Variable Selection Methods Comparison

Hongjie Liu, Jiajun Tao, Shaohan Chen

2023-02-27

Motivation

- Identifying the effect of a treatment, exposure, or intervention is one of the most fundamental tasks we encounter as biostatisticians. . .
- . . . but outside of a randomized control trial (RCT), confounding variables can bias our estimates of treatment effects.
- Propensity score matching (PSM) is a tool that can help us mitigate the effects of confounders. . .
- . . . but there is no consensus on the best way to estimate standard errors when using the PSM algorithm.
- How can we assess which procedures reliably estimate standard errors?

Motivation

A simulation study!

A (Yet) Quick(er) Foray into Propensity Score Matching

(1) We start with an unmatched dataset.
(2) We estimate the propensity score - the probability of treatment given some set of covariates - according to some pre-specified model fitting (e.g., logistic regression).
(3) We pair treated and untreated indiviuals who have similar propensity scores according to some pre-specified matching algorithm (e.g., nearest neighbors).
(4) We end with a matched dataset.

Enter the Bootstrap

- Bootstrapping is one of the most common procedures for estimating standard errors.
- The PSM algorithm intakes an unmatched dataset and outputs a matched one.
- **Primary Research Question:** When do we execute the bootstrap - before the match or after it?
- Let's try both!

Data Generation - Continuous Outcome

For each individual $i \in \{1, \ldots, n\}$, we consider covariates $L_{1i}, L_{2i}, L_{3i} \sim N(0, 1)$. Treatments are distributed according to law $A_i \sim B(\pi_i)$, where π_i - the true propensity to be treated - is subject to the data-generating process

$$\log\left(\frac{\pi_i}{1-\pi_i}\right) = \alpha_0 + \alpha_1 L_{1i} + \alpha_2 L_{2i}.$$

Given this, we further define the data-generating process of our continuous outcome via

$$Y_i = \beta_1 A_i + \beta_2 L_{2i} + \beta_3 L_{3i} + \varepsilon_i,$$

where ε_i denotes random error. Because L_{2i} effects both A_i and Y_i, it acts as a confounder in estimating the treatment effect.

Data Generation - Binary Outcome

For each individual $i \in \{1, \ldots, n\}$, we consider covariates $L_{1i}, L_{2i}, L_{3i} \sim N(0,1)$. Treatments are distributed according to law $A_i \sim B(\pi_i)$, where π_i - the true propensity to be treated - is subject to the data-generating process

$$\log\left(\frac{\pi_i}{1-\pi_i}\right) = \alpha_0 + \alpha_1 L_{1i} + \alpha_2 L_{2i}.$$

Given this, we further define the data-generating process of our binary outcome via $Y_i \sim B(\tau_i)$ where

$$\log\left(\frac{\tau_i}{1-\tau_i}\right) = \beta_0 + \beta_1 A_i + \beta_2 L_{2i} + \beta_3 L_{3i}.$$

Observe that we have omitted a random error term, as realizations of our binary Y_i are innately subject to noise.

Data Generation - Random Number Generation (Binary Outcome)

```r
set.seed(20220217)
seed_vec <- runif(100000, min, max)
  for (i in 1:n) {
    set.seed(seeds[i])
    long_rnorm <- rnorm(size*3, mean = 0, sd = 1)
    long_runif <- runif(size*2)
    beta_error <- rnorm(size, mean = 0, sd = 0.25)
    L1 <- long_rnorm[1:size]
    L2 <- long_rnorm[(size + 1):(2*size)]
    L3 <- long_rnorm[(2*size + 1):(3*size)]

    comp_pA = long_runif[1:size]
    A = (prob_A > comp_pA)
    # function continues...
    }
```

Parameters of Interest

- The sample size of each dataset $n_{\text{sample}} \in \{100, 1000\}$
- The population proportion of treated individuals $\pi \in \{0.113, 0.216, 0.313\}$
- The true average treatment effect $\beta_1 \in \{0.15, 0.30\}$ for binary data; $\beta_1 \in \{-1, 1\}$ for continuous data

Other Parameters

- The number of datasets $m_{\text{sample}} = 100$
- The number of bootstrap re-samples $m_{\text{boot}} = 500$
- The sample size of bootstrap re-samples $n_{\text{simple}} = n_{\text{complex}} = n_{\text{sample}} \times \pi$
- Strength of covariate effect on treatment $\alpha_1 = \log(1.25), \alpha_2 = \log(1.75)$
- Strength of covariate effect on outcome $\beta_2 = \log(1.75), \beta_3 = \log(1.25)$

Evaluation Metrics

Define true predictors as positive and null predictors as negative

Signal Identification

- **Complexity:** Number of Selected Parameters
- **Sensitivity:** $\frac{TP}{TP+FN}$
- **Specificity:** $\frac{TN}{TN+FP}$
- **F1-Score:** $\frac{2 \cdot \text{sensitivity} \cdot \text{precision}}{\text{sensitivity}+\text{precision}}$
- **Accuracy:** $\frac{TP+TN}{TP+TN+FP+FN}$

Parameter Estimation

- **MSE:** $\frac{1}{p}\sum_{i=1}^{p}(\hat{\beta}_i - \beta_i)^2$

Signal Identification Performance - Complexity

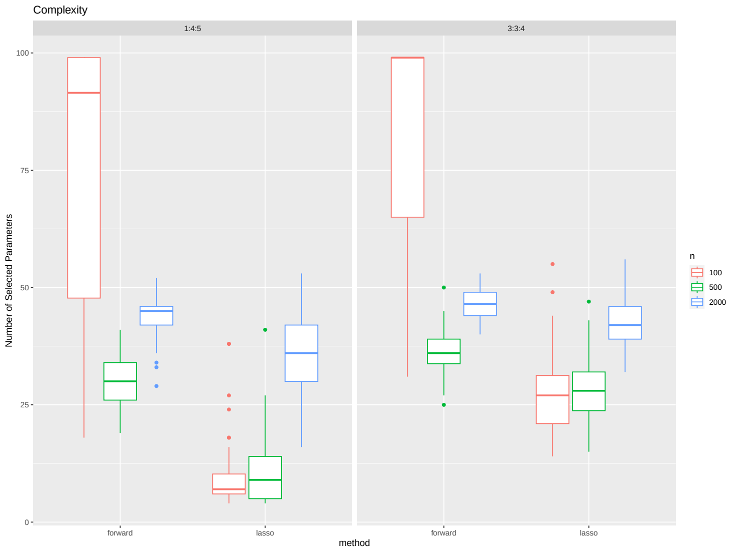

Figure 1: Model Complexity

Signal Identification Performance - Complexity Exploration

- When in high dimensional scenario, forward selection tends to select nearly all of the predictors but Lasso does not
- Lasso tends to select much fewer predictors than forward selection, and the parameters it select will increase as n increases
- As n increases, the predictors that two models select are more precise(closer to true predictor number 40)

Signal Identification Performance - Sensitivity

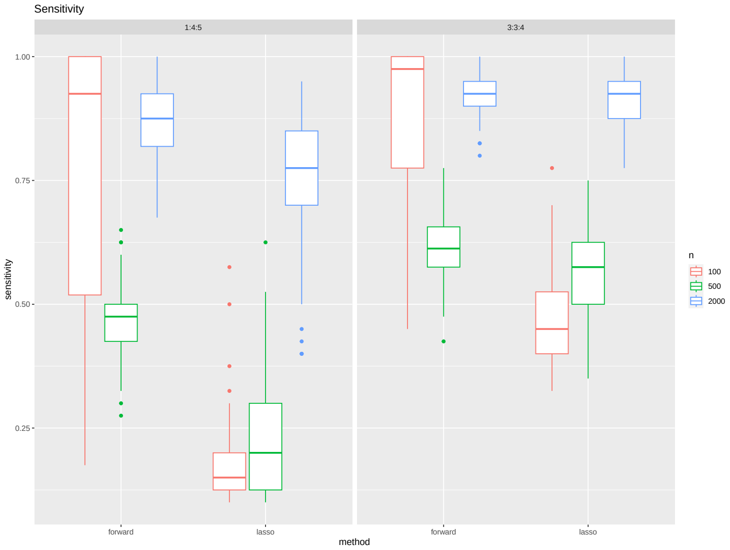

Figure 2: Sensitivity Performance

Signal Identification Performance - Sensitivity

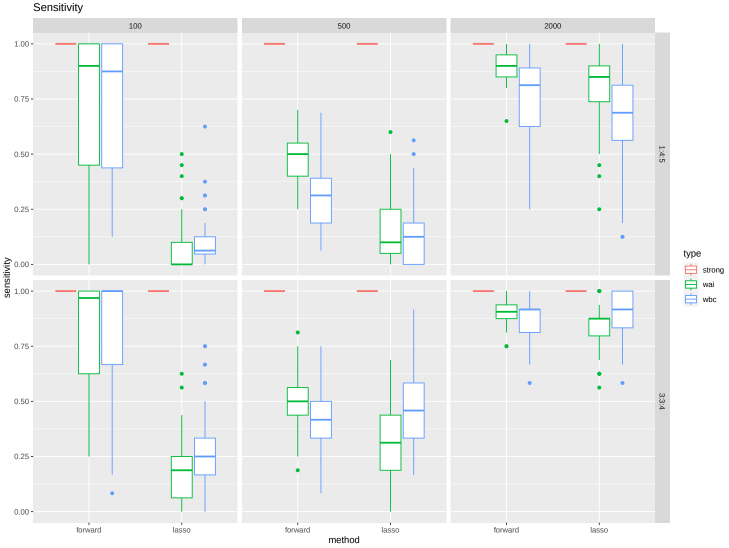

Figure 3: Sensitivity of True Signals

Signal Identification Performance - Sensitivity Exploration

- Forward selection are highly sensitive in high dimensional case(n=100) while Lasso does not
- Overall, the sensitivity of two models increases as n increases
- Both models are sensitive in selecting strong signals. But when it comes to weak predictors, Lasso is much less sensitive in high dimensional scenario than forward selection.
- When n increases, the sensitivity discrepancy of selecting weak predictors between two models becomes smaller. But still, forward selection is overall more sensitive than Lasso
- The sensitivity discrepancy between two models are smaller when the ratio of strong predictors becomes larger

Signal Identification Performance - Specificity

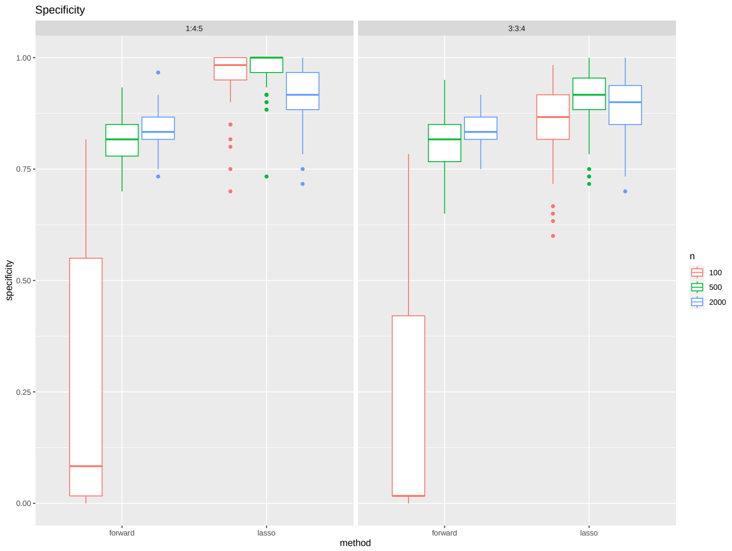

Figure 4: Specificity Performance

Signal Identification Performance - Specificity Exploration

- In high dimensional scenario, the specificity of forward selection is near 0, which means it almost does not identify any null predictor, but Lasso in turn has high specificity
- In high dimensional scenario, forward selection is very assertive and tends to identity all 100 predictors as true, leading to extremely high sensitivity but low specificity
- In high dimensional scenario, Lasso is very conservative and tends to identity most 100 predictors as null,leading to low sensitivity but high specificity
- As n increases, forward selection has higher specificity. And overall, specificity are higher when the ratio of strong predictors are lower

Signal Identification Performance - F1-Score

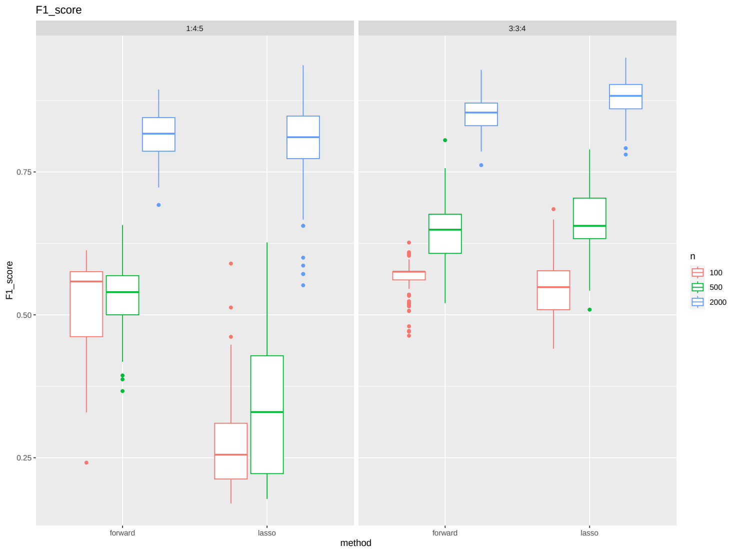

Figure 5: F1-Score Performance

Signal Identification Performance - F1-Score

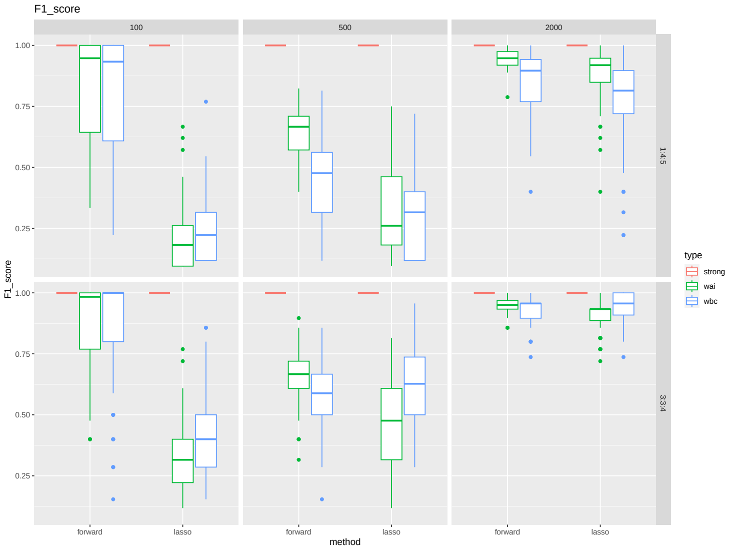

Figure 6: F1-Score of True Signals

Signal Identification Performance - F1-score Exploration

- Lasso has lower F1-Score when n is not large. When n=2000, both models have similarly high F1-score
- F1-score will increase significantly for both models, when the ratio of strong predictors are larger. And F1-score is also higher when the ratio of strong predictors are larger
- Strong predictors have F1-Score=1 for each scenario, and weak predictors have higher F1-score when n increases for Lasso

Signal Identification Performance - Accuracy

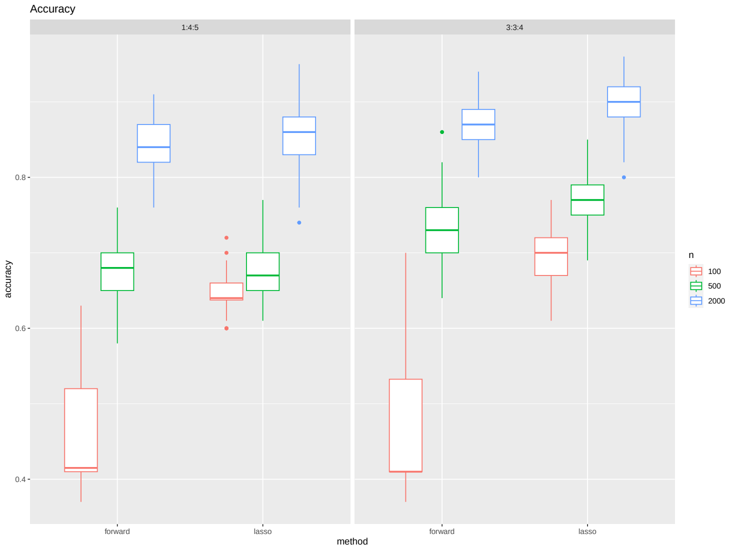

Figure 7: Accuracy Performance

Signal Identification Performance - Accuracy

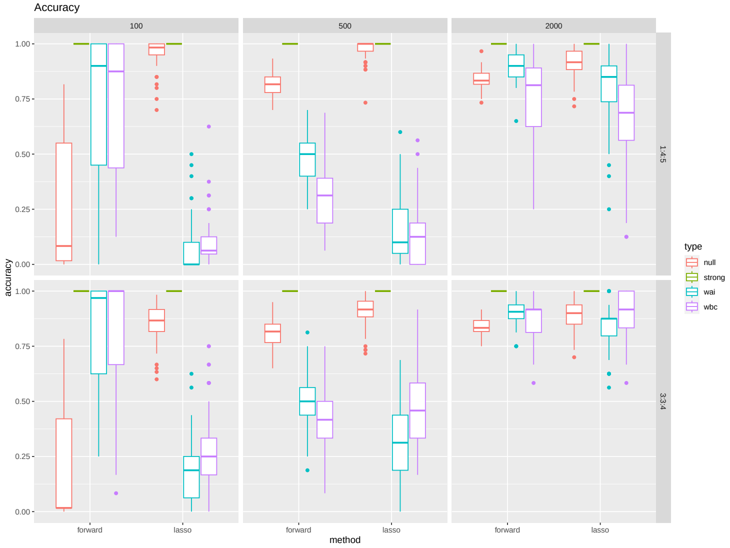

Figure 8: Accuracy of Different Signals

Signal Identification Performance - Accuracy Exploration

- Accuracy is low in high dimensional scenario, especially forward selection
- Accuracy increases for both models when n increase
- Accuracy is higher when the ratio of strong predictors is higher, and Lasso has overall higher accuracy than forward
- In high dimensional, forward has higher accuracy for weak predictors than Lasso

Parameter Estimation Performance - n=100

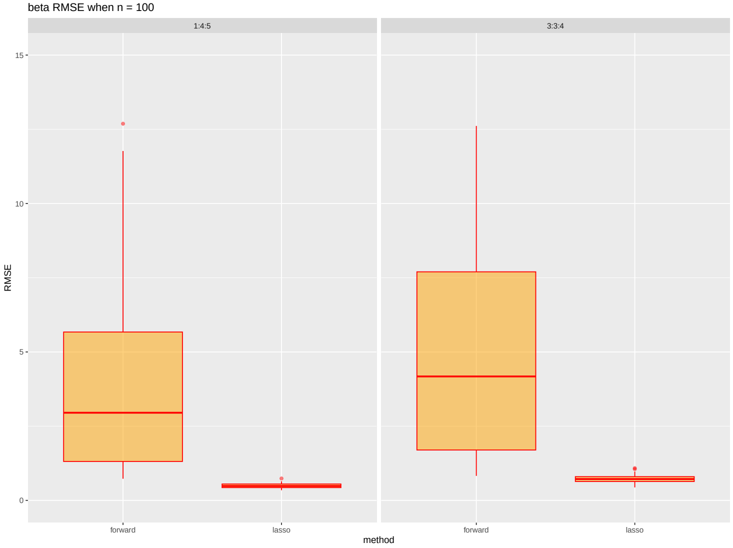

Figure 9: Beta Rmse when n=100

Parameter Estimation Performance - n=500

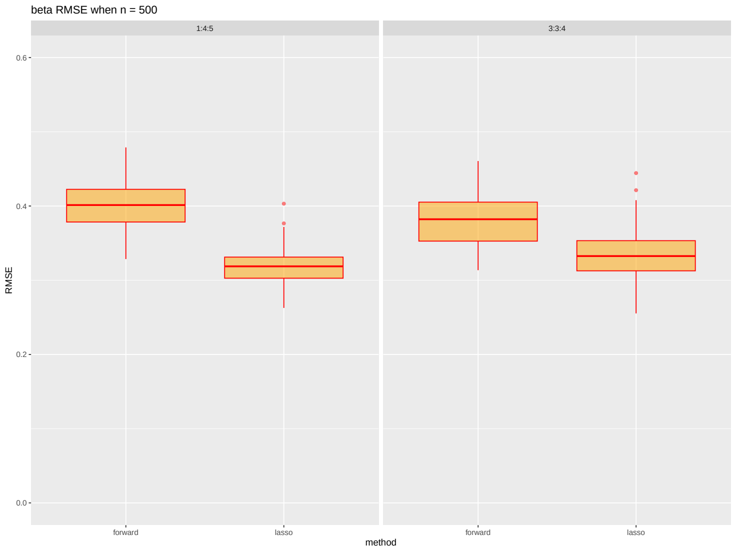

Figure 10: Beta Rmse when n=500

Parameter Estimation Performance - n=2000

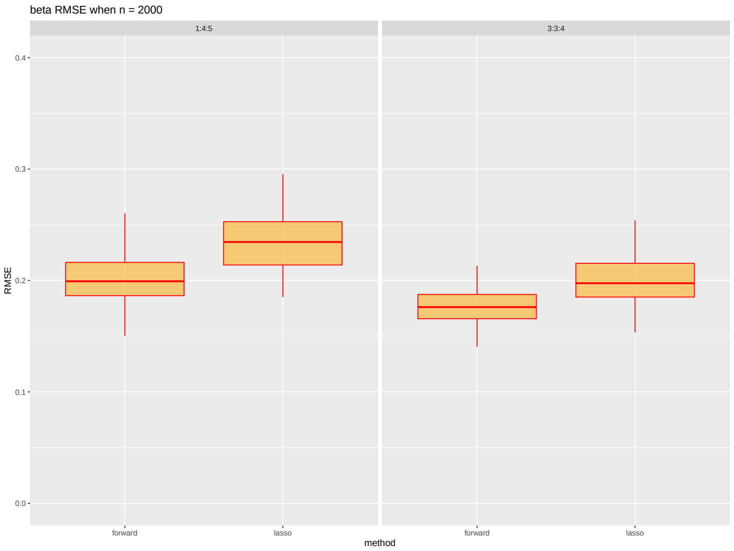

Figure 11: Beta Rmse when n=2000

Parameter Estimation Performance - Exploration

- In high-dimensional scenario, Lasso performs much better than forward
- Lasso performs better than forward when n=500, but worse than forward when n=2000

Predictors Identification Conclusions

High Dimension Scenario - Forward selection tend to be assertive in selecting true predictors while Lasso is conservative, resulting in extremely high sensitivity and more selected predictors for forward selection, and high specificity and less selected predictors for Lasso - In high-dimensional scenarios, forward selection has an overall much higher F1-score, and both methods perfectly identify the strong predictors -

Missing Weak Predictors Analysis - Introduction

How missing "weak" predictors impacts the parameter estimations

Definition: missing weak predictors = true weak predictors but estimated as null

How to value parameter estimations: RMSE

Missing Weak Predictors Analysis - Methods

How to value parameter estimations: RMSE

Most missing: simulations that have the least non-null estimations

Least missing: simulations that have the most non-null estimations

Middle: in between

Missing Weak Predictors Analysis - Result: n=100

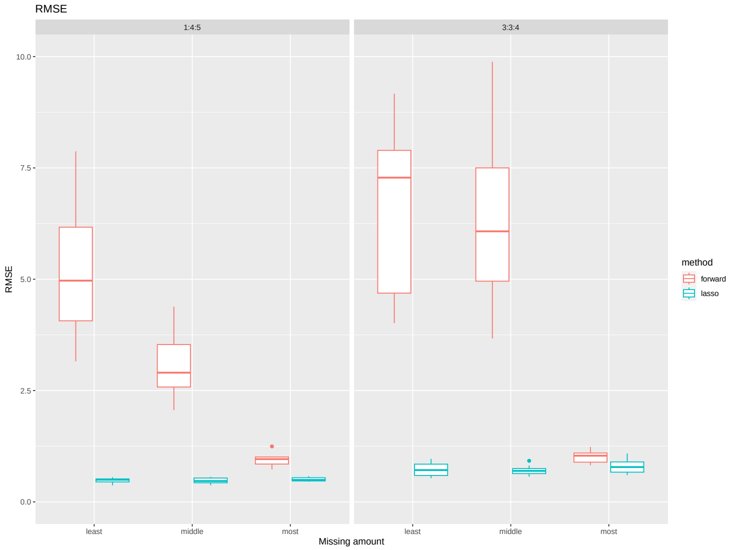

Figure 12: RMSE when n=100

Missing Weak Predictors Analysis - Result: n=500

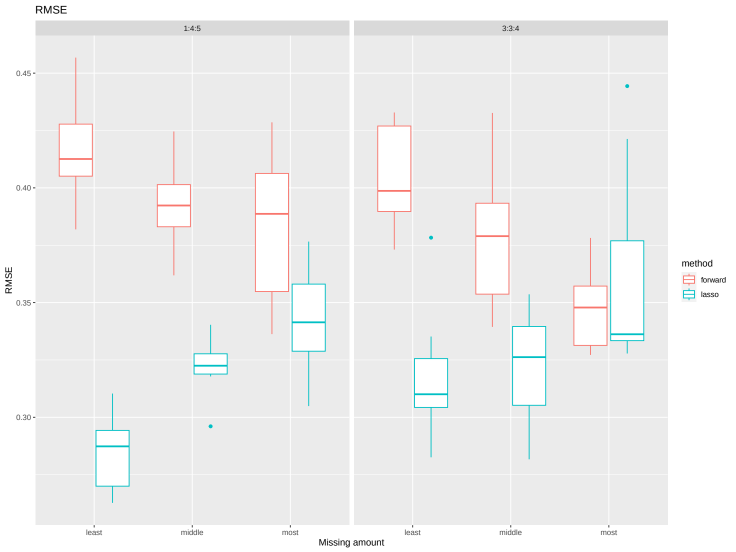

Figure 13: RMSE when n=500

Missing Weak Predictors Analysis - Result: n=2000

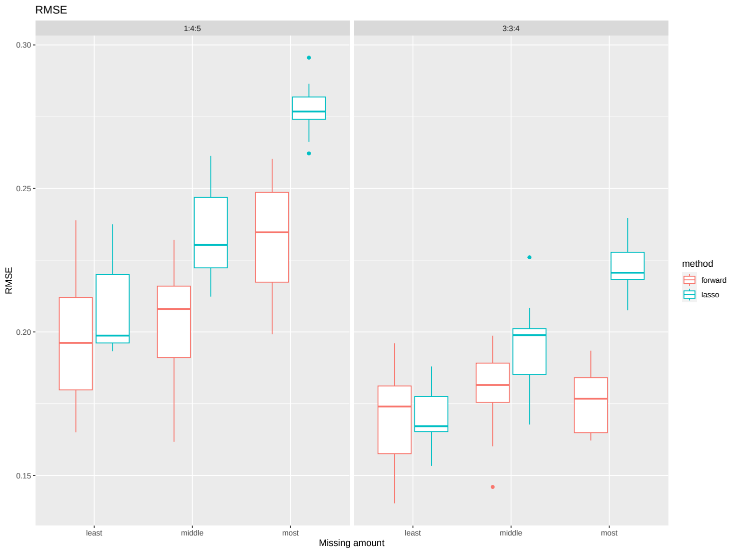

Figure 14: RMSE when n=2000

Missing Weak Predictors Analysis - Discussion

- No apparent patterns between different ratios
- In high-dimensional scenarios, Lasso performs much better than forward selection according to RMSE, no matter how much missing.
- When n is large enough, RMSE of both methods become small.
- When $n = 500$, Lasso is slightly better than forward selection, however, when $n = 2000$, just the reverse.
- In Lasso, RMSE seems to increase if the missing amount increases, but in forward selection, RMSE decreases when missing amount increases.

Summary of Results

- For binary outcomes, the simple bootstrap tended to underestimate the standard error
- Larger standard error estimates from complex bootstrap in binary and continuous settings
- Differences between simple and complex bootstrap were smaller for larger sample sizes
- Complex bootstrap not as reliable in small sample sizes

Limitations

- Sample size / treatment (or exposure) prevalence
- Small number of initial samples, limited in detecting significant differences in coverage rate

Future Work

- Larger number of initial samples, narrower coverage window
- Increased sample size, changes in bootstrap performance?
- Changes in treatment propensity model
- Non-normal distributions of covariates