Replication Document for the Main Results in "Precise Unbiased Estimation in Randomized Experiments using Auxiliary Observational Data"

1 Preliminaries

This document will reproduce all of the tables and figures from the manuscript. The tables and figures will appear in the compiled version of this PDF, as well as in stand-alone files to be incorporated into the main manuscript.

```
opts_chunk$set(message=FALSE,warning=FALSE,error=FALSE,echo=TRUE,cache=FALSE,dev='tikz')

## Error in eval(expr, envir, enclos):  object 'opts_chunk' not found
```

```
set.seed(613)
```

```
library(scales)
#library(tidyverse)
library(dplyr)

##
## Attaching package:  'dplyr'
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(ggplot2)
library(tibble)
library(purrr)

##
## Attaching package:  'purrr'
## The following object is masked from 'package:scales':
##
##     discard
```

```
library(tidyr)
library(loop.estimator)
library(kableExtra)

##
## Attaching package:  'kableExtra'
## The following object is masked from 'package:dplyr':
##
##     group_rows

library(xtable)
library(knitr)
library(tikzDevice)
library(estimatr)
library(forcats)



## specialized versions of the LOOP estimator
source('code/loop_ols.R')
source('code/loop_ext.R')
## functions for estimating effects
source('code/analysisFunctions.r')
```

Names of covariates for within-sample covariate adjustment:

```
covNames <- c(
    "Prior.Problem.Count",
    "Prior.Percent.Correct",
    "Prior.Assignments.Assigned",
    "Prior.Percent.Completion",
    "Prior.Class.Percent.Completion",
    "Prior.Homework.Assigned",
    "Prior.Homework.Percent.Completion",
    "Prior.Class.Homework.Percent.Completion",
    "male",
    "unknownGender")#)
```

2 Data

Here we load in the data for estimating effects and standard errors using several different methods discussed in the manuscript. Note that the predictions from the model fit in

the remnant are already part of the datasets (which are themselves part of the GitHub repository) under the column name `p_complete`.

Load in and clean the data:

```
source('code/dataPrep.r')
```

Replicating Table 1 from the manuscript:

```
source('code/covTable.r')
print(covTable, add.to.row=ATR)
```

	Mean	SD	% Missing
Problem Count	603.11	784.29	2
Percent Correct	0.68	0.13	2
Assignments Assigned	103.92	412.15	13
Percent Completion	0.89	0.21	13
Class Percent Completion	0.90	0.13	22
Homework Assigned	25.82	29.87	50
Homework Percent Completion	0.93	0.16	59
Class Homework Percent Completion	0.93	0.09	56
Guessed Gender	Male: 36%	Female: 36%	Unknown: 28%

Table 1: Pooled summary statistics for aggregate prior ASSISTments performance used as within-sample covariates.

2.1 Imputing Missing Covariates

To impute missing covariate values, when possible we imputed the classroom mean covariate value for students working on that skill builder. When there were no other available values for a covariate for students in the same classroom working on the same skill builder, we imputed with the global mean of students working on that skill builder. Since covariates are all pre-treatment and the imputation did not depend on treatment status, the imputed covariates are themselves covariates, measured for all subjects. Therefore, we need not correct for the imputation scheme in our treatment effect estimation.

```
### first fill in with class/problem_set mean
### if that doesn't work, fill in with problem_set mean
dat <- dat%>%
    group_by(Class.ID,problem_set)%>%
```

```
    mutate(across(all_of(covNames),~ifelse(is.finite(.),.,mean(.,na.rm=TRUE))))%>%
    group_by(problem_set)%>%
    mutate(across(all_of(covNames),~ifelse(is.finite(.),.,mean(.,na.rm=TRUE))))%>%
    ungroup()

stopifnot(all(sapply(covNames,function(x) mean(is.finite(dat[[x]])))==1))
```

3 Estimate Effects

Here we estimate effects of treatment for each of the 33 skill builders in the dataset. The functions for estimating effects are all found in the file `code/analysisFunctions.r`. This includes the function `full()` which estimates all five treatment effects discussed in the paper.

```
fullres <- sapply(levels(dat$problem_set),full,dat=dat,covNames=covNames,simplify=FALSE)

## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
```

```
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
```

```
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
```

```
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
```

```
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
```

```
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
```

```
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
```

```
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
```

```
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
```

```
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
```

```
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
```

```
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
```

```
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
```

```
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
```

```
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
```

```
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
```

```
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
```

```
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
```

```
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
```

```
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
: The response has five or fewer unique values.  Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n): Recycling array of length 1 in
vector-array arithmetic is deprecated.
```

```
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 1, , drop = FALSE], Y[Tr == 1, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
## Warning in randomForest.default(Z[Tr == 0, , drop = FALSE], Y[Tr == 0, ,
:  The response has five or fewer unique values.   Are you sure you want to
do regression?
## Warning in rfout$mse/(var(y) * (n - 1)/n):  Recycling array of length 1 in
vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
```

```
### name the problem sets based on the SE from the simple difference estimator
rnk <- rank(sapply(fullres,function(x) x['simpDiff','se']))
names(fullres) <- as.character(rnk)

for(i in 1:length(fullres))
    attr(fullres[[i]],'psid') <- levels(dat$problem_set)[i]

save(fullres,file='results/fullres.RData')

dat$ps <- rnk[as.character(dat$problem_set)]
```

Replicate Table 2 (Now in an appendix, I think). The numbering of the experiments derives from the estimated standard errors, so this comes after effect estimation.

```
source('code/psTable.r')
```

```
## `summarise()` has grouped output by 'ps'.  You can override using the `.groups`
## argument.
```

```
 kbl(tab,

           booktabs=FALSE,
           col.names=rep(c("",rep(c("Trt","Ctl"),2)),2),
           caption="Sample sizes and \\% homework completion by treatment group in each
           label="info")%>%
    kable_styling()%>%
    column_spec(5,border_right=TRUE)%>%
    add_header_above(rep(c("Experiment"=1,"n"=2,"% Complete"=2),2))
```

Table 2: Sample sizes and % homework completion by treatment group in each of the 33 A/B tests.

Experiment	n		% Complete		Experiment	n		% Complete	
	Trt	Ctl	Trt	Ctl		Trt	Ctl	Trt	Ctl
1	956	961	93	93	18	188	193	89	85
2	330	365	98	96	19	199	213	89	82
3	680	650	86	88	20	264	281	81	79
4	943	921	70	68	21	242	266	81	76
5	931	900	61	64	22	215	211	82	82
6	355	349	88	88	23	281	234	73	69
7	492	463	79	81	24	269	288	65	59
8	231	197	92	91	25	224	233	73	74
9	367	387	83	82	26	270	253	63	61
10	617	587	67	62	27	228	244	68	64
11	338	289	88	84	28	201	228	73	69
12	478	476	76	73	29	238	259	44	54
13	193	209	93	89	30	74	92	91	84
14	404	451	73	69	31	69	67	91	87
15	265	275	85	84	32	76	81	62	70
16	165	170	92	89	33	15	11	73	55
17	259	246	82	85	NA	NA	NA	NA	NA

4 Figures

The following code creates a dataset called `comparisons` that includes the sampling variance ratios comparing each method to the others, for each problem set. It also produces a table (which is not in the manuscript) giving the estimated standard error for each method and each experiment.

```
source('code/figurePrep.r')

pwidePrint <- pwide
names(pwidePrint)[-1] <- paste0('$',methodName[names(pwidePrint)[-1]],'$')
kable(pwidePrint,row.names=FALSE,caption="Estimated standard error for the ATE in each s
```

Figure 1, comparing $\hat{\tau}^{\mathrm{DM}}$, $\hat{\tau}^{\mathrm{RE}}$, and $\hat{\tau}^{\mathrm{SS}}[x^r, \mathrm{LS}]$:

```
p <- comparisons%>%
    filter(method1%in%c('ReLOOP','Rebar'),method2%in%c('ReLOOPEN','Rebar','SimpleDiffere
    ggplot(aes(ssMult))+#,fill=exGroup))+
    geom_dotplot( method="histodot", binwidth = .05 )  +
    labs( x = "Relative Ratio of Sample Variances", y="" ) +
    geom_vline( xintercept = 1, col="red" ) +
    facet_wrap(~comp,nrow=1)+
    theme(legend.position = "none",
        panel.grid = element_blank(),
        axis.title.y = element_blank(),
        axis.text.y= element_blank(),
        axis.ticks.y = element_blank(),
        text=element_text(size=12),
        strip.text=element_text(size=12,lineheight=0.5))

tikz('figure/fig4.tex',width=6.4,height=2,standAlone=FALSE)
print(p)
dev.off()

## pdf
##   2
```

Figure 2, comparing $\hat{\tau}^{\mathrm{DM}}$, $\hat{\tau}^{\mathrm{SS}}[x^r, \mathrm{LS}]$, $\hat{\tau}^{\mathrm{SS}}[\boldsymbol{x}, \mathrm{RF}]$, and $\hat{\tau}^{\mathrm{SS}}[\tilde{\boldsymbol{x}}, \mathrm{EN}]$:

```
p <- comparisons%>%
    filter((method1%in%c('ReLOOPEN')&method2%in%c('Loop','ReLOOP','SimpleDifference')))%
    mutate(comp=factor(comp,levels=unique(as.character(comp))))%>%
```

Table 3: Estimated standard error for the ATE in each skill builder, using each method discussed in the manuscript

experiment	$\hat{\tau}^{\mathrm{SS}}[\tilde{\boldsymbol{x}}, \mathrm{EN}]$	$\hat{\tau}^{\mathrm{SS}}[x^r, \mathrm{LS}]$	$\hat{\tau}^{\mathrm{SS}}[\boldsymbol{x}, \mathrm{RF}]$	$\hat{\tau}^{\mathrm{RE}}$	$\hat{\tau}^{\mathrm{DM}}$
1	0.010	0.011	0.011	0.011	0.012
10	0.021	0.024	0.021	0.024	0.028
11	0.026	0.027	0.026	0.028	0.028
12	0.022	0.026	0.022	0.026	0.028
13	0.029	0.029	0.030	0.032	0.029
14	0.028	0.029	0.030	0.029	0.031
15	0.028	0.029	0.029	0.029	0.031
16	0.031	0.031	0.031	0.031	0.032
17	0.031	0.032	0.031	0.032	0.033
18	0.032	0.032	0.033	0.033	0.034
19	0.035	0.034	0.037	0.038	0.034
2	0.013	0.012	0.013	0.017	0.013
20	0.032	0.033	0.033	0.034	0.034
21	0.034	0.034	0.035	0.034	0.036
22	0.035	0.036	0.034	0.036	0.037
23	0.033	0.038	0.035	0.038	0.040
24	0.030	0.040	0.029	0.041	0.041
25	0.038	0.040	0.038	0.040	0.041
26	0.031	0.034	0.030	0.035	0.042
27	0.038	0.040	0.038	0.040	0.044
28	0.043	0.044	0.043	0.046	0.044
29	0.045	0.045	0.047	0.047	0.045
3	0.016	0.018	0.016	0.018	0.018
30	0.050	0.050	0.054	0.050	0.052
31	0.049	0.049	0.050	0.051	0.054
32	0.063	0.068	0.060	0.067	0.076
33	0.129	0.136	0.153	0.145	0.197
4	0.017	0.019	0.017	0.020	0.021
5	0.019	0.019	0.019	0.019	0.023
6	0.019	0.022	0.019	0.021	0.025
7	0.019	0.022	0.019	0.022	0.026
8	0.026	0.026	0.028	0.027	0.027
9	0.025	0.027	0.025	0.028	0.027

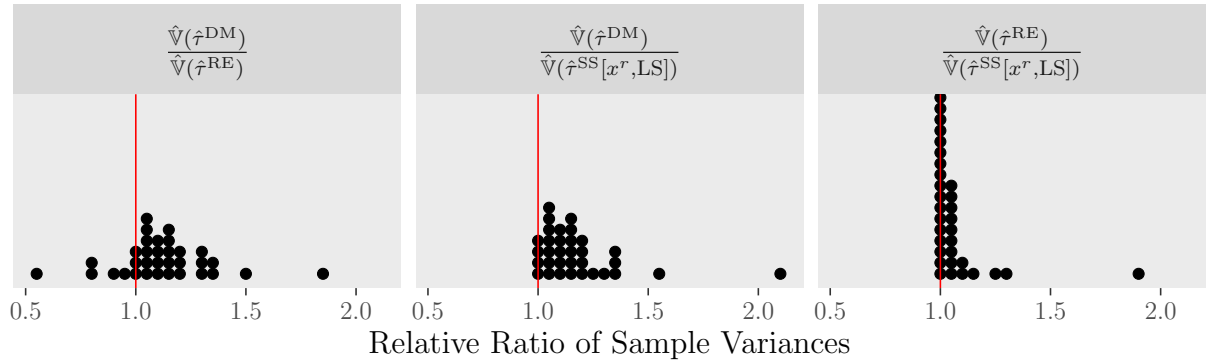

Figure 1: A dotplot showing sample size multipliers (i.e. sampling variance ratios) comparing $\hat{\tau}^{\mathrm{DM}}$, $\hat{\tau}^{\mathrm{RE}}$, and $\hat{\tau}^{\mathrm{SS}}[x^r, \mathrm{LS}]$ on the 33 ASSISTments TestBed experiments.

```
ggplot(aes(ssMult))+#,fill=exGroup))+
    geom_dotplot( method="histodot", binwidth = .05 )  +
    labs( x = "Relative Ratio of Sample Variances", y="" ) +
    geom_vline( xintercept = 1, col="red" ) +
    facet_wrap(~comp,nrow=1)+
    theme(legend.position = "none",
        panel.grid = element_blank(),
        axis.title.y = element_blank(),
        axis.text.y= element_blank(),
        axis.ticks.y = element_blank(),
        text=element_text(size=12),
        strip.text=element_text(size=12,lineheight=0.5))
#print(p)

tikz('figure/fig5alt.tex',width=6.4,height=2,standAlone=FALSE)
print(p)

dev.off()

## pdf
##   2
```

The following code reproduces some of the numbers in the manuscript text describing the results:

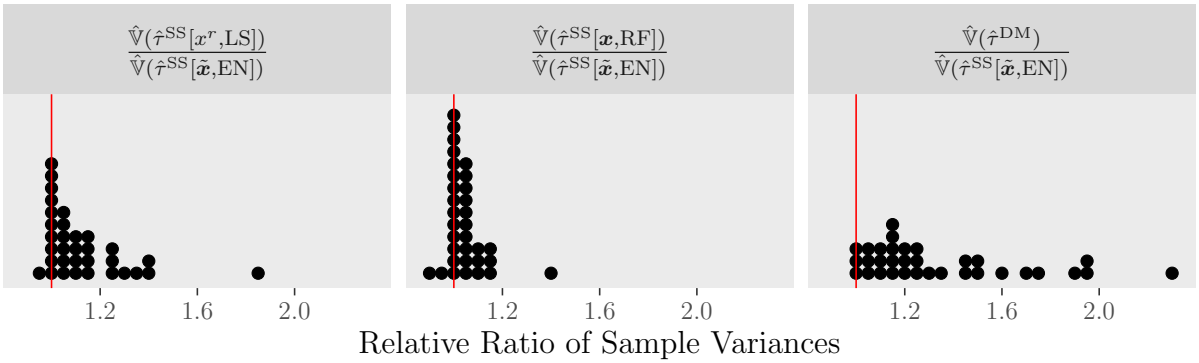

Figure 2: A dotplot showing sample size multipliers (i.e. sampling variance ratios) comparing $\hat{\tau}^{SS}[\tilde{\boldsymbol{x}}, \text{EN}]$ to $\hat{\tau}^{SS}[x^r, \text{LS}]$ $\hat{\tau}^{SS}[\boldsymbol{x}; \text{RF}]$, and $\hat{\tau}^{DM}$, respectively, on the 33 ASSISTments TestBed experiments.

```
compTab <- comparisons%>%group_by(method1,method2)%>%
  summarize(
            worse=sum(ssMult<0.975),
            equal=sum(abs(ssMult-1)<0.025),
            better=sum(ssMult>1.025),
            best=max(ssMult),
            bestPS=experiment[which.max(ssMult)],
            best2=sort(ssMult,decreasing=TRUE)[2],
            best2ps=experiment[rank(ssMult)==32],
            worst=min(ssMult),
            worstPS=experiment[which.min(ssMult)]
            )%>%ungroup()%>%
    mutate(across(starts_with('method'),~paste0('$',methodName[as.character(.)],'$')))

## `summarise()` has grouped output by 'method1'.  You can override using the
## `.groups` argument.

compTab%>%select(method1:bestPS)%>%kable(escape = FALSE)
```

method1	method2	worse	equal	better	best	bestPS
$\hat{\tau}^{\text{SS}}[\tilde{\boldsymbol{x}}, \text{EN}]$	$\hat{\tau}^{\text{SS}}[x^r, \text{LS}]$	1	10	22	1.844856	24
$\hat{\tau}^{\text{SS}}[\tilde{\boldsymbol{x}}, \text{EN}]$	$\hat{\tau}^{\text{SS}}[\boldsymbol{x}, \text{RF}]$	2	14	17	1.400882	33
$\hat{\tau}^{\text{SS}}[\tilde{\boldsymbol{x}}, \text{EN}]$	$\hat{\tau}^{\text{RE}}$	1	2	30	1.905476	24
$\hat{\tau}^{\text{SS}}[\tilde{\boldsymbol{x}}, \text{EN}]$	$\hat{\tau}^{\text{DM}}$	0	3	30	2.314750	33
$\hat{\tau}^{\text{SS}}[x^r, \text{LS}]$	$\hat{\tau}^{\text{SS}}[\boldsymbol{x}, \text{RF}]$	18	3	12	1.266307	33
$\hat{\tau}^{\text{SS}}[x^r, \text{LS}]$	$\hat{\tau}^{\text{RE}}$	0	18	15	1.898485	2
$\hat{\tau}^{\text{SS}}[x^r, \text{LS}]$	$\hat{\tau}^{\text{DM}}$	0	4	29	2.092385	33
$\hat{\tau}^{\text{SS}}[\boldsymbol{x}, \text{RF}]$	$\hat{\tau}^{\text{RE}}$	5	4	24	1.926845	24
$\hat{\tau}^{\text{SS}}[\boldsymbol{x}, \text{RF}]$	$\hat{\tau}^{\text{DM}}$	5	1	27	1.957049	26
$\hat{\tau}^{\text{RE}}$	$\hat{\tau}^{\text{DM}}$	5	3	25	1.848485	33

```
compTab%>%select(method1,method2,best2:worstPS)%>%kable(escape=FALSE)
```

method1	method2	best2	best2ps	worst	worstPS
$\hat{\tau}^{\text{SS}}[\tilde{\boldsymbol{x}}, \text{EN}]$	$\hat{\tau}^{\text{SS}}[x^r, \text{LS}]$	1.417204	12	0.9680705	2
$\hat{\tau}^{\text{SS}}[\tilde{\boldsymbol{x}}, \text{EN}]$	$\hat{\tau}^{\text{SS}}[\boldsymbol{x}, \text{RF}]$	1.142609	30	0.9088411	32
$\hat{\tau}^{\text{SS}}[\tilde{\boldsymbol{x}}, \text{EN}]$	$\hat{\tau}^{\text{RE}}$	1.837867	2	0.9692726	30
$\hat{\tau}^{\text{SS}}[\tilde{\boldsymbol{x}}, \text{EN}]$	$\hat{\tau}^{\text{DM}}$	1.933998	26	0.9893321	13
$\hat{\tau}^{\text{SS}}[x^r, \text{LS}]$	$\hat{\tau}^{\text{SS}}[\boldsymbol{x}, \text{RF}]$	1.150106	30	0.5360364	24
$\hat{\tau}^{\text{SS}}[x^r, \text{LS}]$	$\hat{\tau}^{\text{RE}}$	1.277514	13	0.9756321	30
$\hat{\tau}^{\text{SS}}[x^r, \text{LS}]$	$\hat{\tau}^{\text{DM}}$	1.563645	26	0.9931606	28
$\hat{\tau}^{\text{SS}}[\boldsymbol{x}, \text{RF}]$	$\hat{\tau}^{\text{RE}}$	1.832200	2	0.8482977	30
$\hat{\tau}^{\text{SS}}[\boldsymbol{x}, \text{RF}]$	$\hat{\tau}^{\text{DM}}$	1.949483	24	0.8744120	19
$\hat{\tau}^{\text{RE}}$	$\hat{\tau}^{\text{DM}}$	1.515003	26	0.5588199	2

4.1 Comparing Sample Splitting to ANCOVA Estimators

The following creates the figures in 4.3 (plus some others)

This estimates the effects and their SEs:

```
ols <- sapply(levels(dat$problem_set),full,dat=dat,covNames=covNames,simplify=FALSE,
                  methods=c('reloopLin','reloopPoor','reloopPlusLin','reloopPlusPoor','l

save(ols,file='results/ols.RData')
```

```
source('code/olsFigurePrep.r')
```

```
pLin1 <- comparisonsLin%>%
    mutate(comp=factor(comp,levels=unique(as.character(comp))))%>%
ggplot(aes(ssMult))+#,fill=exGroup))+
    geom_dotplot( method="histodot", binwidth = .05 )  +
    labs( x = "Relative Ratio of Sample Variances", y="" ) +
    geom_vline( xintercept = 1, col="red" ) +
    facet_wrap(~comp)+#,nrow=1)+
    theme(legend.position = "none",
        panel.grid = element_blank(),
        axis.title.y = element_blank(),
        axis.text.y= element_blank(),
        axis.ticks.y = element_blank(),
        text=element_text(size=12),
        strip.text=element_text(size=12,lineheight=0.5))

tikz('figure/appendixLin.tex',width=6.4,height=4,standAlone=FALSE,packages= c(getOption(
print(pLin1)

## Warning:  Removed 3 rows containing missing values (`stat_bindot()`).

dev.off()

## pdf
##   2

### compare Lin to Reloop

pLin2 <- comparisonsReloopLin%>%
    mutate(comp=factor(comp,levels=unique(as.character(comp))))%>%
ggplot(aes(ssMult))+#,fill=exGroup))+
    geom_dotplot( method="histodot", binwidth = .02 )  +
    labs( x = "Relative Ratio of Sample Variances (Log 10 Scaled)", y="" ) +
    geom_vline( xintercept = 1, col="red" ) +
    facet_wrap(~comp,nrow=1)+
    theme(legend.position = "none",
        panel.grid = element_blank(),
        axis.title.y = element_blank(),
        axis.text.y= element_blank(),
        axis.ticks.y = element_blank(),
        text=element_text(size=12),
        strip.text=element_text(size=12,lineheight=0.5))+
  scale_x_continuous(trans="log10")
```

```
tikz('figure/appendixLinReloop.tex',width=4,height=2,standAlone=FALSE,packages= c(getOpt
print(pLin2)

## Warning:  Removed 1 rows containing missing values (`stat_bindot()`).

dev.off()

## pdf
##   2
```

```
pOls1 <- comparisonsPoor%>%
    mutate(comp=factor(comp,levels=unique(as.character(comp))))%>%
ggplot(aes(ssMult))+#,fill=exGroup))+
    geom_dotplot( method="histodot", binwidth = .05 )  +
    labs( x = "Relative Ratio of Sample Variances (Log 10 Scaled)", y="" ) +
    geom_vline( xintercept = 1, col="red" ) +
    facet_wrap(~comp)+#,nrow=1)+
    theme(legend.position = "none",
        panel.grid = element_blank(),
        axis.title.y = element_blank(),
        axis.text.y= element_blank(),
        axis.ticks.y = element_blank(),
        text=element_text(size=12),
        strip.text=element_text(size=12,lineheight=0.5))

tikz('figure/appendixPoor.tex',width=6.4,height=4,standAlone=FALSE,packages= c(getOptior
print(pOls1)
dev.off()

## pdf
##   2

pOls2 <- comparisonsReloopPoor%>%
    mutate(comp=factor(comp,levels=unique(as.character(comp))))%>%
ggplot(aes(ssMult))+#,fill=exGroup))+
    geom_dotplot( method="histodot", binwidth = .02 )  +
    labs( x = "Relative Ratio of Sample Variances", y="" ) +
    geom_vline( xintercept = 1, col="red" ) +
    facet_wrap(~comp,nrow=1)+
    theme(legend.position = "none",
        panel.grid = element_blank(),
```

```
        axis.title.y = element_blank(),
        axis.text.y= element_blank(),
        axis.ticks.y = element_blank(),
        text=element_text(size=12),
        strip.text=element_text(size=12,lineheight=0.5))

tikz('figure/appendixOlsReloop.tex',width=4,height=2,standAlone=FALSE,packages= c(getOpt
print(pOls2)
dev.off()

## pdf
##   2

#### this one actually made it into the manuscript:

pOls3 <- ggplot(newcomp,aes(ssMult))+#,fill=exGroup))+
    geom_dotplot( method="histodot", binwidth = .01 )  +
    labs( x = "Relative Ratio of Sample Variances", y="" ) +
    geom_vline( xintercept = 1, col="red" ) +
    facet_wrap(~comp,nrow=3)+
    theme(legend.position = "none",
        panel.grid = element_blank(),
        axis.title.y = element_blank(),
        axis.text.y= element_blank(),
        axis.ticks.y = element_blank(),
        text=element_text(size=12),
        strip.text=element_text(size=12,lineheight=0.5))+
  scale_x_continuous(trans="log10",breaks=c(0.85,1,1.2,1.4,1.7,2,2.5))


tikz('figure/OlsReloop.tex',width=5,height=6,standAlone=FALSE,packages= c(getOption('tik
print(pOls3)
dev.off()

## pdf
##   2
```

```
sessionInfo()

## R version 4.2.2 (2022-10-31)
## Platform: x86_64-pc-linux-gnu (64-bit)
```

```
## Running under: Debian GNU/Linux 11 (bullseye)
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/blas/libblas.so.3.9.0
## LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.9.0
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
##  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] forcats_0.5.2         estimatr_1.0.0        tikzDevice_0.12.3.1
##  [4] knitr_1.40            xtable_1.8-4          kableExtra_1.3.4
##  [7] loop.estimator_1.0.0  tidyr_1.2.0           purrr_0.3.4
## [10] tibble_3.1.8          ggplot2_3.4.1         dplyr_1.0.10
## [13] scales_1.2.1          languageserver_0.3.15 httpgd_1.3.1
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_1.1.2     xfun_0.32            colorspace_2.0-3
##  [4] vctrs_0.5.2          generics_0.1.3       htmltools_0.5.3
##  [7] viridisLite_0.4.1    utf8_1.2.2           rlang_1.0.6
## [10] later_1.3.0          pillar_1.8.1         glue_1.6.2
## [13] withr_2.5.0          DBI_1.1.3            lifecycle_1.0.3
## [16] stringr_1.4.1        munsell_0.5.0        gtable_0.3.0
## [19] rvest_1.0.3          evaluate_0.16        labeling_0.4.2
## [22] callr_3.7.3          fastmap_1.1.0        ps_1.7.1
## [25] parallel_4.2.2       fansi_1.0.3          highr_0.9
## [28] Rcpp_1.0.10          webshot_0.5.4        filehash_2.4-3
## [31] farver_2.1.1         systemfonts_1.0.4    digest_0.6.29
## [34] stringi_1.7.12       processx_3.7.0       grid_4.2.2
## [37] cli_3.6.0            tools_4.2.2          magrittr_2.0.3
## [40] randomForest_4.7-1.1 Formula_1.2-4        pkgconfig_2.0.3
## [43] ellipsis_0.3.2       xml2_1.3.3           assertthat_0.2.1
## [46] rmarkdown_2.16       svglite_2.1.0        httr_1.4.5
```

```
## [49] rstudioapi_0.14      R6_2.5.1             compiler_4.2.2
```