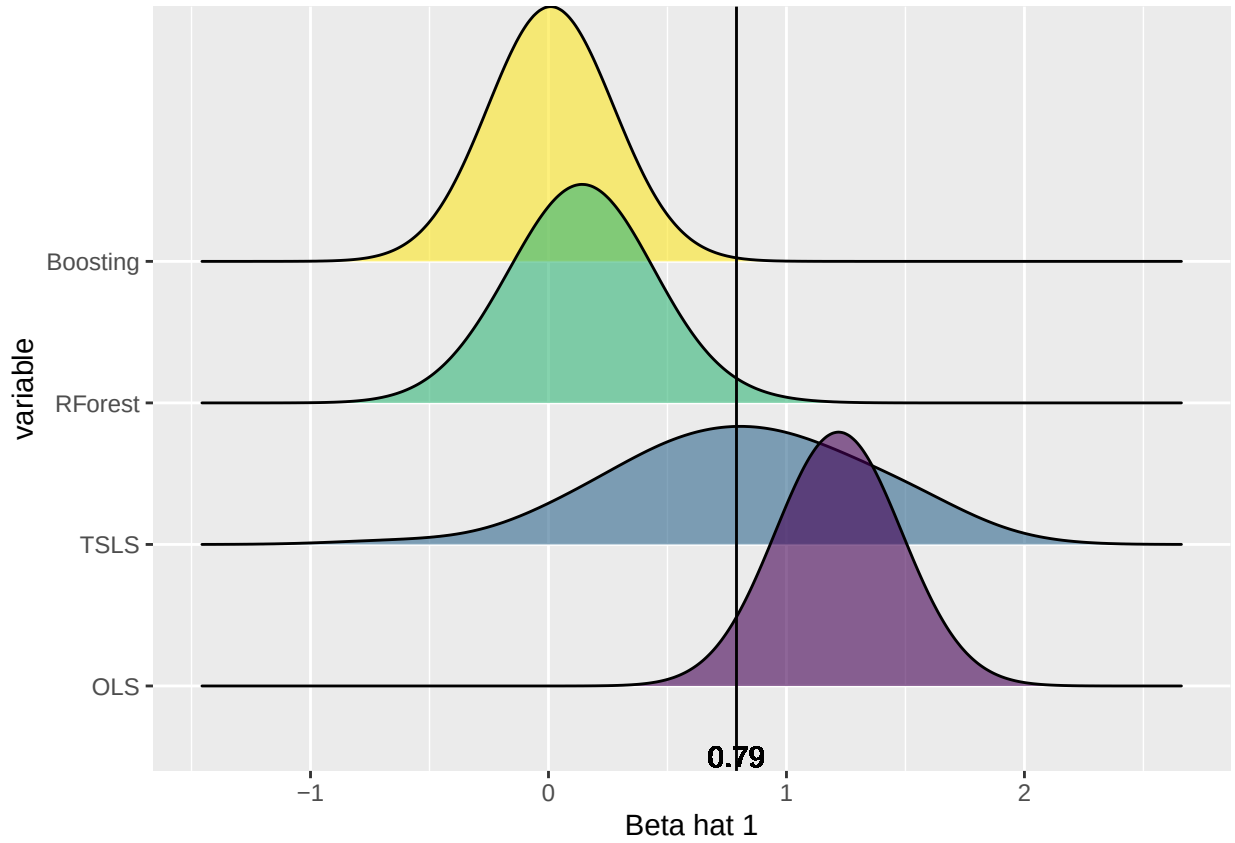
	Train	Test
Logit	0.36674	0.36774
randomForest	0.44306	0.43896
Boost	0.33382	0.48952




```
## Warning: Ignoring unknown parameters: text
```

Table 2: Beta hat 1 distribution across competing two-stage methods

	Coefficient	Std. dev.	Bias
OLS	1.2190903	0.0894529	-0.4690903
TSLS train	0.7583112	0.5026004	-0.0083112
TSLS test	0.8133937	0.5026245	-0.0633937
randomForest train	0.0786142	0.1725044	0.6713858
randomForest test	0.1414581	0.1731754	0.6085419
Boost train	0.2829207	0.0678792	0.4670793
Boost test	0.0109307	0.0740360	0.7390693



I showed that highly non-linear tree-based methods (random forest and boosted forest) can amplify bias, providing parameter estimates farther from truth similar to naive OLS regressions that ignore endogeneity. In this low complexity scenario, tree-based 2SLS methods are overkill: neither variable selection, non-linearity are necesary. As our results demonstrate, random forest and boosting can increase bias relatively to 2SLS (with logit) and even endogenous OLS.

Case 2: Many instruments with strong sparcity and no covariates

As a high-complexity case, we construct a DGP with two extensions. This DGP allows the researcher to customize instruments' strength, and with many instruments. In the tree high-complexity cases injecting boosting or random forest in the first stage, on average, produces more biased estimates than naive endogenous OLS. In short, one can worsen endogeneity issues by using ML-based 2SLS estimators.

The data $D = \{(y_i, x_i, z_i)_i | i = 1, 2, ..., N\}$ is generated using the following GDP. (2);

$$y = \beta_0 + \beta_1 x + u$$

(3);$x = 1[\alpha_0 + H(z)'\alpha_1 + v > 0]$ (4);

$$\begin{pmatrix} u \\ v \end{pmatrix} = N(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma_u & \sigma_{uv} \\ \sigma_{vu} & \sigma_v \end{bmatrix})$$

(5);

$$z \sim N_J(0, 1)$$

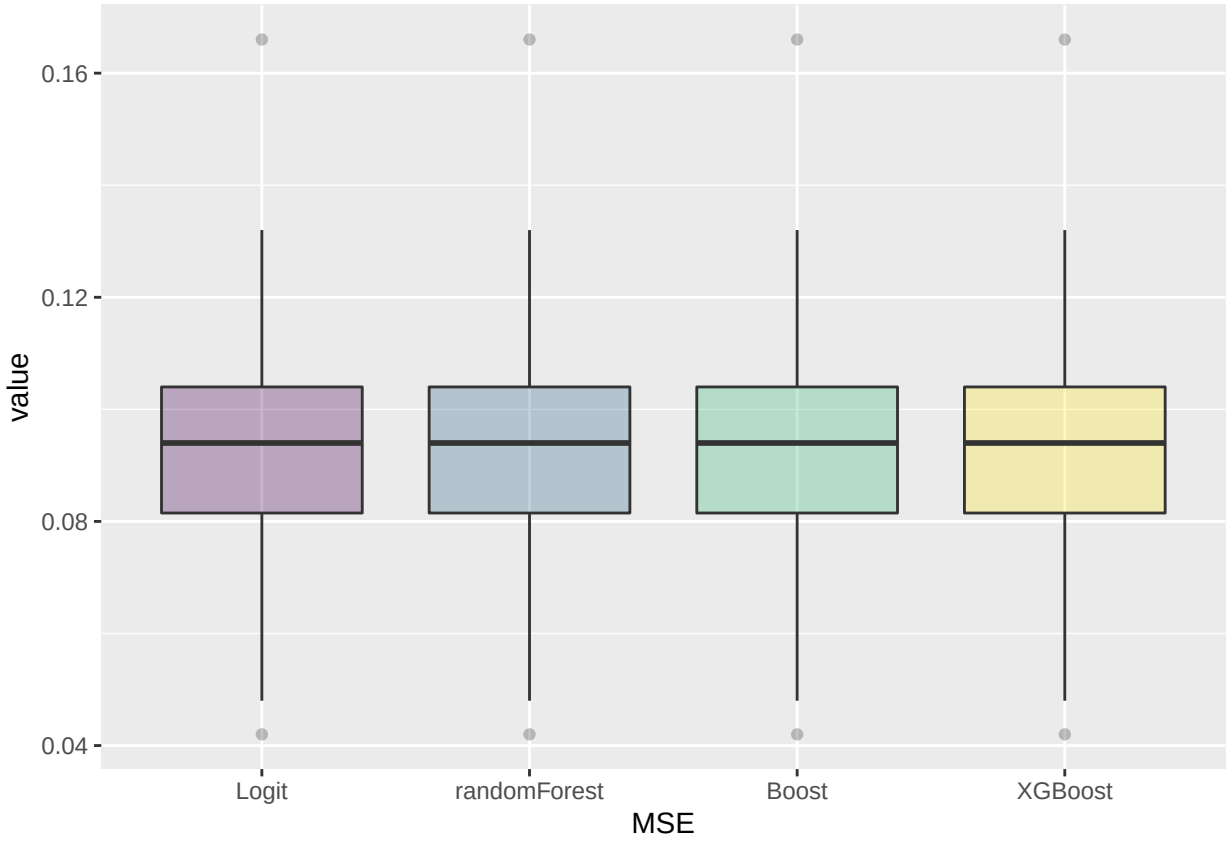
Where $\beta = (\beta_0, \beta_1) = (-0.9, 0.75)$ are the parameters of interest. I also let $\sigma_u = 1$,$\sigma_v = 1$ and $J = 100$ for all of our simulations. The correlation structure in (4), leads to $E(x.u) \neq 0$ implying an endogeneity issue. The sample size n is 1000; I consider for σ_{uv}: 0.3. For $H(.)$, I assume, $H(z) = [1, 1, ..., 1, 1, 0, 0, ..., 0, 0]z$, with the

1s representing s strong instruments, s= 25. The strong sparcity assumption is met because a small subset of the 100 candidate instruments are valid. I use $\alpha_1 = \frac{5}{s}$ which ensures that the instruments have the same impact on x independent of the value of s.

For the first-stage prediction I expect logit under performs or performs similar to non-linear models like random forest or boosted tree since it is needed variable selection. For the second stage, I can expect that 2SLS, which uses logit in the first-stage and OLS in the second-stage, yields consistent and unbiased estimates of the true parameter. I can also expect that the OLS estimator that entirely ignores endogeneity generates a biased beta 1. Regarding the tree-based methods, I can expect that boosting performs poorly given a previous simulation study performed in high-complexity scenario (Lennon, Rubin, and Waddell 2021). However, I would expect random forest to perform well.

Table 3: Test MSE in the first-stage prediction of a 2SLS

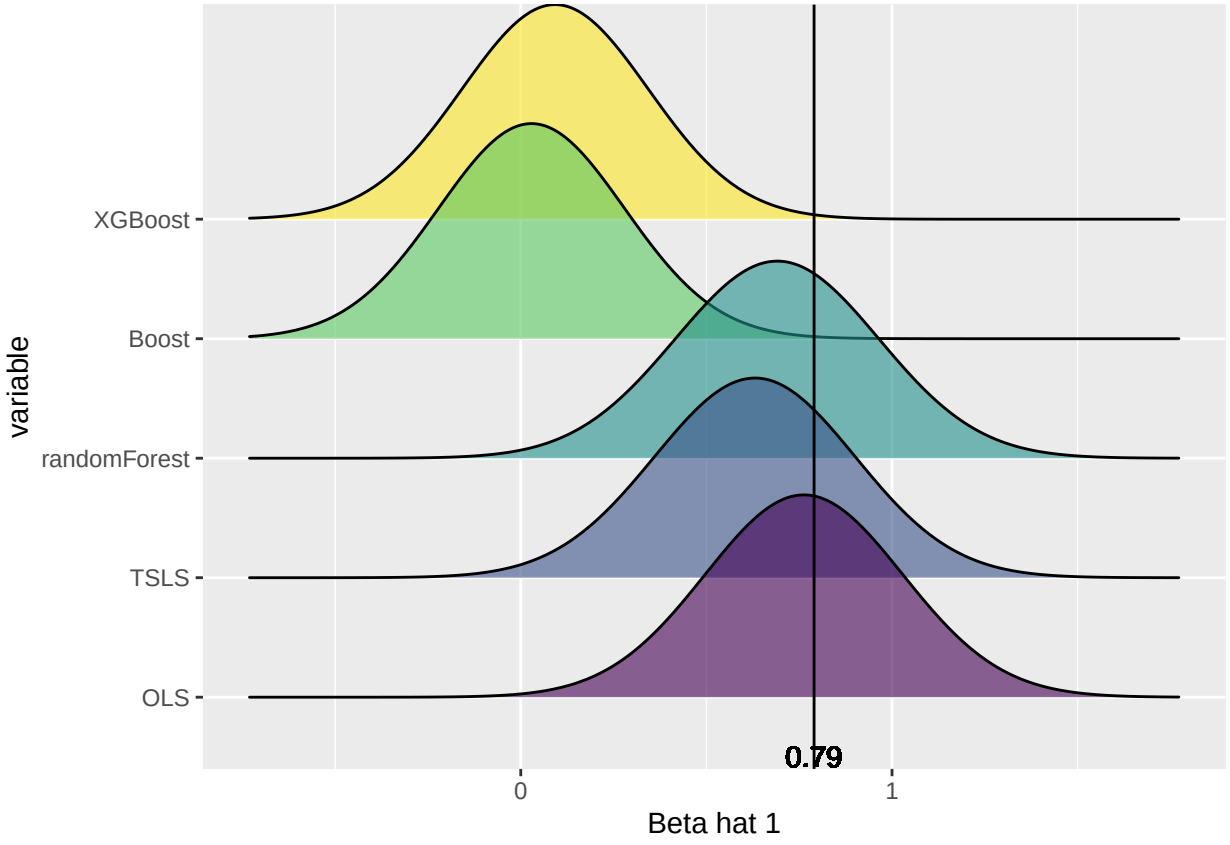
	Train	Test
Logit	0e+00	0.09248
randomForest	0e+00	0.09248
regression_forest	6e-04	0.49838
Boost	0e+00	0.09248
XGBoost	0e+00	0.09248




```
## Warning: Ignoring unknown parameters: text
```

Table 4: Beta hat 1 distribution across competing two-stage methods

	Coefficient	Std. dev.	Bias
OLS	0.7630221	0.0894472	-0.0130221
TSLS train	0.7440293	0.0895129	0.0059707
TSLS test	0.6297210	0.0922472	0.1202790
randomForest tr	0.8227180	0.0991156	-0.0727180
randomForest ts	0.6896219	0.1014390	0.0603781
regression_forest tr	0.8255292	0.0999375	-0.0755292
regression_forest ts	0.0095527	0.1067887	0.7404473
Boost train	0.0346703	0.0041711	0.7153297
Boost test	0.0289572	0.0042700	0.7210428
XGBoost train	0.7498166	0.0902092	NA
XGBoost test	0.6262645	0.0923475	NA



Ensemble methods such as random forests or gradient boosted trees combine the results of multiple trees in order to improve prediction accuracy and to reduce variance. The last amplify bias. This variance-reduction aspect is particularly relevant for non-linear methods Tree-based methods. The results of this second case (many instruments with strong sparcity) shows that boosting can amplify bias, providing parameter estimates farther from truth. Random forest performs well and it is very similar to the TSLS. It is surprising that the naive OLS performs well, even better than boosting.

Case 3: Many week instruments no covariates

We next consider the case of a linear causal model with many instruments, all of which are weak- The issue with many week instruments is that, when the sparcity assumption breaks down, variable selection methods like random forests and boosting tend to select any variable or select all variables, which leads to poor asymptotics.

The data $D = \{(y_i, x_i, z_i)_i | i = 1, 2, ..., N\}$ is generated using the following GDP. (2);

$$y = \beta_0 + x\beta_1 + u$$

(3);

$$x = 1[\alpha_0 + H(z)'\alpha_1 + v > 0]$$

(4);

$$\begin{pmatrix} u \\ v \end{pmatrix} = N(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma_u & \sigma_{uv} \\ \sigma_{vu} & \sigma_v \end{bmatrix})$$

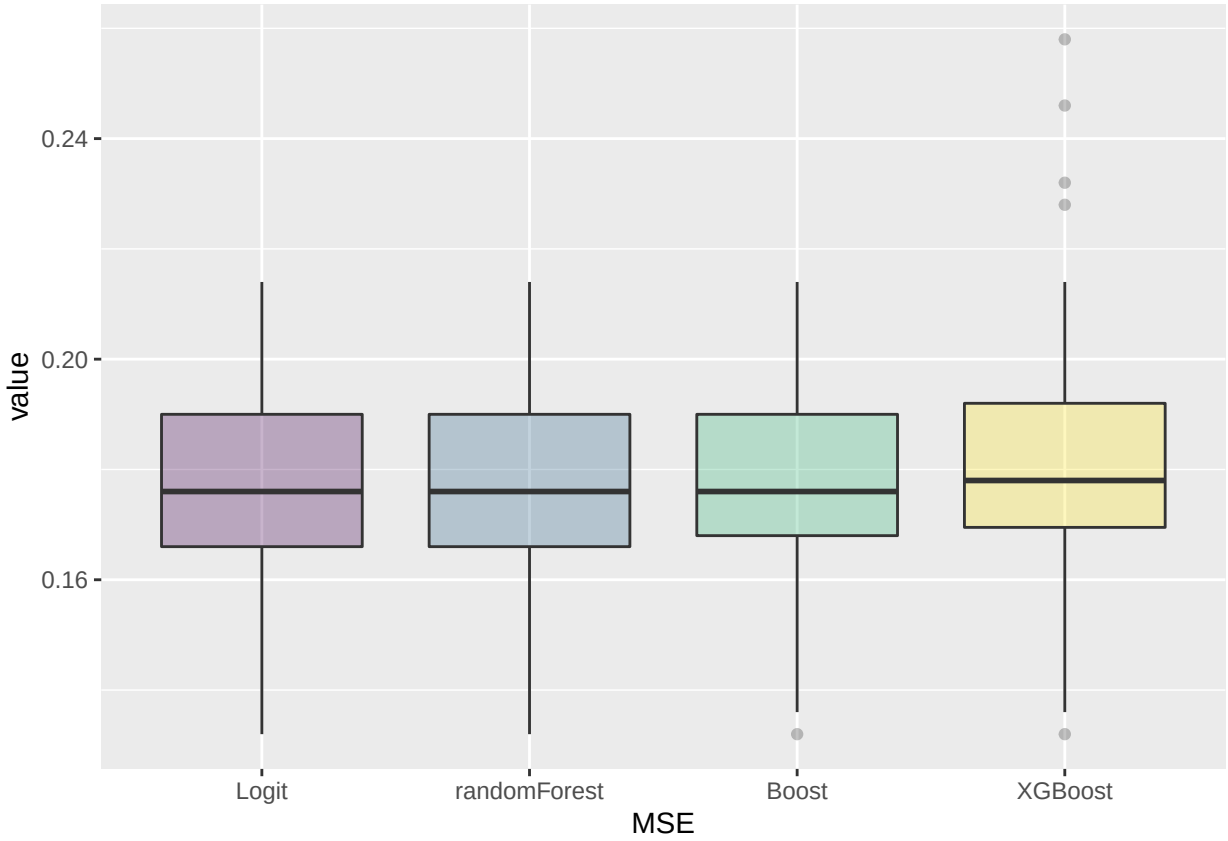
(5);

$$z \sim N_J(0, 1)$$

Where $\beta = (\beta_0, \beta_1) = (-0.9, 0.75)$ are the parameters of interest. We also let $\sigma_u = 1$,$\sigma_v = 1$ and $J = 100$ for all our simulations. The correlation structure in (4), leads to $E(x.u) \neq 0$ implying an endogeneity issue. The sample size n is 1000; I consider a value for σ_{uv}: 0.3. For $H(.)$, we assume $H(z) = [1, 1, ..., 1, 1]z$ with the 1s representing J week instruments. We assume $\alpha = (\alpha_0, \alpha_1) = (0.3, 0.03)$.

Table 5: Test MSE in the first-stage prediction of a 2SLS

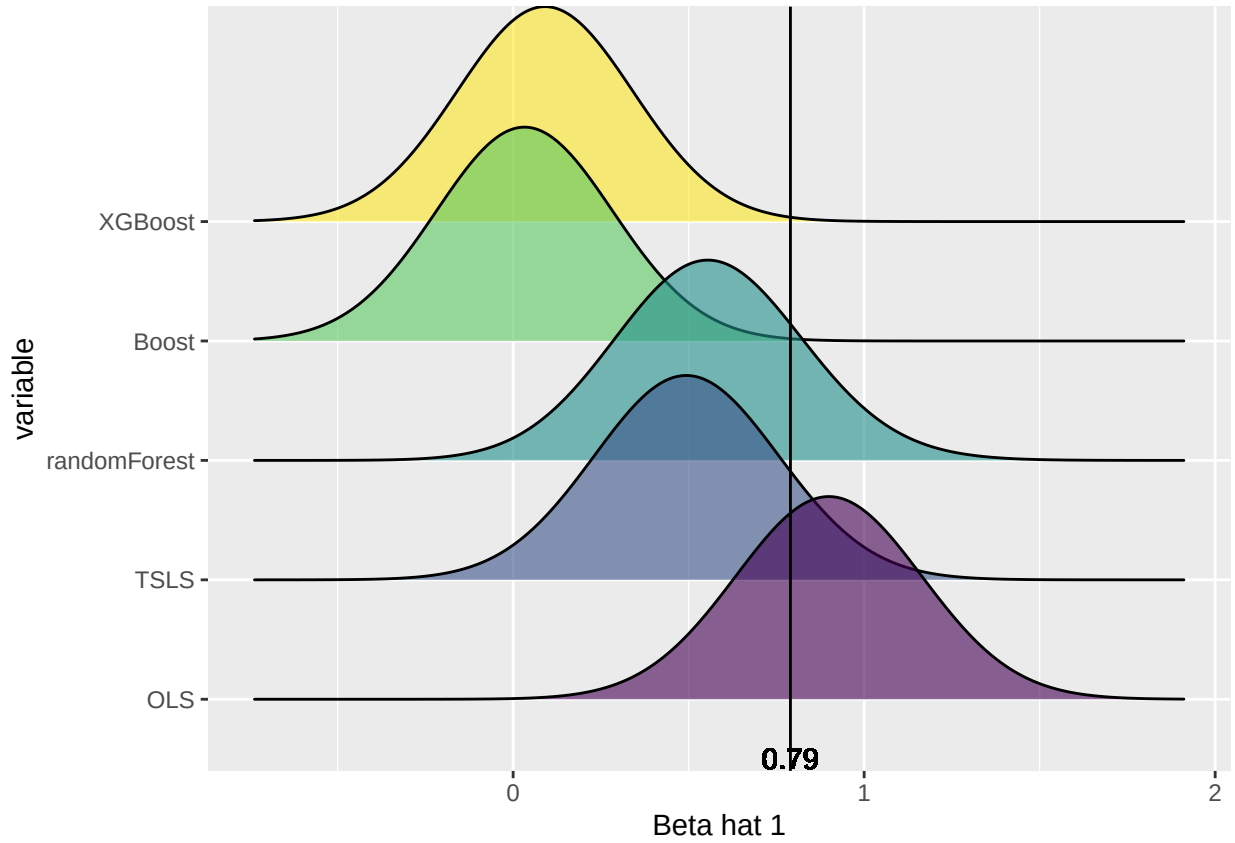
	Train	Test
Logit	0.01004	0.17652
randomForest	0.01040	0.17646
regression_forest	0.01104	0.49664
Boost	0.00794	0.17740
XGBoost	0.00000	0.18038




```
## Warning: Ignoring unknown parameters: text
```

Table 6: Beta hat 1 distribution across competing two-stage methods

	Coefficient	Std. dev.	Bias
OLS	0.8988992	0.0892023	-0.1488992
TSLS train	0.8794602	0.0910955	-0.1294602
TSLS test	0.4953488	0.0958931	0.2546512
randomForest tr	0.9912731	0.1036937	-0.2412731
randomForest ts	0.5614028	0.1090721	0.1885972
regression_forest tr	-0.2352932	0.0134273	0.1119802
regression_forest ts	0.7365727	0.0558263	0.0059060
Boost train	0.6941737	0.0326331	0.0065060
Boost test	0.7173669	0.9110947	0.0907501
XGBoost train	NA	0.4929253	0.0970325
XGBoost test	NA	0.8988992	0.0892023



Case 4:

Carneiro, Heckman, and Vytlacil (2011) uses multiple instrumental variables to estimate the return to college in a sample of white men from the National Longitudinal Survey of Youth (NLSY) 1979 Cohort. We use their sample.

An extensive literature argues that OLS estimates of the return to schooling may be biased by unobserved differences in ability (e.g., Card 2001). Motivated by this concern, consider a two-stage least squares estimates of the system describing schooling (S_i) and log wages (Y_i) for individual i.

We begin with a first stage equation predicting college attendance, S_i. S_i equals one for individuals that attended college. This equation include a vector of controls X_i and instruments Z_i. The instruments Z are (i) the presence of a four-year college in the county of residence at age 14 as a measure of distance to college, (ii) local wage in the county of residence at age 17, (iii) local unemployment in the state of residence at age 17, and (iv) average tuition in public four-year colleges in the county of residence at age 17. In the selection equation we include as well the interactions of the four instruments with AFQT, maternal education, and number of siblings.

$$(7)\ S_i = 1[\psi + Z_i\pi + X_i\delta + v_i > 0]$$

Our second stage estimates are given by:

$$(8)\ Y_i = \alpha + \beta\hat{S}_i + X_i\gamma + u_i$$

The outcome variable Y_i is the log of individuals i's average hourly wage from 1989 to 1993. $\hat{S}_i$ is the prediction of college attendance. Throughout our analysis we control for a vector X_i of covariates that includes AFQT (Armed Forces Qualification Test) scores, mother's years of education, number of siblings,

urban residence at age 14, year of birth indicators, permanent local earnings (defined as average log earnings in the age 17 county of residence from 1973 to 2000), average unemployment in state of residence at 17 and average log earnings in the county of residence in 1991.

The Table below reports descriptive statistics for key variables in the Carneiro, Heckman, and Vytlacil (2011).

```
# Load data
local_level_dta <- haven::read_dta(file.path(data_dir,'localvariables.dta'))
basic_level_dta <- haven::read_dta(file.path(data_dir,'basicvariables.dta'))

card_data <- cbind.data.frame(basic_level_dta,local_level_dta)

stargazer(card_data, header=FALSE, type='latex',digits = 2)
```

Table 7:

Statistic	N	Mean	St. Dev.	Min	Max
caseid	1,747	3,330.38	2,541.55	6	12,139
urban14	1,747	0.74	0.44	0	1
numsibs	1,747	2.93	1.91	0	15
mhgc	1,747	12.10	2.33	0	20
school	1,747	2.67	0.97	1	4
d57	1,747	0.10	0.30	0	1
d58	1,747	0.10	0.30	0	1
d59	1,747	0.10	0.31	0	1
d60	1,747	0.13	0.34	0	1
d61	1,747	0.13	0.34	0	1
d62	1,747	0.17	0.37	0	1
d63	1,747	0.14	0.35	0	1
cafqt	1,747	0.45	0.95	−2.66	2.73
state	1,747	0.50	0.50	0	1
wage	1,747	2.38	0.50	0.06	4.31
const	1,747	1.00	0.00	1	1
exp	1,747	8.49	3.58	0.00	16.60
expsq	1,747	84.87	61.39	0.00	275.43
pub4	1,747	0.52	0.50	0	1
avurate	1,747	6.25	0.99	3.47	8.72
lavlocwage17	1,747	10.28	0.19	9.71	10.85
tuit4c	1,747	21.57	7.98	0.00	67.43
lwage5	1,747	10.29	0.17	9.93	10.69
lurate	1,747	6.81	1.27	2.60	10.50
lwage5_17	1,747	10.28	0.16	9.93	11.18
lurate_17	1,747	7.08	1.81	2.80	12.50
newid	1,747	7,095.07	3,223.85	1,282	12,684

In this section the results of the simulations

The baseline DGP where the individual observations $i = 1, ..., n$ are independently drawn from is assumed as follows:

The error terms (u, v) are chosen to jointly follow a bivariate normal distribution:

(4);

$$\begin{pmatrix} u \\ v \end{pmatrix} = N(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma_u & \sigma_{uv} \\ \sigma_{vu} & \sigma_v \end{bmatrix})$$

Without loss of generality, the exogenous variables x and z are normally distributed with mean and standard deviation corresponding to the variables in the original data set. The binary endogenous explanatory variable (S) is generated such that the shares of zeros and ones is close to 50%.

$$(7)\ S_i^* = \psi + Z_i\pi + X_i\delta + v_i$$

$$(7)\ S_i = 1[S_i^* > mean(S_i^*)]$$

The dependent variable is generated as:

$$(8)\ Y_i = \alpha + \beta S_i + X_i\gamma + u_i$$

```
dgp_card <- function(n_obs,basic_level_data,local_level_data,alpha_1,alpha_2,beta,
                     mean_eps,sigma_eps,interaction="FALSE") {
        # basic variables
        x0 <- rep(1,n_obs)
        x1_cafqt <- rnorm(n_obs,mean(basic_level_dta$cafqt),sd(basic_level_dta$cafqt))
        x2_cafqtsq <- x1_cafqt^2
        x3_msch  <- rnorm(n_obs,mean(basic_level_dta$mhgc),sd(basic_level_dta$mhgc))
        x4_mschsq <- x3_msch^2
        x5_numsibs <- rnorm(n_obs,mean(basic_level_dta$numsibs),sd(basic_level_dta$numsibs))
        x6_numsibs_sq <- x5_numsibs^2
        x7_urban14 <- rbinom(n_obs,1,mean(basic_level_dta$urban14))
        prob.x_d <- as.vector(colMeans(basic_level_dta[,c(6:12)]))
        x8_d  <- sample(c(57:63), n_obs, replace=TRUE, prob=prob.x_d)
        x8_year <- model.matrix(~ factor(x8_d) + 0)
        x9_exp <- rnorm(n_obs,mean(basic_level_dta$exp),sd(basic_level_dta$exp))
        x10_expsq <- x9_exp^2
        # local variables
        x11_avlocwage17 <- rnorm(n_obs,mean(local_level_dta$lavlocwage17),sd(local_level_dta$lavlocwage
        x12_avlocwage17_sq <- x11_avlocwage17^2
        x13_avurate <- rnorm(n_obs,mean(local_level_dta$avurate),sd(local_level_dta$avurate))
        x14_avurate_sq <- x13_avurate^2
        x15_lwage_1991 <- rnorm(n_obs,mean(local_level_dta$lwage5),sd(local_level_dta$lwage5))
        x16_lurate_1991 <- rnorm(n_obs,mean(local_level_dta$lurate),sd(local_level_dta$lurate))
        # Instruments
        z1_pub4 <- rbinom(n_obs,1,mean(local_level_dta$pub4)) # college proximity
        z2_lwage5_17 <- rnorm(n_obs,mean(local_level_dta$lwage5_17),sd(local_level_dta$lwage5_17))
        z3_lurate_17 <- rnorm(n_obs,mean(local_level_dta$lurate_17),sd(local_level_dta$lurate_17))
        z4_tuit4c <- rnorm(n_obs,mean(local_level_dta$tuit4c),sd(local_level_dta$tuit4c))
        # Defining x and y
        sigma_ev <- matrix(c(1,sigma_eps,sigma_eps,1),2)
        u <- mvrnorm(n_obs,mean_eps,sigma_ev)
        x.2 <- cbind(x0,x1_cafqt,x2_cafqtsq,x3_msch,x4_mschsq,x5_numsibs,x6_numsibs_sq,
                     x7_urban14,x8_year[,-1],x11_avlocwage17,x12_avlocwage17_sq,
                     x13_avurate,x14_avurate_sq,x9_exp,x10_expsq,x15_lwage_1991,x16_lurate_1991)
        if (interaction==FALSE & length(alpha_1)==5){
                z.1 <- cbind(rep(1,n_obs),z1_pub4,z2_lwage5_17,z3_lurate_17,z4_tuit4c)
                mean.con.x1 <- mean(z.1%*%alpha_1 + x.2[,2:22]%*%alpha_2 + u[,1])
                x.1 <- (z.1%*%alpha_1 + x.2[,2:22]%*%alpha_2 + u[,1] > mean.con.x1)*1
        } else if (interaction==TRUE & length(alpha_1)==17){
                x.inter <- cbind(x1_cafqt,x3_msch,x5_numsibs)
                z.1 <- cbind(rep(1,n_obs),z1_pub4,z2_lwage5_17,z3_lurate_17,z4_tuit4c,
                             z1_pub4*x.inter,z2_lwage5_17*x.inter,z3_lurate_17*x.inter,z4_tuit4c*x.inte
```

```
                mean.con.x1 <- mean(z.1%*%alpha_1 + x.2[,2:22]%*%alpha_2 + u[,1])
                x.1 <- (z.1%*%alpha_1 + x.2[,2:22]%*%alpha_2 + u[,1] > mean.con.x1)*1
        }

        y = beta[1]*x.2[,1] + beta[2]*x.1 + x.2[,2:22]%*%beta[3:23] + u[,2]

        data <- data.frame("y"=y,"x.1"=x.1,x.2[,-1],z.1[,-1])
        return(data)
}
```

```
prediction_first_stage_iv_card  <- function(n_simulations,n_observations,
                                             basic_level_data,local_level_data,
                                             alpha_1,alpha_2,beta,mean_eps,sigma_eps,
                                             interaction="FALSE"){
        mse_logit_card <- matrix(NA,n_simulations,2)
        mse_rf_card    <- matrix(NA,n_simulations,2)
        mse_boost_card <- matrix(NA,n_simulations,2)
        mse_xgboost_card <- matrix(NA,n_simulations,2)

        beta_1_ols_card   <- matrix(NA,n_simulations,4)
        beta_1_tsls_card  <- matrix(NA,n_simulations,4)
        beta_1_rf_card    <- matrix(NA,n_simulations,4)
        beta_1_boost_card <- matrix(NA,n_simulations,4)
        beta_1_xgboost_card <- matrix(NA,n_simulations,4)

        bias_beta_1_ols_card   <- rep(NA,n_simulations)
        bias_beta_1_tsls_card  <- matrix(NA,n_simulations,2)
        bias_beta_1_rf_card    <- matrix(NA,n_simulations,2)
        bias_beta_1_boost_card <- matrix(NA,n_simulations,2)
        bias_beta_1_xgboost_card <- matrix(NA,n_simulations,2)

        for (i in 1:n_simulations){
                train_data <- dgp_card(n_observations,basic_level_data,local_level_data,
                                       alpha_1,alpha_2,beta,mean_eps,
                                       sigma_eps,interaction="FALSE")
                test_data <- dgp_card(n_observations,basic_level_data,local_level_data,
                                      alpha_1,alpha_2,beta,mean_eps,
                                      sigma_eps,interaction="FALSE")

                # OLS model
                ols <- lm(y ~ .,data=test_data[,1:23])
                beta_1_ols_card[i,1] = coefficients(ols)[2]
                beta_1_ols_card[i,2] = sqrt(diag(vcov(ols)))[2]
                bias_beta_1_ols_card[i] = beta[2]- beta_1_ols_card[i,1]

                # Estimation of the first stage with logit
                logit <- glm(x.1 ~ . -y, family = "binomial", data = train_data)
                train_data$pred_logit_x <- predict(logit,type ="response")
                test_data$pred_logit_x  <- predict(logit,newdata=test_data,type ="response")
                x_hat_logit_tr <- ifelse(train_data$pred_logit_x  > 0.5,1,0)
                x_hat_logit_ts <- ifelse(test_data$pred_logit_x > 0.5,1,0)
                mse_logit_card[i,1] = mean(train_data$x.1 != x_hat_logit_tr)
                mse_logit_card[i,2] = mean(test_data$x.1 != x_hat_logit_ts)
                # Estimation of the second stage with OLS
```

```
if (interaction == TRUE){
        tsls_train <- lm(y ~ . - x.1,data=train_data[,-c(24:39)])
        tsls_test  <- lm(y ~ . - x.1,data=test_data[,-c(24:39)])
} else if (interaction == FALSE){
        tsls_train <- lm(y ~ . - x.1,data=train_data[,-c(24:27)])
        tsls_test  <- lm(y ~ . - x.1,data=test_data[,-c(24:27)])
}
beta_1_tsls_card[i,1] = coefficients(tsls_train)[23]
beta_1_tsls_card[i,2] = sqrt(diag(vcov(tsls_train)))[23]
beta_1_tsls_card[i,3] = coefficients(tsls_test)[23]
beta_1_tsls_card[i,4] = sqrt(diag(vcov(tsls_test)))[23]
bias_beta_1_tsls_card[i,1] = beta[2]- beta_1_tsls_card[i,1]
bias_beta_1_tsls_card[i,2] = beta[2]- beta_1_tsls_card[i,3]

# Estimation of the first stage with randomForest
rf <- randomForest(x.1 ~ .-y,data=train_data,ntrees=100)
train_data$predict_rf_x <- predict(rf,type="class",ntrees=100) # Predict on out-of-bag
x_hat_rf_tr <- ifelse(train_data$predict_rf_x > 0.5,1,0)
test_data$predict_rf_x <- predict(rf,type="class",newdata = test_data,ntrees=100)
x_hat_rf_ts <- ifelse(test_data$predict_rf_x > 0.5,1,0)
mse_rf_card[i,1] = mean(train_data$x.1 != x_hat_rf_tr)
mse_rf_card[i,2] = mean(test_data$x.1  != x_hat_rf_ts)
# Estimation of the second stage with OLS
if (interaction == TRUE){
        tsls_rf_train <- lm(y ~ . - x.1,data=train_data[,-c(24:40)])
        tsls_rf_test  <- lm(y ~ . - x.1,data=test_data[,-c(24:40)])
} else if (interaction == FALSE){
        tsls_rf_train <- lm(y ~ . - x.1,data=train_data[,-c(24:28)])
        tsls_rf_test  <- lm(y ~ . - x.1,data=test_data[,-c(24:28)])
}
beta_1_rf_card[i,1] = coefficients(tsls_rf_train)[23]
beta_1_rf_card[i,2] = sqrt(diag(vcov(tsls_rf_train)))[23]
beta_1_rf_card[i,3] = coefficients(tsls_rf_test)[23]
beta_1_rf_card[i,4] = sqrt(diag(vcov(tsls_rf_test)))[23]
bias_beta_1_rf_card[i,1] = beta[2]- beta_1_rf_card[i,1]
bias_beta_1_rf_card[i,2] = beta[2]- beta_1_rf_card[i,3]

# Estimation of the first stage with boost
boost <- gbm(x.1 ~ .-y,data=train_data,distribution="bernoulli",n.trees=100)
train_data$predict_boost_x <- predict(boost,n.trees=100) # Predict on out-of-bag traini
x_hat_boost_tr <- ifelse(train_data$predict_boost_x > 0.5,1,0)
test_data$predict_boost_x <- predict(boost,newdata = test_data,n.trees=100)
x_hat_boost_ts <- ifelse(test_data$predict_boost_x > 0.5,1,0)
mse_boost_card[i,1] = mean(train_data$x.1 != x_hat_boost_tr)
mse_boost_card[i,2] = mean(test_data$x.1 != x_hat_boost_ts)
# Estimation of the second stage with OLS
if (interaction == TRUE){
        tsls_boost_train <- lm(y ~ . - x.1,data=train_data[,-c(24:41)])
        tsls_boost_test  <- lm(y ~ . - x.1,data=test_data[,-c(24:41)])
} else if (interaction == FALSE){
        tsls_boost_train <- lm(y ~ . - x.1,data=train_data[,-c(24:29)])
        tsls_boost_test  <- lm(y ~ . - x.1,data=test_data[,-c(24:29)])
}
```

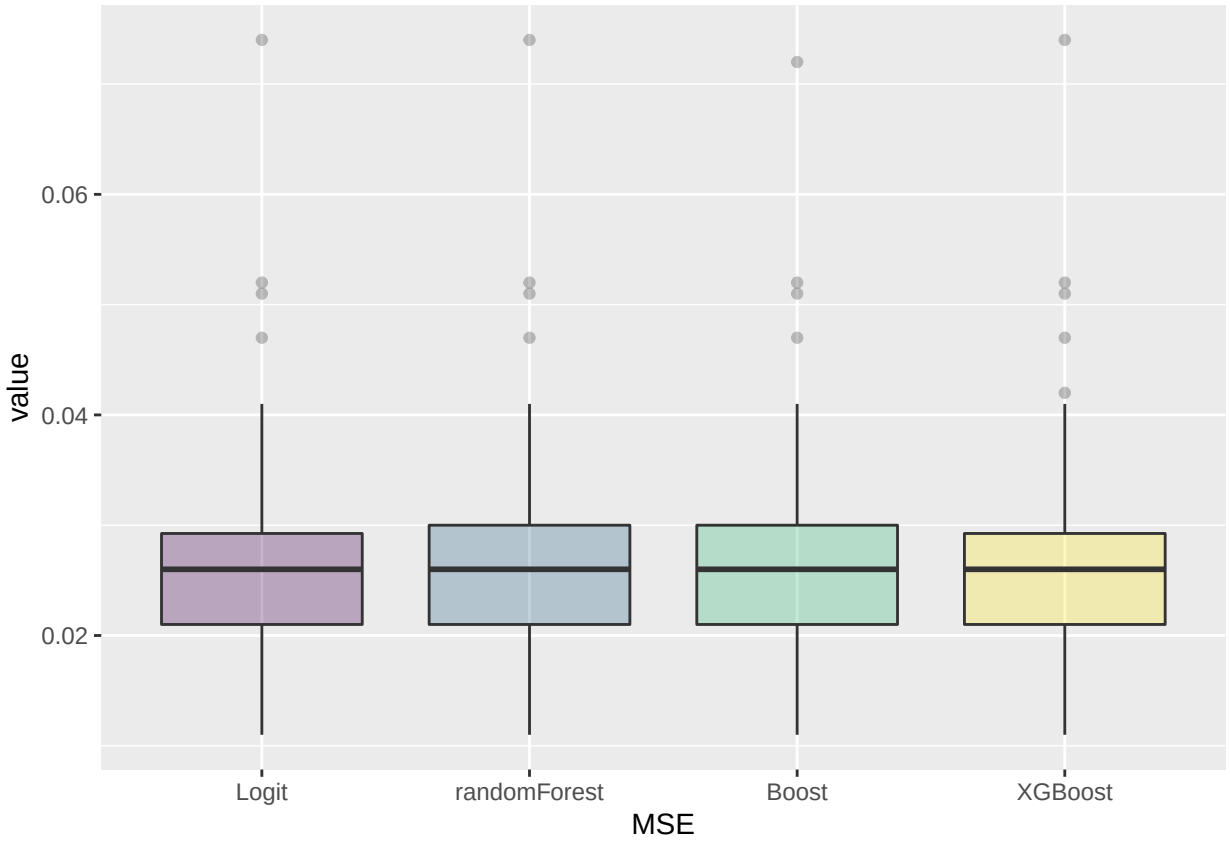
```
                beta_1_boost_card[i,1] = coefficients(tsls_boost_train)[23]
                beta_1_boost_card[i,2] = sqrt(diag(vcov(tsls_boost_train)))[23]
                beta_1_boost_card[i,3] = coefficients(tsls_boost_test)[23]
                beta_1_boost_card[i,4] = sqrt(diag(vcov(tsls_boost_test)))[23]
                bias_beta_1_boost_card[i,1] = beta[2]- beta_1_boost_card[i,1]
                bias_beta_1_boost_card[i,2] = beta[2]- beta_1_boost_card[i,3]

                # Estimation of the first stage with XGBoost
                train_data_mod <- as.matrix(train_data)
                test_data_mod <- as.matrix(test_data)
                train_x <- train_data_mod[,2]
                train_z <- train_data_mod[,-c(1:2)]
                xgb_train <- xgb.DMatrix(data = train_z,label = train_x)
                test_x <- test_data_mod[,2]
                test_z <- test_data_mod[,-c(1:2)]
                xgb_test <- xgb.DMatrix(data = test_z, label = test_x)
                xg_boost <- xgboost(data=xgb_train,max.depth=3,eta=0.3,nrounds=60,
                            objective = "binary:logistic",verbose=FALSE)
                train_data$predict_xg_boost_x <- predict(xg_boost,xgb_train)
                x_hat.tr <- ifelse(train_data$predict_xg_boost_x > 0.5,1,0)
                test_data$predict_xg_boost_x <- predict(xg_boost,newdata=xgb_test)
                x_hat.ts <- ifelse(test_data$predict_xg_boost_x > 0.5,1,0)
                mse_xgboost_card[i,1] = mean(train_data$x.1 != x_hat.tr) #mse of xgboost in train data
                mse_xgboost_card[i,2] = mean(test_data$x.1 != x_hat.ts) #mse of xgboost in test data
                # Estimation of the second stage with OLS
                if (interaction == TRUE){
                        tsls_xgboost_train <- lm(y ~ . - x.1,data=train_data[,-c(24:42)])
                        tsls_xgboost_test  <- lm(y ~ . - x.1,data=test_data[,-c(24:42)])
                } else if (interaction == FALSE){
                        tsls_xgboost_train <- lm(y ~ . - x.1,data=train_data[,-c(24:30)])
                        tsls_xgboost_test  <- lm(y ~ . - x.1,data=test_data[,-c(24:30)])
                }
                beta_1_xgboost_card[i,1] = coefficients(tsls_xgboost_train)[23]
                beta_1_xgboost_card[i,2] = sqrt(diag(vcov(tsls_xgboost_train)))[23]
                beta_1_xgboost_card[i,3] = coefficients(tsls_xgboost_test)[23]
                beta_1_xgboost_card[i,4] = sqrt(diag(vcov(tsls_xgboost_test)))[23]
                bias_beta_1_xgboost_card[i,1] = beta[2]- beta_1_xgboost_card[i,1]
                bias_beta_1_xgboost_card[i,2] = beta[2]- beta_1_xgboost_card[i,3]
        }
        return(list(mse_logit=mse_logit_card,mse_rf=mse_rf_card,mse_boost=mse_boost_card,
                    mse_xgboost=mse_xgboost_card,beta_1_ols=beta_1_ols_card,
                    beta_1_tsls=beta_1_tsls_card,beta_1_rf=beta_1_rf_card,
                    beta_1_boost=beta_1_boost_card,beta_1_xgboost=beta_1_xgboost_card,
                    bias_beta_1_ols=bias_beta_1_ols_card,
                    bias_beta_1_tsls=bias_beta_1_tsls_card,bias_beta_1_rf=bias_beta_1_rf_card,
                    bias_beta_1_boost=bias_beta_1_boost_card,
                    bias_beta_1_xgboost=bias_beta_1_xgboost_card))
}
```

Table 8: Test MSE in the first-stage prediction of a 2SLS

	Train	Test
Logit	0.00286	0.02688

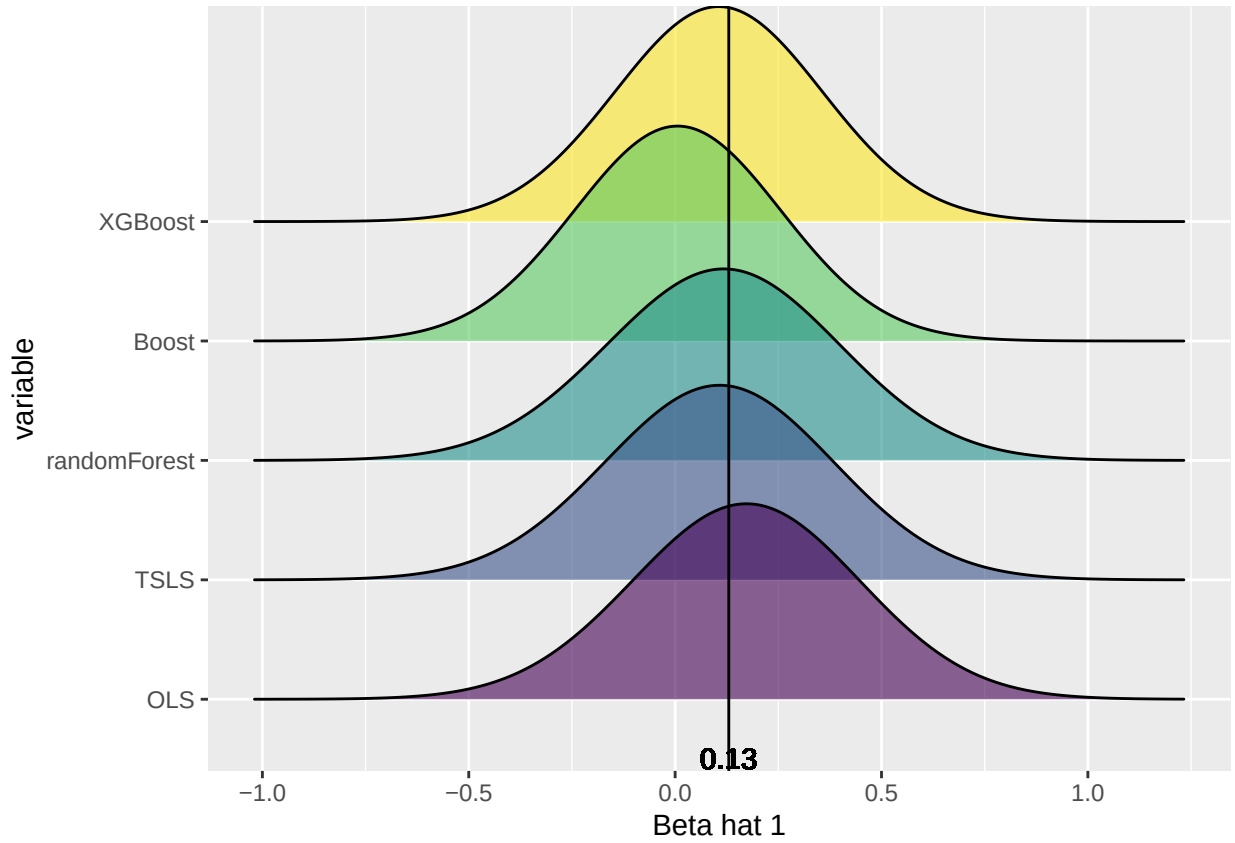
	Train	Test
RF	0.00315	0.02688
Boost	0.00187	0.02690
XGBoost	0.00003	0.02700




```
## Warning: Ignoring unknown parameters: text
```

Table 9: Beta hat 1 distribution across competing two-stage methods

	Coefficient	Std. dev.	Bias
OLS	0.1704453	0.1057750	-0.0404453
TSLS train	0.1684623	0.1059907	-0.0384623
TSLS test	0.1059147	0.1067482	0.0240853
RF train	0.1788420	0.1137713	-0.0488420
RF test	0.1143509	0.1145947	0.0156491
Boost train	0.0104832	0.0063958	0.1195168
Boost test	0.0060284	0.0064235	0.1239716
XGBoost train	0.1797366	0.1055446	-0.0497366
XGBoost test	0.1041419	0.1064620	0.0258581



The results of this last case shows that compared to the previous scenarios, tree-based methods and logit regression yield similar MSEs in the firs-stage prediction. Regarding the second-stage, 2SLS, naive OLS, random forest and XGBoost give almost consistent and unbiased estimators of $\hat{\beta}_1$, being the exception the boosting forest.

References

Angrist, Joshua D, and Brigham Frandsen. 2022. "Machine Labor." *Journal of Labor Economics* 40 (S1): S97–140. https://doi.org/10.1086/717933.

Belloni, Alexandre, Daniel Chen, Victor Chernozhukov, and Christian Hansen. 2012. "Sparse Models and Methods for Optimal Instruments with an Application to Eminent Domain." *Econometrica* 80 (6): 2369–429. https://doi.org/10.3982/ECTA9626.

Breiman, Leo. 2001. "Random Forests." *Machine Learning* 45 (1): 5–32. https://doi.org/10.1023/A:1010933404324.

Carneiro, Pedro, James J Heckman, and Edward J Vytlacil. 2011. "Estimating Marginal Returns to Education." *American Economic Review* 101 (6): 2754–81. https://www.aeaweb.org/articles?id=10.1257/aer.101.6.2754.

Friedman, Jerome H. 2001. "Greedy Function Approximation: A Gradient Boosting Machine." *Annals of Statistics*, 1189–1232. https://www.jstor.org/stable/2699986.

Hausman, Jerry A. 1975. "An Instrumental Variable Approach to Full Information Estimators for Linear and Certain Nonlinear Econometric Models." *Econometrica: Journal of the Econometric Society*, 727–38. https://doi.org/10.2307/1913081.

Lennon, Connor, Edward Rubin, and Glen R Waddell. 2021. "What Can We Machine Learn (Too Much of) in 2sls? Insights from a Bias Decomposition and Simulation." Working Paper. http://edrub.in/Papers/draft-mliv.pdf.

Mogstad, Magne, Alexander Torgovitsky, and Christopher R Walters. 2021. "The Causal Interpretation of

Two-Stage Least Squares with Multiple Instrumental Variables." *American Economic Review* 111 (11): 3663–98. https://www.aeaweb.org/articles?id=10.1257/aer.20190221.

Mullainathan, Sendhil, and Jann Spiess. 2017. "Machine Learning: An Applied Econometric Approach." *Journal of Economic Perspectives* 31 (2): 87–106. https://www.aeaweb.org/articles?id=10.1257/jep.31.2.87.