Growth Mindset on Student Achievements

Minh Thu Bui, Vindyani Herath

Exploratory Data Analysis:

```
print(paste("The number of rows in the dataset is:", dim(data)[1]))
```

```
## [1] "The number of rows in the dataset is: 10391"
```

```
print(paste("The number of columns in the dataset is:", dim(data)[2]))
```

```
## [1] "The number of columns in the dataset is: 13"
```

```
print(paste("The number of students who received the intervention is:", sum(data$Intervention == 1)))
```

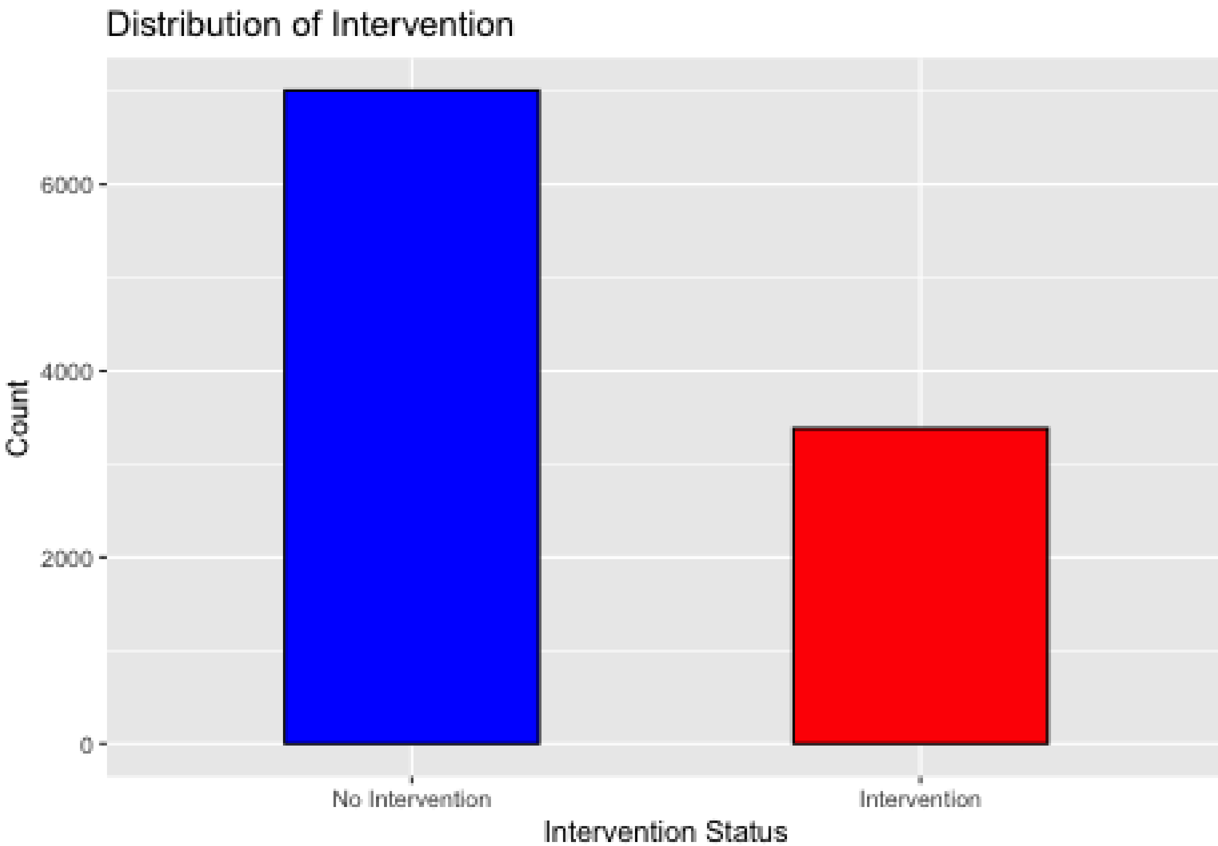
```
## [1] "The number of students who received the intervention is: 3384"
```

```
print(paste("The number of students who did not received the intervention is:", sum(data$Intervention =
```

```
## [1] "The number of students who did not received the intervention is: 7007"
```

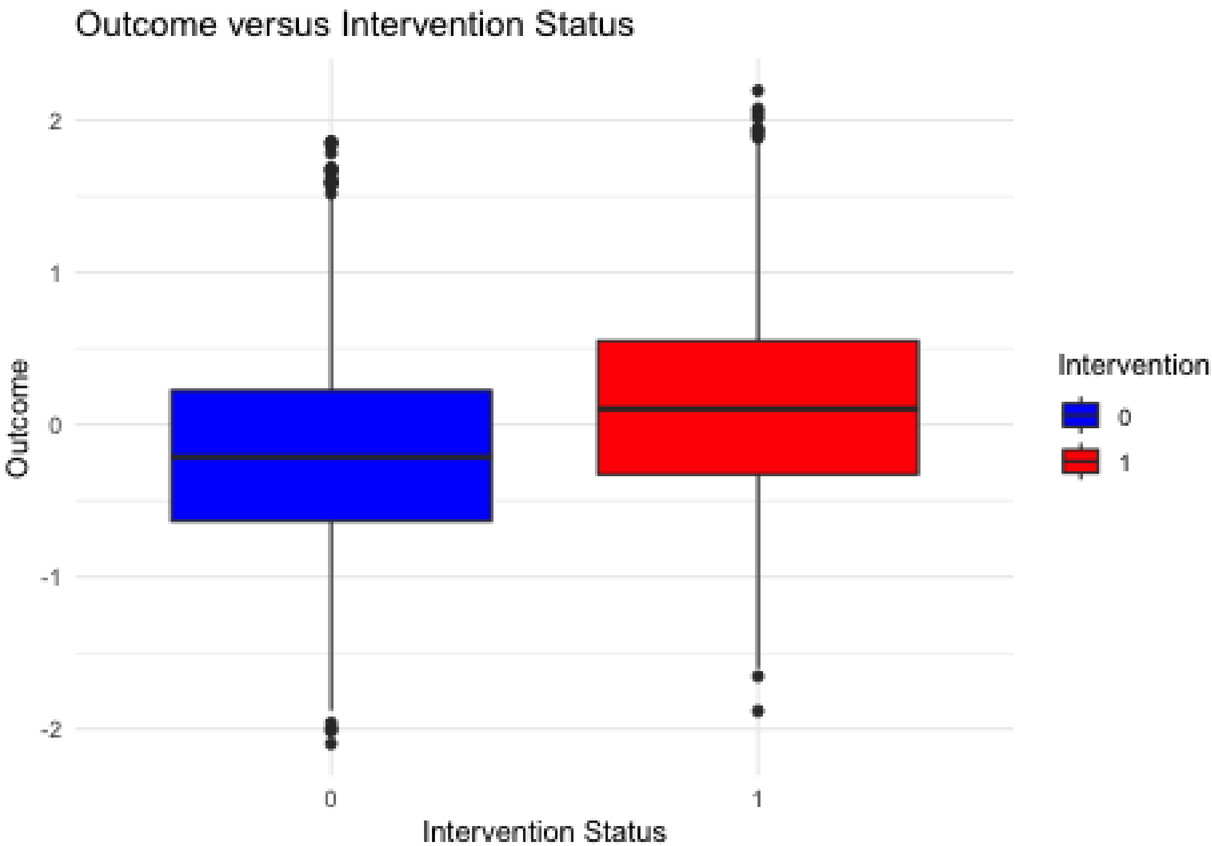
First, we want to see the distribution of people who received the intervention and who did not in the total dataset.

```
# Plot the number of students who receive intervention and the amount of people that did not.
ggplot(data, aes(x = Intervention)) +
  geom_bar(fill = c("blue", "red"), color = "black", width = 0.5) +
  labs(x = "Intervention Status", y = "Count",
       title = "Distribution of Intervention") +
  scale_x_discrete(labels = c("No Intervention", "Intervention"))
```



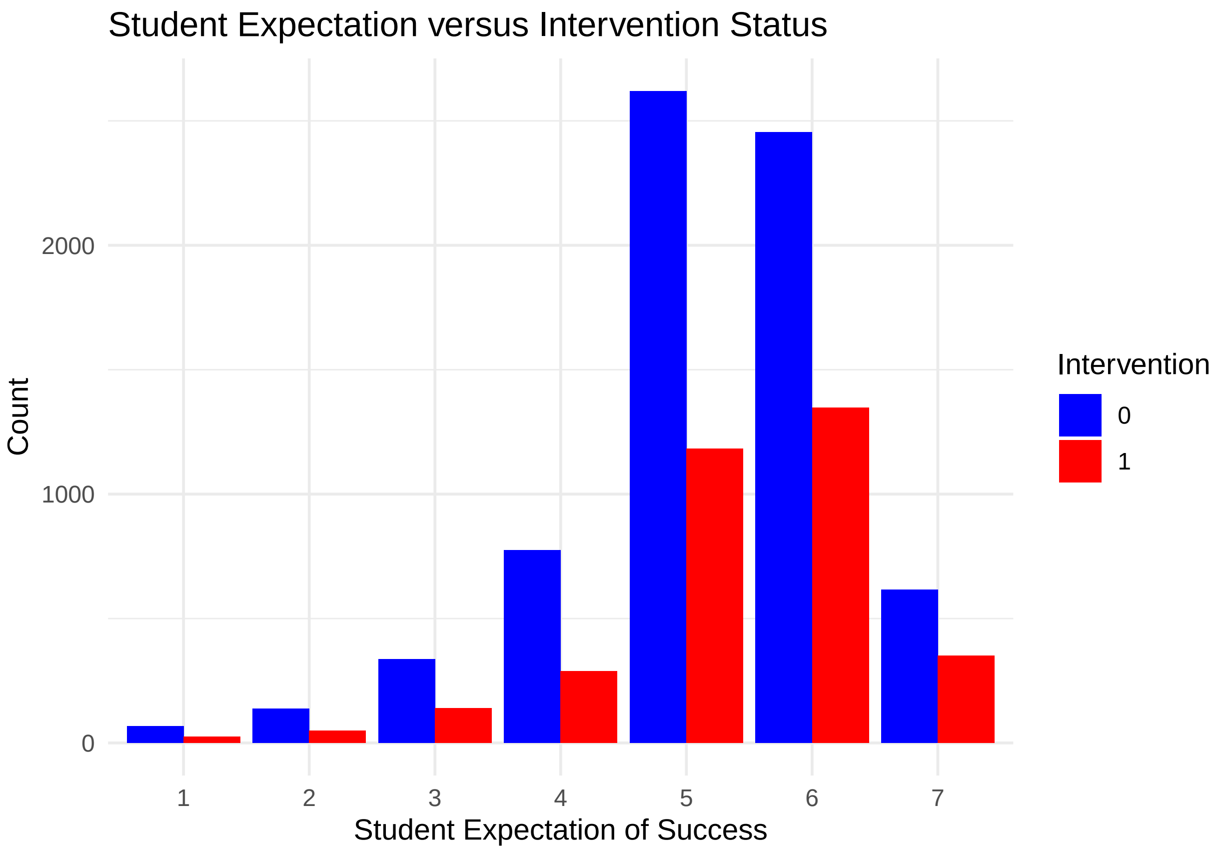
Next, we also want to investigate the relationship between the intervention status (whether they received it or not) and the outcome. It seems like the mean of the outcomes is higher and on the positive scale for people who received treatment while the mean is negative for the control group.

```
ggplot(data, aes(x = Intervention, y = Outcome, fill = Intervention)) +
  geom_boxplot() +
  labs(x = "Intervention Status", y = "Outcome", title = "Outcome versus Intervention Status") +
  scale_fill_manual(values = c("blue", "red")) +
  theme_minimal()
```



We also want to investigate whether the assignment of treatment is randomized or if there was some selection bias to consider in our analysis. First, we visualize to inspect whether students with a higher expectation of success appear to be more likely to receive treatment.

```
ggplot(data, aes(x = StudentExpectation, fill = Intervention)) +
  geom_bar(position = "dodge") +
  labs(x = "Student Expectation of Success", y = "Count",
       title = "Student Expectation versus Intervention Status") +
  scale_fill_manual(values = c("blue", "red")) +
  theme_minimal()
```

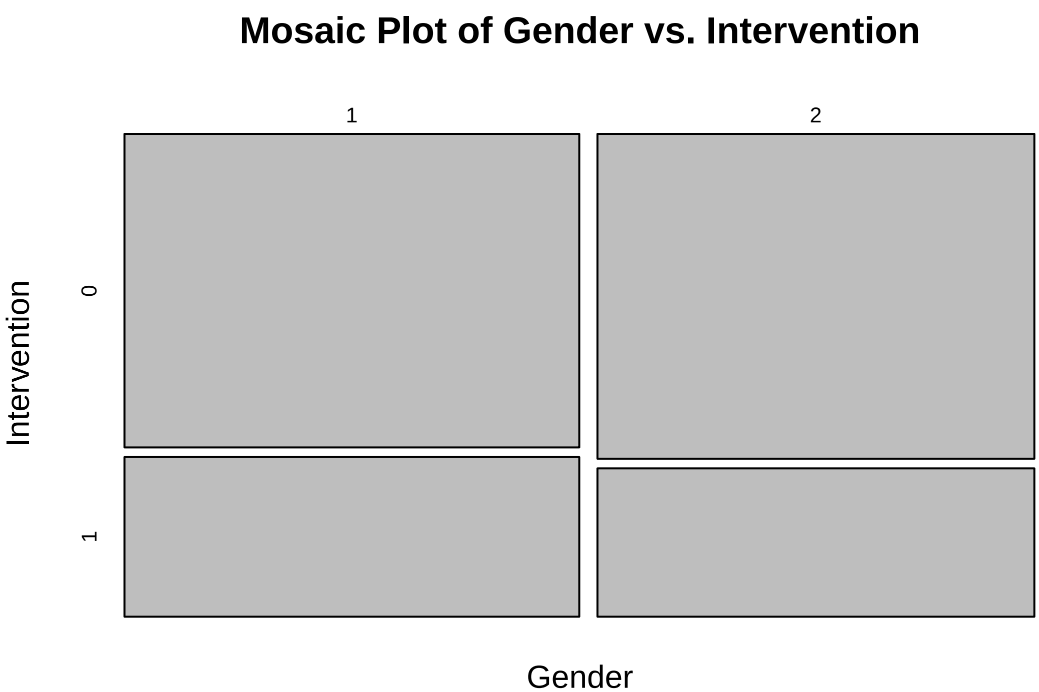


We also create a contingency table between the binary variable Gender and the binary variable for Intervention. We see that for Gender decoded as 1 (Male) tends to have higher chances of getting the treatment compared to the Gender decoded as 2 (Female).

```
table_data <- table(data$Gender, data$Intervention)
print(table_data)
```

```
##
##        0    1
##   1 3512 1788
##   2 3495 1596
```

```
mosaicplot(table_data, main="Mosaic Plot of Gender vs. Intervention", xlab="Gender", ylab="Intervention
```



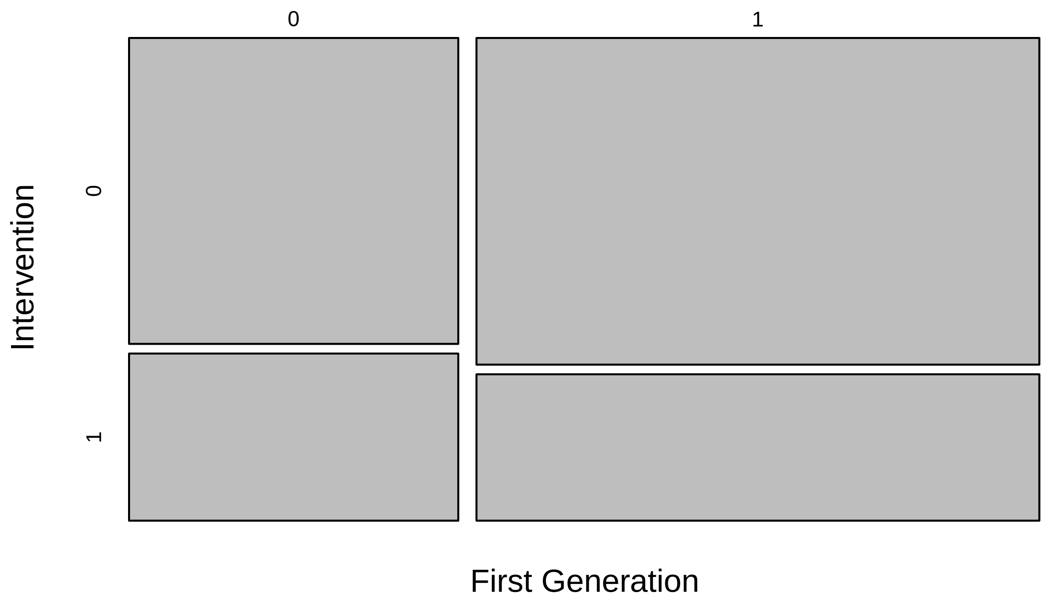
Then, the relationship between being first-generation versus being assigned the treatment shown in the table below. It seems like the randomization does occur here but there could be some sort of selection bias because the number of first-generation students receiving the treatment or not is almost double compared to the number of non-first generation students.

```
table_data <- table(data$FirstGen, data$Intervention)
print(table_data)
```

```
##
##        0    1
##   0 2480 1355
##   1 4527 2029
```

```
mosaicplot(table_data, main="Mosaic Plot of First Generation vs. Intervention", xlab="First Generation"
```

Mosaic Plot of First Generation vs. Intervention




```r
#Checking whether school achievement level have a relationship with the intervention and outcome
#Achievement levels: low, 25th percentile or lower, middle, 25th - 75th percentile; high, 75th percenti
summary(data$AchievementLevels)
```

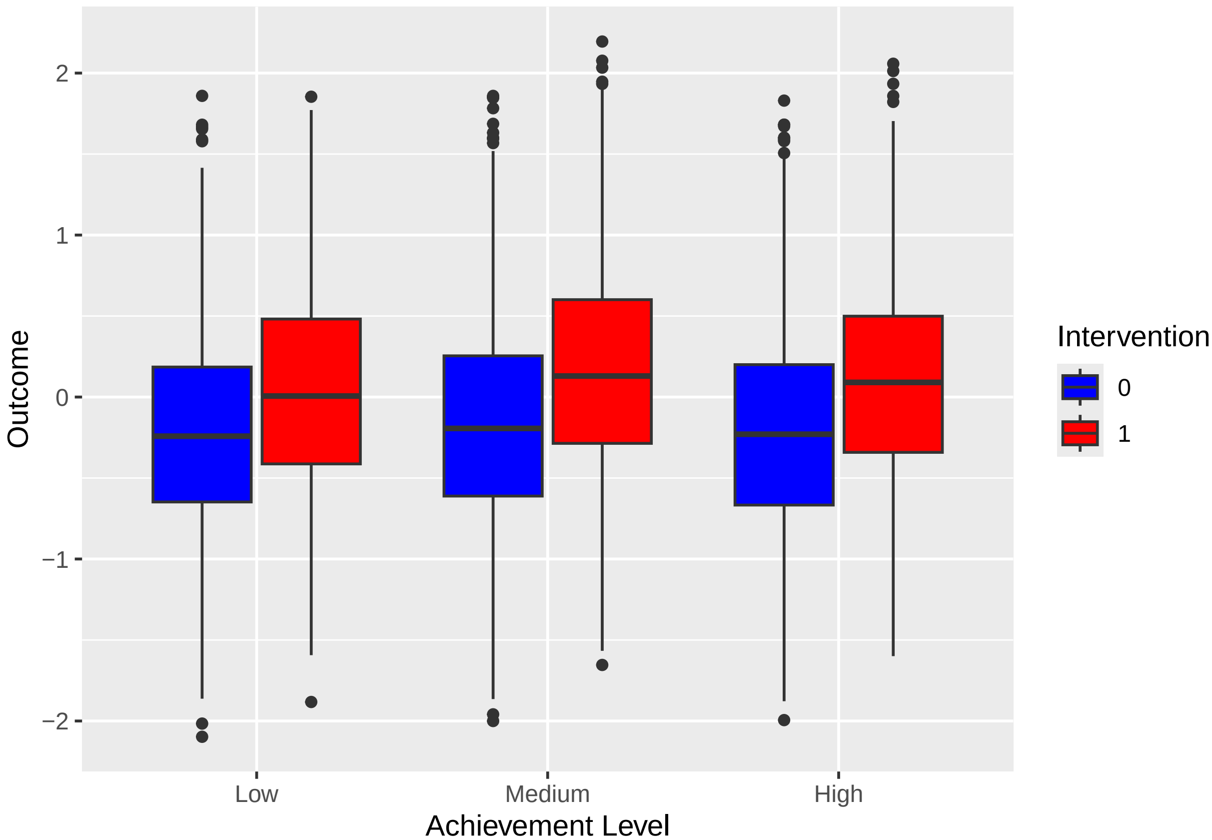
```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -3.34782 -0.54451 -0.02251  0.05484  0.72684  2.17182
```

```r
data$AchievementLevels_Cat <- ifelse(data$AchievementLevels > 0.72684 , "High",
                                  ifelse(data$AchievementLevels < -0.54451, "Low", "Medium"))
data$AchievementLevels_Cat <- factor(data$AchievementLevels_Cat, levels = c("Low", "Medium", "High"))
table(data$AchievementLevels_Cat)
```

```
##
##    Low Medium   High
##   2532   5488   2371
```

```r
# library
#library(ggplot2)

# grouped boxplot
ggplot(data, aes(x=AchievementLevels_Cat, y=Outcome, fill=Intervention)) +
    geom_boxplot() +  scale_fill_manual(values = c("blue", "red")) + labs(x = "Achievement Level", y =
```



This dataset exhibits two methodological challenges. First, although the National Study itself was a randomized study, there seems to be some selection effects in the synthetic data used here. Second, the students in this study are not independently sampled; rather, they are all drawn from 76 randomly selected schools, and there appears to be considerable heterogeneity across schools. Such a situation could arise if there are unobserved school-level features that are important treatment effect modifiers; for example, some schools may have leadership teams who implemented the intervention better than others, or may have a student culture that is more receptive to the treatment [Athey and Wager, 2018].

Modeling:

```
set.seed(1)
sample <- sample.int(n = nrow(data), size = floor(0.5*nrow(data)), replace = F)
train <- data[sample, ]
test  <- data[-sample, ]
```

Outcome Regression Approach:

```
compute_OR_ATE <- function(train_data, test_data){
    model1 <- lm(Outcome ~ Race + Gender + FirstGen + Urbanicity + FixedMindsets + AchievementLevels + S
                         + SchoolPovertyPercent + SchoolSize, data = train_data[train_data$Interventio
    model0 <- lm(Outcome ~ Race + Gender + FirstGen + Urbanicity + FixedMindsets + AchievementLevels + S
                         + SchoolPovertyPercent + SchoolSize, data = train_data[train_data$Interventio
```

```
    mu1 <- mean(as.matrix(predict(model1, newdata = test_data, type = "response")))
    mu0 <- mean(as.matrix(predict(model0, newdata = test_data, type = "response")))

    return(mu1 - mu0)
}
compute_OR_ATE(train, test)
```

```
## [1] 0.278849
```

Propensity Score Approach:

```
ate_OR <- function(train_data, test_data) {
    propensity_model <- glm(formula = Intervention ~ StudentExpectation + Race + Gender + FirstGen + Url
    train_data$propensity_score <- predict(propensity_model, type = "response")

    propensity_treatment <- lm(Outcome ~ Race + Gender + FirstGen + Urbanicity + FixedMindsets + Achieve
                                + SchoolPovertyPercent + SchoolSize, data = train_data[train_data$Interventio
    propensity_control <- lm(Outcome ~ Race + Gender + FirstGen + Urbanicity + FixedMindsets + Achieveme
                                + SchoolPovertyPercent + SchoolSize, data = train_data[train_data$Interventi
    test_data$propensity_score <- predict(propensity_model, newdata = test_data, type = "response")
    mu1 <- mean(as.matrix(predict(propensity_treatment, newdata = test_data)))
    mu0 <- mean(as.matrix(predict(propensity_control, newdata = test_data)))
    ATE_x = mu1 - mu0
    return(ATE_x)
}
ate_OR(train,test)
```

```
## [1] 0.278849
```

```
n_bootstrap <- 1000
# Bootstrap procedure
set.seed(123)
bootstrap <- function(train_data, test_data, n_bootstrap = 1000){
  results <- numeric(n_bootstrap)
  for (i in 1:n_bootstrap){
    bootstrap_sample <- train_data[sample(nrow(train_data), replace = TRUE), ]
    results[i] <- ate_OR(bootstrap_sample, test_data)
  }
  return(results)
}

bootstrap_results <- bootstrap(train, test, n_bootstrap)
#Standard Error
OR_SE <- sd(bootstrap_results)
OR_SE
```

```
## [1] 0.01867229
```

```r
# Compute confidence intervals
confidence_interval <- quantile(bootstrap_results, c(0.025, 0.975))
confidence_interval
```

```
##      2.5%     97.5%
## 0.2420477 0.3161149
```

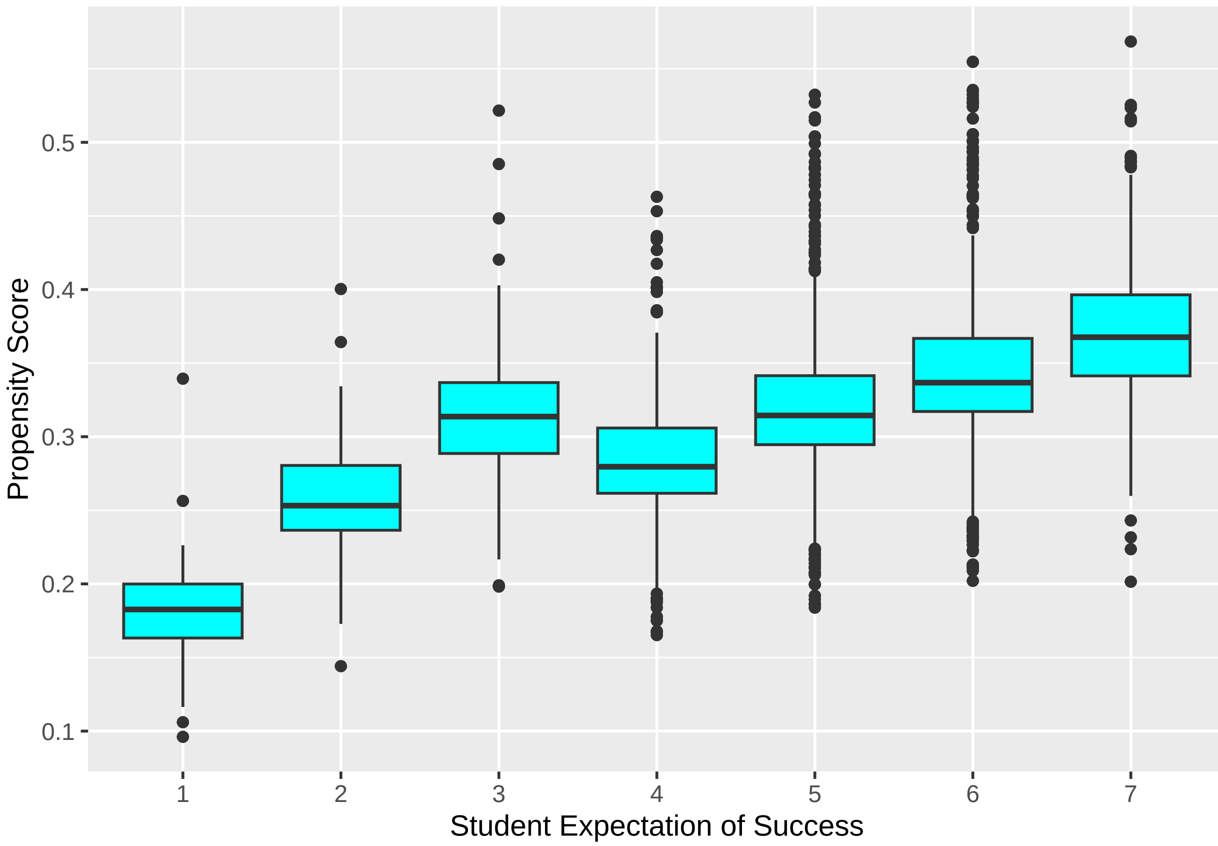
```r
cat("Estimated ATE:", ate_OR(train,test), "\n", "95% Confidence Interval:", confidence_interval, "\n",
    "Standard Error:", OR_SE)
```

```
## Estimated ATE: 0.278849
##  95% Confidence Interval: 0.2420477 0.3161149
##  Standard Error: 0.01867229
```

```r
#Propensity Score
compute_propensity <- function(train_data, test_data){
    ps_model <- glm(formula = Intervention ~ StudentExpectation + Race + Gender
                    + FirstGen + Urbanicity + FixedMindsets + AchievementLevels +
                      SchoolMinorityPercent + SchoolPovertyPercent + SchoolSize,
                    family = binomial(link = "logit"), data = train_data)

  #train_data$ps <- predict(ps_model, new_data = train_data, type = "response")
  test_data$ps <- predict(ps_model, newdata = test_data, type = "response")
return(test_data$ps)
}

test$ps <- compute_propensity(train,test)
# grouped boxplot
ggplot(test, aes(x=StudentExpectation, y=ps)) +
  geom_boxplot(fill = "cyan") +
  labs(x = "Student Expectation of Success", y = "Propensity Score")
```



Inverse Weighting Approach:

```
# IPW estimator
test$Intervention <- as.numeric(test$Intervention)
compute_IPW_ATE <- function(train_data, test_data){
  ps_model <-  glm(formula = Intervention ~ StudentExpectation + Race + Gender
                   + FirstGen + Urbanicity + FixedMindsets + AchievementLevels +
                     SchoolMinorityPercent + SchoolPovertyPercent + SchoolSize,
                   family = binomial(link = "logit"), data = train_data)

  test_data$ps <- predict(ps_model, newdata = test_data, type = "response")
  m1_ipw <- mean((test_data$Intervention/ test_data$ps) * test_data$Outcome)
  m0_ipw <- mean(((1 - test_data$Intervention)/(1 - test_data$ps))*test_data$Outcome)
  return(m1_ipw - m0_ipw)
}

compute_IPW_ATE(train,test)
```

```
## [1] -0.3347782
```

```
set.seed(123)
bootstrap_ipw <- function(train_data, test_data, n_bootstrap = 5000){
  results <- numeric(n_bootstrap)
  for (i in 1:n_bootstrap){
```

```
    bootstrap_sample <- train_data[sample(nrow(train_data), replace = TRUE), ]
    results[i] <- compute_IPW_ATE(bootstrap_sample, test_data)
  }
  return(results)
}

bootstrap_results_ipw <- bootstrap_ipw(train, test, n_bootstrap)
IPW_SE <- sd(bootstrap_results_ipw)
IPW_SE
```

```
## [1] 0.03849028
```

```
#Confidence Intervals
IPW_CI <- quantile(bootstrap_results_ipw, c(0.025, 0.975))
IPW_CI
```

```
##       2.5%      97.5%
## -0.4308998 -0.2832002
```

Doubly Robust Estimation:

```
# DR estimator
test$Intervention <- as.numeric(test$Intervention)
compute_DR_ATE <- function(train_data, test_data){
 ps_model <-  glm(formula = Intervention ~ StudentExpectation + Race + Gender
                    + FirstGen + Urbanicity + FixedMindsets + AchievementLevels +
                      SchoolMinorityPercent + SchoolPovertyPercent + SchoolSize,
                    family = binomial(link = "logit"), data = train_data)
  model1 <- lm(Outcome ~ Race + Gender + FirstGen + Urbanicity + FixedMindsets + AchievementLevels + Scl
                           + SchoolPovertyPercent + SchoolSize, data = train_data[train_data$Interventi
  model0 <- lm(Outcome ~ Race + Gender + FirstGen + Urbanicity + FixedMindsets + AchievementLevels + Scl
                           + SchoolPovertyPercent + SchoolSize, data = train_data[train_data$Interventi

    mu1 <- predict(model1, newdata = test_data, type = "response")
    mu0 <- predict(model0, newdata = test_data, type = "response")
    OR_est <- mean(mu1 - mu0)

  test_data$ps <- predict(ps_model, newdata = test_data, type = "response")
  M1 <- mean(test_data$Intervention*(test_data$Outcome - mu1)/test_data$ps)
 M2 <- mean(((1 - test_data$Intervention)* (test_data$Outcome - mu0))/(1 - test_data$ps))
return(OR_est+ M1 - M2)
}
DR_ATE <- compute_DR_ATE(train, test)
DR_ATE
```

```
## [1] -0.3415459
```

```
bootstrap_dr <- function(train_data, test_data, n_bootstrap = 1000){
  results <- numeric(n_bootstrap)
  for (i in 1:n_bootstrap){
```

```
    bootstrap_sample <- train_data[sample(nrow(train_data), replace = TRUE), ]
    results[i] <- compute_DR_ATE(bootstrap_sample, test_data)
  }
  return(results)
}
set.seed(123)
bootstrap_results_dr <- bootstrap_dr(train, test, n_bootstrap)
DR_SE <- sd(bootstrap_results_dr)
DR_SE
```

```
## [1] 0.07402885
```

```
#Confidence Intervals
DR_CI <- quantile(bootstrap_results_dr, c(0.025, 0.975))
DR_CI
```

```
##       2.5%      97.5%
## -0.5150961 -0.2147740
```

Hajek Estimator:

```
hajek_func <- function(data1, data2) {
  hajek_model <- glm(formula = Intervention ~ StudentExpectation + Race + Gender
                    + FirstGen + Urbanicity + FixedMindsets + AchievementLevels +
                      SchoolMinorityPercent + SchoolPovertyPercent + SchoolSize,
                    family = binomial(link = "logit"), data = data1)
  data1$hajek <- predict(hajek_model, type = "response")
  data2$hajek <- predict(hajek_model, newdata = data2, type = "response")
  mu1_hajek <- mean((data2$Intervention/ data2$hajek)/mean((data2$Intervention/ data2$hajek)) * data2$Ou
  mu0_hajek <- mean(( (1 - data2$Intervention)/ (1-data2$hajek))/mean((1 - data2$Intervention)/ (1-data2
  ATE_x_hajek = mu1_hajek - mu0_hajek
  return(ATE_x_hajek)
}
ATE_x_hajek <- hajek_func(train, test)
print(paste("ATE for Hajek estimator is:", ATE_x_hajek))
```

```
## [1] "ATE for Hajek estimator is: -0.211971949529218"
```

```
suppressWarnings (bootstrap_hajek <- replicate(n_bootstrap , {sample_boot <- train[sample(nrow(train), r
                    hajek_func(sample_boot, test)}))
hajek_ci <- quantile(bootstrap_hajek, c(0.025, 0.975))
hajek_ci
```

```
##       2.5%      97.5%
## -0.2396494 -0.1906684
```

```
hajek_se <- sd(bootstrap_hajek)
print(paste("Standard errors for Hajek estimator is:",hajek_se))
```

```
## [1] "Standard errors for Hajek estimator is: 0.0206401801665278"
```

Sensitivity Analysis

```
eta0 <- c(1/2, 1/1.7, 1/1.5, 1/1.3, 1, 1.3, 1.5, 1.7, 2)
eta1 <- c(1/2, 1/1.7, 1/1.5, 1/1.3, 1, 1.3, 1.5, 1.7, 2)
ATE_calculation <- function(train_data, test_data, mu1, mu0, eta0_val, eta1_val) {
  test_data$Intervention <- as.numeric(test_data$Intervention)
  treatment <- mean(test_data$Intervention * mu1 + (1 - test_data$Intervention)*(mu1/eta1_val))
  control <- mean(test_data$Intervention * mu0 * eta0_val + (1 - test_data$Intervention)*mu0)
  ATE_val <- treatment - control
  return(ATE_val)
}

sensitivity_analysis_ATE <- function(train_data, test_data, eta0_val, eta1_val){
  model1 <- glm(Outcome ~ Race + Gender + FirstGen + Urbanicity + FixedMindsets + AchievementLevels + Sc
                + SchoolPovertyPercent + SchoolSize, family = gaussian, data = train_data[train_data$Inte
  model0 <- glm(Outcome ~ Race + Gender + FirstGen + Urbanicity + FixedMindsets + AchievementLevels + Sc
                + SchoolPovertyPercent + SchoolSize, family = gaussian, data = train_data[train_data$Inte
  mu1_sen <- mean(as.matrix(predict(model1, newdata = test_data, type = "response")))
  mu0_sen <- mean(as.matrix(predict(model0, newdata = test_data, type = "response")))
  ATE_vals <- matrix(NA, length(eta0_val), length(eta1_val), dimnames = list(eta0_val, eta1_val))


  for (i in 1:length(eta0)) {
    for (j in 1:length(eta1)) {
      ATE_vals[i, j] <- ATE_calculation(train_data, test_data, mu1_sen, mu0_sen, eta0_val[i], eta1_val[j
    }
  }
  return(ATE_vals)
}
suppressWarnings(ATE_values <- sensitivity_analysis_ATE(train, test, eta0, eta1))
#View(ATE_values)
```

```
n_bootstrap <- 1000
eta0_str <- as.character(eta0)
eta1_str <- as.character(eta1)

bootstrap_results <- array(dim = c(length(eta0), length(eta1), n_bootstrap),
                           dimnames = list(eta0_val = eta0_str, eta1_val = eta1_str, Sample = NULL))
set.seed(123)
suppressWarnings(for (b in 1:n_bootstrap) {
  sample_boot <- train[sample(nrow(train), replace = TRUE), ]
  ATE_vals_boot <- sensitivity_analysis_ATE(sample_boot, test, eta0, eta1)
  bootstrap_results[,,b] <- ATE_vals_boot
})
se_matrix <- apply(bootstrap_results, c(1, 2), sd)
#View(se_matrix)

library(knitr)
library(kableExtra)
```

```
##
## Attaching package: 'kableExtra'
```

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

```r
kable(ATE_values, caption = "ATE estimates") %>%
  kable_styling(bootstrap_options = c("striped", "scale_down"))
```

Table 1: ATE estimates

	0.5	0.588235294117647	0.666666666666667	0.769230769230769	1	1.3
0.5	0.1277089	0.1373063	0.1437045	0.1501028	0.1597001	0.1670827
0.588235294117647	0.1487352	0.1583326	0.1647308	0.1711290	0.1807264	0.1881089
0.666666666666667	0.1674252	0.1770226	0.1834208	0.1898190	0.1994164	0.2067990
0.769230769230769	0.1918660	0.2014634	0.2078616	0.2142598	0.2238572	0.2312398
1	0.2468578	0.2564551	0.2628534	0.2692516	0.2788490	0.2862315
1.3	0.3183471	0.3279444	0.3343427	0.3407409	0.3503383	0.3577208
1.5	0.3660066	0.3756040	0.3820022	0.3884005	0.3979978	0.4053804
1.7	0.4136662	0.4232635	0.4296618	0.4360600	0.4456573	0.4530399
2	0.4851555	0.4947528	0.5011511	0.5075493	0.5171466	0.5245292

```r
kable(se_matrix, caption = "Bootstrapped Standard Errors") %>%
  kable_styling(bootstrap_options = c("striped", "scale_down"))
```

Table 2: Bootstrapped Standard Errors

	0.5	0.588235294117647	0.666666666666667	0.769230769230769	1	1.3
0.5	0.0111221	0.0125688	0.0135408	0.0145174	0.0159889	0.0171253
0.588235294117647	0.0115551	0.0129577	0.0139051	0.0148601	0.0163041	0.0174222
0.666666666666667	0.0120206	0.0133780	0.0142998	0.0152321	0.0166467	0.0177454
0.769230769230769	0.0127291	0.0140225	0.0149072	0.0158065	0.0171774	0.0182467
1	0.0146522	0.0157977	0.0165937	0.0174113	0.0186723	0.0196660
1.3	0.0176202	0.0185935	0.0192806	0.0199948	0.0211105	0.0220007
1.5	0.0197862	0.0206635	0.0212877	0.0219403	0.0229670	0.0237917
1.7	0.0220510	0.0228468	0.0234164	0.0240145	0.0249609	0.0257254
2	0.0255758	0.0262720	0.0267733	0.0273024	0.0281449	0.0288299

Causal Forests:

From our dataset, notice that all of the observations are pooled from uneven clusters based on school ID. Thus, this will change our inferential approach as in finding an optimal way to quantify the causal effects accurately given the information. From the paper, for example, in our setting, do we want to fit a model that accurately reflects heterogeneity in our available sample of $J = 76$ schools, or a model that will generalize to students from other schools also? Should we give more weight in our analysis to schools from which we observe more students? The approach they choose in the paper is to assume that we want a predictive model that generalizes to more than J schools with equal weights to any new school added to the dataset. In other words, we want a predictive model that can predict the causal effect when we add a new observation from a new school to the data.

```r
library(grf)
```

```r
data = read.csv("synthetic_data.csv")
data$schoolid = factor(data$schoolid)

names(data) <- c("SchoolID", "Intervention", "Outcome", "StudentExpectation","Race","Gender","FirstGen"
                 "Urbanicity","FixedMindsets","AchievementLevels","SchoolMinorityPercent","SchoolPovert

DF = data[,-1]
school.id = as.numeric(data$SchoolID)

school.mat = model.matrix(~ SchoolID + 0, data = data)
school.size = colSums(school.mat)

# It appears that school ID does not affect pscore. So ignore it in modeling, and just treat it as sour
w.lm = glm(Intervention ~ ., data = data[,-3], family = binomial)
summary(w.lm)
```

```
##
## Call:
## glm(formula = Intervention ~ ., family = binomial, data = data[,
##     -3])
##
## Coefficients: (6 not defined because of singularities)
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)            -0.9524636  0.2845173  -3.348 0.000815 ***
## SchoolID2               0.0697302  0.2766287   0.252 0.800986
## SchoolID3               0.0382080  0.2911323   0.131 0.895586
## SchoolID4               0.1761334  0.2784711   0.633 0.527059
## SchoolID5              -0.0033389  0.2950180  -0.011 0.990970
## SchoolID6               0.0583548  0.3067481   0.190 0.849124
## SchoolID7              -0.1313759  0.3188190  -0.412 0.680288
## SchoolID8               0.1233661  0.3023736   0.408 0.683279
## SchoolID9              -0.1955428  0.3073344  -0.636 0.524611
## SchoolID10             -0.1892794  0.2968750  -0.638 0.523752
## SchoolID11             -0.2224060  0.5461005  -0.407 0.683816
## SchoolID12             -0.3312420  0.5414374  -0.612 0.540682
## SchoolID13             -0.0408540  0.3989507  -0.102 0.918436
## SchoolID14             -0.8681934  0.6033674  -1.439 0.150175
## SchoolID15             -0.1059135  0.3263162  -0.325 0.745504
## SchoolID16             -0.1063268  0.2885387  -0.369 0.712500
## SchoolID17              0.0854323  0.3119435   0.274 0.784184
## SchoolID18             -0.1924441  0.2997822  -0.642 0.520908
## SchoolID19             -0.0265326  0.3229712  -0.082 0.934526
## SchoolID20             -0.2179554  0.3041336  -0.717 0.473594
## SchoolID21             -0.2147440  0.2982822  -0.720 0.471565
## SchoolID22             -0.5115966  0.4410779  -1.160 0.246098
## SchoolID23              0.0039231  0.3475373   0.011 0.990994
## SchoolID24             -0.0848314  0.3259572  -0.260 0.794668
## SchoolID25              0.0521087  0.2754586   0.189 0.849959
## SchoolID26              0.0241212  0.2876511   0.084 0.933171
## SchoolID27             -0.2300630  0.3104796  -0.741 0.458698
## SchoolID28             -0.3519010  0.2924774  -1.203 0.228909
```

```
## SchoolID29          -0.2198764  0.3293288  -0.668 0.504357
## SchoolID30          -0.3146292  0.3257994  -0.966 0.334187
## SchoolID31           0.1398555  0.6137901   0.228 0.819759
## SchoolID32           0.1555524  0.3916156   0.397 0.691215
## SchoolID33          -0.0991693  0.3939370  -0.252 0.801243
## SchoolID34          -0.0073688  0.2980808  -0.025 0.980278
## SchoolID35          -0.3528987  0.3997273  -0.883 0.377318
## SchoolID36          -0.3751465  0.3988972  -0.940 0.346982
## SchoolID37          -0.0343169  0.3219646  -0.107 0.915117
## SchoolID38          -0.1346432  0.3851869  -0.350 0.726674
## SchoolID39          -0.4339936  0.3612869  -1.201 0.229657
## SchoolID40          -0.3993958  0.3834495  -1.042 0.297604
## SchoolID41          -0.1490784  0.3542105  -0.421 0.673846
## SchoolID42          -0.1545715  0.3551857  -0.435 0.663428
## SchoolID43          -0.5679567  0.4277455  -1.328 0.184247
## SchoolID44          -0.1425896  0.3774795  -0.378 0.705623
## SchoolID45          -0.1337888  0.3232493  -0.414 0.678957
## SchoolID46          -0.2573249  0.3129119  -0.822 0.410874
## SchoolID47           0.0027726  0.2770108   0.010 0.992014
## SchoolID48          -0.3406079  0.3470361  -0.981 0.326358
## SchoolID49          -0.3236117  0.3434541  -0.942 0.346077
## SchoolID50          -0.1185119  0.4086074  -0.290 0.771787
## SchoolID51           0.4087898  0.4506822   0.907 0.364382
## SchoolID52          -0.3144014  0.4118342  -0.763 0.445214
## SchoolID53          -0.2733677  0.4511280  -0.606 0.544538
## SchoolID54          -0.0889588  0.3872532  -0.230 0.818311
## SchoolID55          -0.1558106  0.4155020  -0.375 0.707665
## SchoolID56           0.1050353  0.3149235   0.334 0.738737
## SchoolID57          -0.0314901  0.2901719  -0.109 0.913581
## SchoolID58          -0.0383183  0.2730077  -0.140 0.888379
## SchoolID59          -0.0529637  0.2934895  -0.180 0.856790
## SchoolID60          -0.1624792  0.3972885  -0.409 0.682561
## SchoolID61          -0.0289549  0.3201953  -0.090 0.927946
## SchoolID62           0.0993158  0.2669678   0.372 0.709882
## SchoolID63           0.1684702  0.3282167   0.513 0.607749
## SchoolID64          -0.0693060  0.2770896  -0.250 0.802493
## SchoolID65          -0.0004197  0.4072922  -0.001 0.999178
## SchoolID66          -0.2130911  0.2984091  -0.714 0.475171
## SchoolID67           0.0358440  0.2921158   0.123 0.902341
## SchoolID68          -0.0871303  0.3290814  -0.265 0.791188
## SchoolID69          -0.2550387  0.2908992  -0.877 0.380636
## SchoolID70          -0.0268947  0.4032160  -0.067 0.946820
## SchoolID71           0.0037464  0.4268290   0.009 0.992997
## SchoolID72          -0.1304085  0.2881512  -0.453 0.650859
## SchoolID73          -0.2160697  0.2840030  -0.761 0.446776
## SchoolID74          -0.0935320  0.2842612  -0.329 0.742129
## SchoolID75          -0.1056241  0.3024204  -0.349 0.726892
## SchoolID76          -0.1052261  0.2939262  -0.358 0.720342
## StudentExpectation   0.1036077  0.0197345   5.250 1.52e-07 ***
## Race                -0.0015919  0.0053900  -0.295 0.767728
## Gender              -0.1038596  0.0424020  -2.449 0.014309 *
## FirstGen            -0.1319218  0.0461833  -2.856 0.004284 **
## Urbanicity                  NA         NA      NA       NA
## FixedMindsets               NA         NA      NA       NA
```

```
## AchievementLevels             NA         NA      NA       NA
## SchoolMinorityPercent         NA         NA      NA       NA
## SchoolPovertyPercent          NA         NA      NA       NA
## SchoolSize                    NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 13115  on 10390  degrees of freedom
## Residual deviance: 13009  on 10311  degrees of freedom
## AIC: 13169
##
## Number of Fisher Scoring iterations: 4
```

```r
W = DF$Intervention
Y = DF$Outcome
X.raw = DF[,-(1:2)]

Race.exp = model.matrix(~ factor(X.raw$Race) + 0)
Urbancity.exp = model.matrix(~ factor(X.raw$Urbanicity) + 0)

X = cbind(X.raw[,-which(names(X.raw) %in% c("Race", "Urbancity"))], Race.exp, Urbancity.exp)

#
# Grow a forest. Add extra trees for the causal forest.
# Y: outcome (Outcome), W: treatment (Intervention)

Y.forest = regression_forest(X, Y, clusters = school.id, equalize.cluster.weights = TRUE)
Y.hat = predict(Y.forest)$predictions
W.forest = regression_forest(X, W, clusters = school.id, equalize.cluster.weights = TRUE)
W.hat = predict(W.forest)$predictions

cf.raw = causal_forest(X, Y, W,
                       Y.hat = Y.hat, W.hat = W.hat,
                       clusters = school.id,
                       equalize.cluster.weights = TRUE)
varimp = variable_importance(cf.raw)
selected.idx = which(varimp > mean(varimp))
varimp
```

```
##               [,1]
##  [1,] 0.1325341354
##  [2,] 0.0363396990
##  [3,] 0.0314975582
##  [4,] 0.0375087018
##  [5,] 0.1592282098
##  [6,] 0.1218523419
##  [7,] 0.0993921531
##  [8,] 0.1083040050
##  [9,] 0.1124552250
## [10,] 0.0155414239
## [11,] 0.0175869456
## [12,] 0.0000000000
```

```
## [13,] 0.0414399149
## [14,] 0.0011291143
## [15,] 0.0000000000
## [16,] 0.0000000000
## [17,] 0.0000000000
## [18,] 0.0000000000
## [19,] 0.0000000000
## [20,] 0.0000000000
## [21,] 0.0017124521
## [22,] 0.0000000000
## [23,] 0.0129009937
## [24,] 0.0007521587
## [25,] 0.0055631918
## [26,] 0.0150802038
## [27,] 0.0179558106
## [28,] 0.0260828633
## [29,] 0.0051428981
```

```r
cf = causal_forest(X[,selected.idx], Y, W,
                   Y.hat = Y.hat, W.hat = W.hat,
                   clusters = school.id,
                   equalize.cluster.weights = TRUE,
                   tune.parameters = "all")
```

```r
# Tau is average treatment effect
tau.hat = predict(cf)$predictions
#
# Estimate ATE
#
ATE = average_treatment_effect(cf)
ATE
```

```
##   estimate    std.err
## 0.24848322 0.02030945
```

```r
paste("95% CI for the ATE:", round(ATE[1], 3),
      "+/-", round(qnorm(0.975) * ATE[2], 3))
```

```
## [1] "95% CI for the ATE: 0.248 +/- 0.04"
```

```r
print(paste("Confidence interval for ATE is (0.208, 0.286)"))
```

```
## [1] "Confidence interval for ATE is (0.208, 0.286)"
```

```r
test_calibration(cf)
```

```
##
## Best linear fit using forest predictions (on held-out data)
## as well as the mean forest prediction as regressors, along
## with one-sided heteroskedasticity-robust (HC3) SEs:
##
```

```
##                               Estimate Std. Error t value Pr(>t)
## mean.forest.prediction         0.998741   0.082805  12.061 <2e-16 ***
## differential.forest.prediction 0.385498   0.639263   0.603 0.2732
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

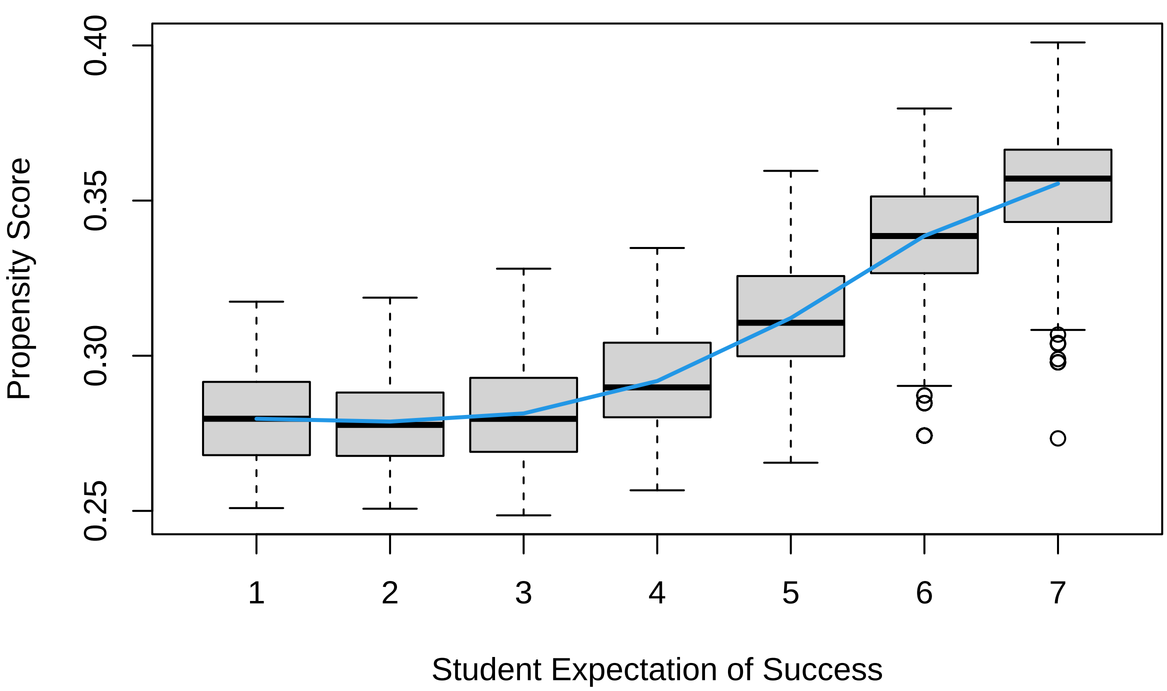
```
#
# Look at variation in propensity scores
#

DF = X
DF$W.hat = cf$W.hat

pdf("pscore.pdf")
pardef = par(mar = c(5, 4, 4, 2) + 0.5, cex.lab=1.5, cex.axis=1.5, cex.main=1.5, cex.sub=1.5)
boxplot(W.hat ~ StudentExpectation, data = DF, ylab = "Propensity Score", xlab = "Student Expectation o
lines(smooth.spline(X$StudentExpectation, cf$W.hat), lwd = 2, col = 4)
dev.off()
```

```
## pdf
##   2
```

```
boxplot(W.hat ~ StudentExpectation, data = DF, ylab = "Propensity Score", xlab = "Student Expectation o
lines(smooth.spline(X$StudentExpectation, cf$W.hat), lwd = 2, col = 4)
```




```
#
# Make some plots...
#

pdf("tauhat_hist.pdf")
pardef = par(mar = c(5, 4, 4, 2) + 0.5, cex.lab=1.5, cex.axis=1.5, cex.main=1.5, cex.sub=1.5)
```

```r
hist(tau.hat, xlab = "estimated CATE", main = "")
dev.off()
```

```
## pdf
##   2
```

Causal Forest non-clustering-robustness:

```r
#
# Analysis ignoring clusters
#

cf.noclust = causal_forest(X[,selected.idx], Y, W,
                           Y.hat = Y.hat, W.hat = W.hat,
                           tune.parameters = "all")

ATE.noclust = average_treatment_effect(cf.noclust)
ATE.noclust
```

```
##   estimate    std.err
## 0.25383522 0.01144392
```

```r
paste("95% CI for the ATE:", round(ATE.noclust[1], 3),
      "+/-", round(qnorm(0.975) * ATE.noclust[2], 3))
```

```
## [1] "95% CI for the ATE: 0.254 +/- 0.022"
```

```r
test_calibration(cf.noclust)
```

```
##
## Best linear fit using forest predictions (on held-out data)
## as well as the mean forest prediction as regressors, along
## with one-sided heteroskedasticity-robust (HC3) SEs:
##
##                                Estimate Std. Error t value    Pr(>t)
## mean.forest.prediction         1.008263   0.044874 22.4687 < 2.2e-16 ***
## differential.forest.prediction 0.506361   0.122926  4.1192 1.915e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Comparing the test_calibration
test_calibration(cf)
```

```
##
## Best linear fit using forest predictions (on held-out data)
## as well as the mean forest prediction as regressors, along
## with one-sided heteroskedasticity-robust (HC3) SEs:
##
##                                Estimate Std. Error t value Pr(>t)
```

```
## mean.forest.prediction         0.998741   0.082805  12.061 <2e-16 ***
## differential.forest.prediction 0.385498   0.639263   0.603 0.2732
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
test_calibration(cf.noclust)
```

```
##
## Best linear fit using forest predictions (on held-out data)
## as well as the mean forest prediction as regressors, along
## with one-sided heteroskedasticity-robust (HC3) SEs:
##
##                                Estimate Std. Error t value    Pr(>t)
## mean.forest.prediction         1.008263   0.044874 22.4687 < 2.2e-16 ***
## differential.forest.prediction 0.506361   0.122926  4.1192 1.915e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```