

# Data Analysis 2

Ziying Li, Matt Capaldi, Yujuan Gao, Olivia Morales

October 25, 2022

```
## libraries
libs <- c("tidyverse", "haven", "bibtex", "psych", "knitr", "pastecs", "kableExtra", "survey", "cobalt",
         "baguette", "parsnip", "SimDesign", "randomForest", "bartCause", "lme4", "grf", "GenericML", "car")
sapply(libs, require, character.only = TRUE)
```

##	tidyverse	haven	bibtex	psych	knitr	pastecs
##	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
##	kableExtra	survey	cobalt	randomForest	ipred	rpart
##	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
##	baguette	parsnip	SimDesign	bartCause	lme4	grf
##	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
##	GenericML	car				
##	TRUE	TRUE				

```
covariateNames <- c(
  "X1RTHETK1",
  "X1MTHETK1",
  "X1TCHAPP",
  "X1TCHCON",
  "X1TCHPER",
  "X1TCHEXT",
  "X1TCHINT",
  "X1ATTNFS",
  "X1INBCNT",
  "X12MOMAR",
  "X1NUMSIB",
  "P1OLDMOM",
  "P1CHLDBK",
  "P2DISTHM",
  "P1NUMPLA",
  "T2PARIN",
  "X12PAR1ED_I",
  "X12PAR2ED_I",
  "X2INCCAT_I",
  "X1PAR1EMP",
  "S2LUNCH",
  "X2KRCETH",
  "S2NGHBOR",
  "S2OUTSID",
  "S2USDABR",
  "S2PUBSOC",
)
```

```

    "X1LOCALE",
    "S1_ID",
    "W1_2POPSU",
    "prop.missing")

covariateNamesBART <- c(
  "X1RTHETK1",
  "X1MTHETK1",
  "X1TCHAPP",
  "X1TCHCON",
  "X1TCHPER",
  "X1TCHEXT",
  "X1TCHINT",
  "X1ATTNFS",
  "X1INBCNT",
  "X12MOMAR",
  "X1NUMSIB",
  "P1OLDMOM",
  "P1CHLDBK",
  "P2DISTHM",
  "P1NUMPLA",
  "T2PARIN",
  "X12PAR1ED_I",
  "X12PAR2ED_I",
  "X2INCCAT_I",
  "X1PAR1EMP",
  "S2LUNCH",
  "X2KRCETH",
  "S2NGHBOR",
  "S2OUTSID",
  "S2USDABR",
  "S2PUBSOC",
  "X1LOCALE")

## directory path (assignment_2 as current working directory)

data_dir <- file.path(".", "data")

## loading data
load(file.path(data_dir, "chapter_10_data_cleaned_and_imputed.Rdata"))

```

In this paper, we explore heterogeneity of treatment using three different methods; CausalBART, GenericML, and CausalForest. We will explore the heterogeneity (answer the assignment questions) separately for each model before concluding and comparing the findings at the conclusion of the paper.

```

psFormula <- paste(covariateNames, collapse="+")
psFormula <- formula(paste("treated~", psFormula, sep=""))
print(psFormula)

## treated ~ X1RTHETK1 + X1MTHETK1 + X1TCHAPP + X1TCHCON + X1TCHPER +
##          X1TCHEXT + X1TCHINT + X1ATTNFS + X1INBCNT + X12MOMAR + X1NUMSIB +
##          P1OLDMOM + P1CHLDBK + P2DISTHM + P1NUMPLA + T2PARIN + X12PAR1ED_I +
##          X12PAR2ED_I + X2INCCAT_I + X1PAR1EMP + S2LUNCH + X2KRCETH +

```

```
##      S2NGHBOR + S2OUTSID + S2USDABR + S2PUBSOC + X1LOCALE + S1_ID +
##      W1_2POPSU + prop.missing
```

## BART

```
# credit to Matt, esp for the covs matrices!
train <- data %>%
  sample_frac(size = 0.5)
test <- anti_join(data, train)

matrix_covsBART <- as.matrix(train %>% select(covariateNamesBART) %>%
  mutate(across(.fns = as.numeric)))

matrix_covsBART_2 <- as.matrix(test %>% select(covariateNamesBART) %>%
  mutate(across(.fns = as.numeric)))

# estimating conditional average treatment effects (CATEs) using BART

bart <- bartc(response = train$X2MTHETK1,
  treatment = as.numeric(as.character(train$treated)),
  confounders = matrix_covsBART,
  method.rsp = "bart",
  method.trt = "glm",
  keepTrees = TRUE,
  estimand = "ate")
```

```
## fitting treatment model via method 'glm'
## fitting response model via method 'bart'
```

```
cate <- predict(bart,
  newdata = matrix_covsBART_2,
  type = "icate")

cate_m <- apply(cate, 2, mean)

library(bartMachine)

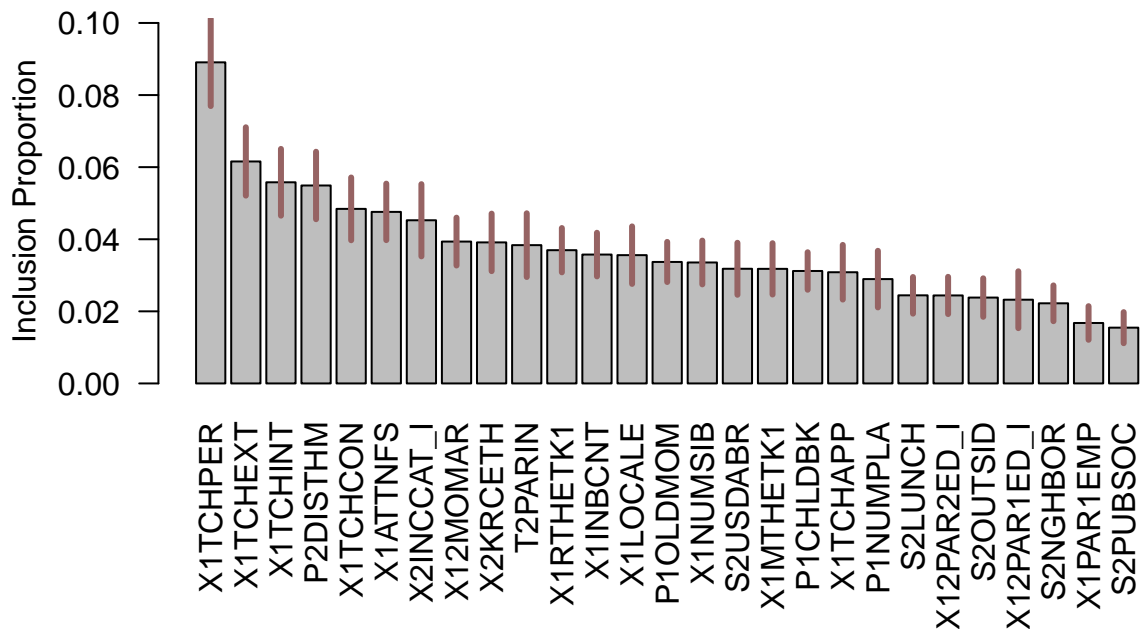
bart_machine <- bartMachine(X=as.data.frame(matrix_covsBART),
  y=cate_m,
  serialize = T,
  mem_cache_for_speed = F)
```

```
## bartMachine initializing with 50 trees...
## bartMachine vars checked...
## bartMachine java init...
## bartMachine factors created...
## bartMachine before preprocess...
## bartMachine after preprocess... 27 total features...
## bartMachine sigsq estimated...
```

```
## bartMachine training data finalized...
## Now building bartMachine for regression...
## evaluating in sample data...done
## serializing in order to be saved for future R sessions...done
```

```
var.importance <- investigate_var_importance(bart_machine,
                                              num_replicates_for_avg = 25)
```

```
## .....
```



```
#select variables whose importance is greater than the median
important.vars = names(var.importance$avg_var_props)[
  var.importance$avg_var_props > median(var.importance$avg_var_props)]

test <- test %>% mutate(across(.fns = as.numeric))
test$treated <- test$treated - 1

bart2 <- bartc(response = test$X2MTHETK1,
               treatment = test$treated,
               confounders = data.frame(test[,important.vars]),
               method.rsp = "bart",
               method.trt = "glm",
               estimand = "ate",
               keepTrees = T)
```

```
## fitting treatment model via method 'glm'
## fitting response model via method 'bart'
```

```
cate2 <- predict(bart2,
                 newdata = data.frame(test[,important.vars]),
                 type = "icate")

test$cate2 <- apply(cate2, 2, mean)

model <- paste(important.vars, collapse="+")
model <- paste(c("cate2",model), collapse="~")
model.cate <- lm(model,data = test)
summary(model.cate)
```

```
##
## Call:
## lm(formula = model, data = test)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.012596 -0.003997 -0.001413  0.001916  0.061475
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.472e-02  1.186e-03  12.414 < 2e-16 ***
## X1TCHPER     -1.535e-04  2.129e-04  -0.721 0.470911
## X1TCHEXT      1.406e-03  2.154e-04   6.527 7.22e-11 ***
## X1TCHINT      4.202e-03  2.031e-04  20.687 < 2e-16 ***
## P2DISTHM      2.245e-04  2.453e-05   9.153 < 2e-16 ***
## X1TCHCON      5.945e-04  2.305e-04   2.579 0.009941 **
## X1ATTNFS     -1.142e-03  1.126e-04 -10.149 < 2e-16 ***
## X2INCCAT_I   -8.057e-05  2.101e-05  -3.835 0.000127 ***
## X12MOMAR      2.230e-03  2.271e-04   9.820 < 2e-16 ***
## X2KRCETH      8.193e-05  3.428e-06  23.899 < 2e-16 ***
## T2PARIN       4.292e-03  1.308e-04  32.816 < 2e-16 ***
## X1RTHETK1    -4.359e-03  1.255e-04 -34.723 < 2e-16 ***
## X1INBCNT     -1.085e-03  1.269e-04  -8.552 < 2e-16 ***
## X1LOCALE     -3.826e-04  9.359e-05  -4.088 4.40e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.007523 on 6328 degrees of freedom
## Multiple R-squared:  0.5093, Adjusted R-squared:  0.5083
## F-statistic: 505.3 on 13 and 6328 DF,  p-value: < 2.2e-16
```

```
# correlation matrix
```

```
# cor(bart2, cate2, method = c("pearson", "kendall", "spearman"))
```

```
# qqplot
```

```
# plot1 <- ggplot(sample = bart2, data = chapter_10_data_cleaned_and_imputed.Rdata, color=cyl)+theme_bw()
```

## GenericML

```
##' @Matt
learners <- c("random_forest", "lasso")
matrix_covs <- as.matrix(data %>% select(all_of(covariateNames)) %>%
  mutate(across(.fns = as.numeric)))
X1 <- setup_X1(funs_Z = c("B", "S"))
  #fixed_effects = vil_pair)
vcov <- setup_vcov(estimator = "vcovHC")
  #arguments = list(cluster = demi_paire))

library(parsnip)
ps_rf <- rand_forest(mode = "classification",
  engine = "ranger",
  trees = 1000) %>%
  fit(psFormula,
    data = data)
data$ps_rf <- predict(ps_rf,
  new_data = data,
  type = "prob")[,2]
data$ps_rf <- data$ps_rf$.pred_1 ## remove the $column.name
data <- data %>%
  mutate(ps_rf = ifelse(ps_rf >= 0.95, 0.94, ps_rf), #Rounding to avoid error Dr. L
    ps_rf = ifelse(ps_rf <= 0.05, 0.55, ps_rf))

genML <- GenericML(
  Z = matrix_covs, #covariates
  D = as.numeric(as.character(data$treated)), #treatment
  Y = as.numeric(data$T2TTABS), #outcome
  learners_GenericML = learners, # learners specified above
  learner_propensity_score = as.numeric(data$ps_rf), #as.numeric(data$ps) #ps
  num_splits = 10, # number splits of the data
  quantile_cutoffs = c(0.2, 0.4, 0.6, 0.8), # grouping for CATEs
  significance_level = 0.05, # significance level
  X1_BLP = X1, X1_GATES = X1, # regression setup
  vcov_BLP = vcov, vcov_GATES = vcov, # covariance setup
  parallel = F, #num_cores = 6L, # parallelization
  seed = 20220621) # RNG seed

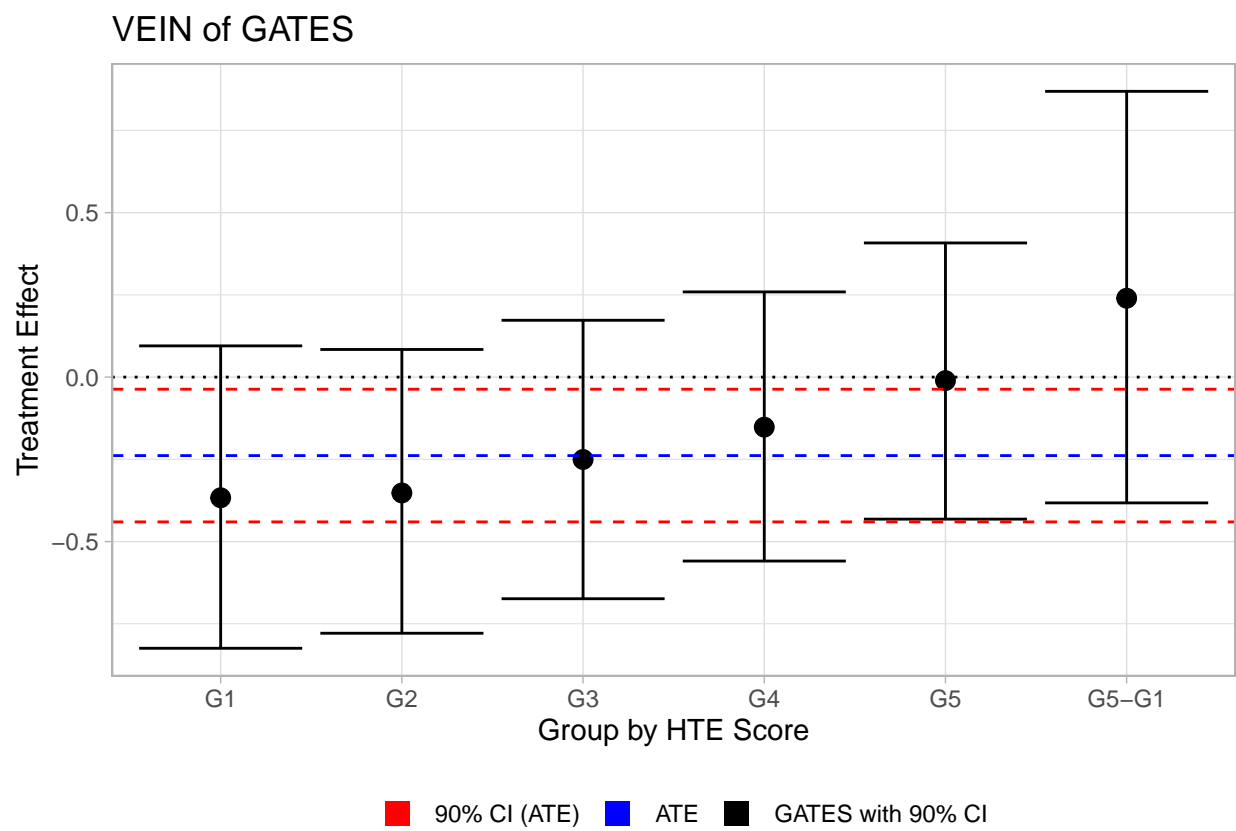
best <- get_best(genML)
## random_forest is best, becomes the default for all future GenML functions
base <- get_BLP(genML)
## significant indicating treatment heterogeneity
a <- get_GATES(genML) %>%
  plot()
# Plot parental education
b <- get_CLAN(genML,
  variable = "X12PAR1ED_I") %>%
  plot() +
  labs(title = "Heterogeneity by First Parents Education",
    y = str_wrap("Average Value of Parental Education", 25))
# Plot family marriage status
```

```

c <- get_CLAN(genML,
  variable = "X12MOMAR") %>%
  plot() +
  labs(title = "Heterogeneity by Parental Marriage Status",
    y = str_wrap("Average Value of Parental Marriage Status", 25))
# Plot lunch variable
d <- get_CLAN(genML,
  variable = "S2LUNCH") %>%
  plot() +
  labs(title = "Heterogeneity by Free School Lunch",
    y = str_wrap("Average Value of Students Receiving Free Lunch", 25))
# Plot percent coming from neighborhood
e <- get_CLAN(genML,
  variable = "S2NGHBOR") %>%
  plot() +
  labs(title = "Heterogeneity by Percent of School coming from
    Surrounding Neighborhood",
    y = str_wrap("Average Value of Percent Coming from Neighborhood", 25))

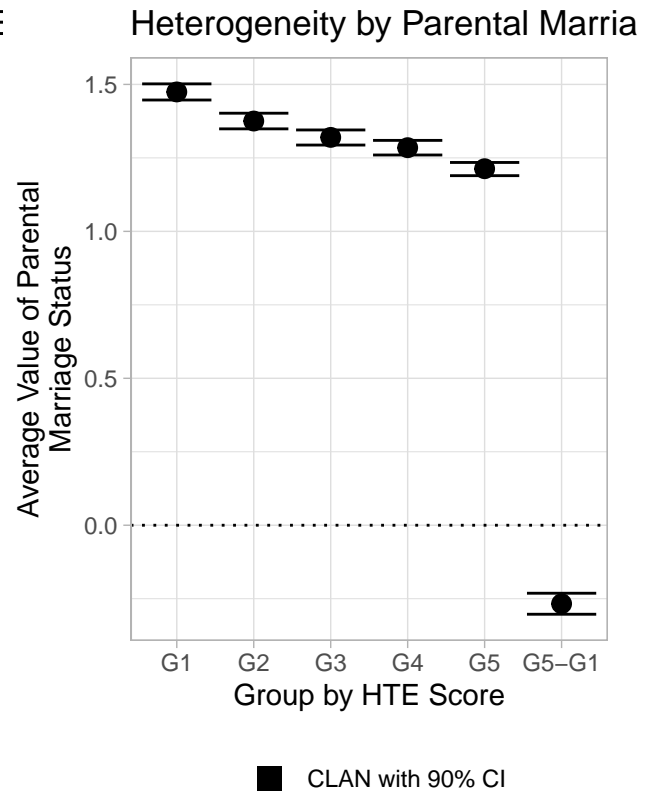
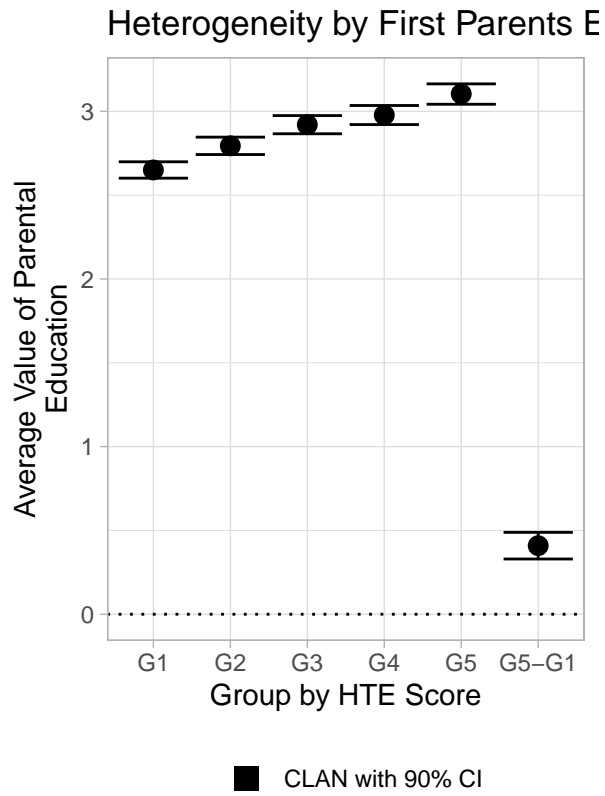
```

```
library(patchwork)
a
```

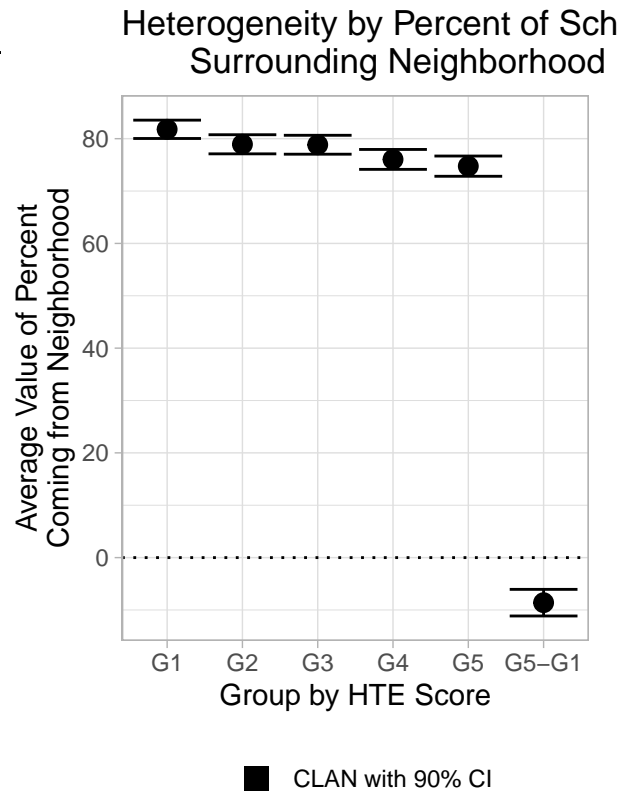
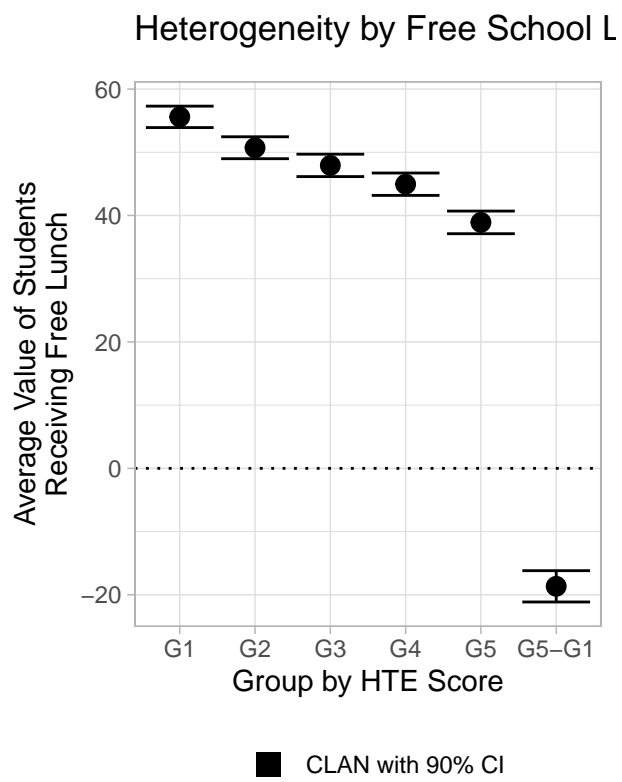


b+c





d+e



## Causal Forests

```
#the grf package only takes numeric covariates
data2 <- data
#So convert those factor variables to be the numeric class
for (i in 1:length(covariateNames)) {
  if(class(data2[,covariateNames[i]])=="factor"){
    data2[, covariateNames[i]] <- as.numeric(as.character(data2[,covariateNames[i]]))
  }
}

#Step 1: Split data into training data set and testing data set
#In this case, we split it to be 50/50
set.seed(123)
train_index <- sample(1:nrow(data2), nrow(data2)/2)
train_index <- train_index[order(train_index)]

train_data <- data2[train_index,]
test_data <- data2[-train_index,]

#Step 2: model fit, using causal forest
#Tuning mtry and min.node.size parameters by setting tune.parameters
train.forest = causal_forest(X=train_data[,covariateNames],
                             Y = train_data$X2MTHETK1, num.trees = 5000,
                             W = as.numeric(as.character(train_data$treated)),
                             W.hat = train_data$ps,
                             tune.parameters = c("mtry", "min.node.size"),
                             seed = 0)

train.forest[["tuning.output"]]
```

```
## Tuning status: tuned.
## This indicates tuning found parameters that are expected to perform better than default.
##
## Predicted debiased error: 0.192557529272496
##
## Tuned parameters:
## mtry: 1
## min.node.size: 1
##
## Average error by 5-quantile:
##
##      mtry      error
## [1,6] 0.1934612
## (6,11] 0.1934513
## (11,16] 0.1934947
## (16,21] 0.1935313
## (21,26] 0.1934668
##
## min.node.size      error
##      [1,2] 0.1926552
##      (2,9] 0.1931186
```

```
##      (9,32.4] 0.1935640
##      (32.4,124] 0.1939467
##      (124,393] 0.1941735
```

*#The results showed that mtry = 16 and min.node.size = 1 perform better than the default setting*

*#Step 3: Obtain estimates of the conditional average treatment effect (CATE)*

*#with standard errors*

```
tau.hat = predict(train.forest,X= test_data[,covariateNames], estimate.variance = T)
CATE_causalForest = tau.hat$predictions
```

*# Causal Forests*

*#1. correlation matrix*

*#causal forest only output the best tuning parameters' model fit outcomes*

*#To answer Q2, I run one more model fit with mtry = 4 and min.node.size = 50*

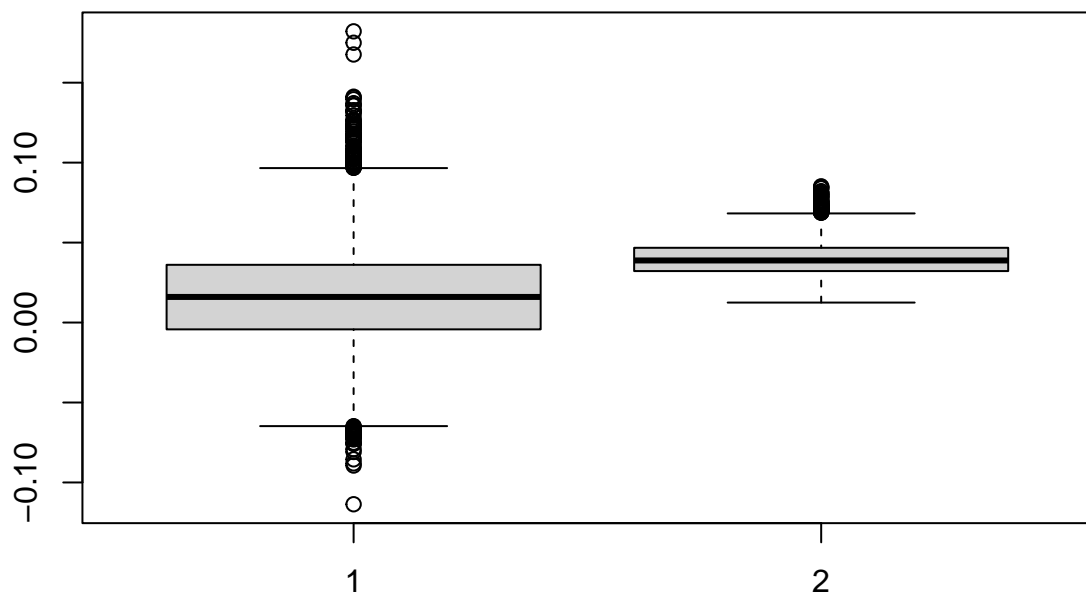
```
train.forest2 = causal_forest(X=train_data[,covariateNames],
                             Y = train_data$X2MTHETK1, num.trees = 5000,
                             W = as.numeric(as.character(train_data$treated)),
                             W.hat = train_data$ps,
                             mtry = 4, min.node.size = 50,
                             seed = 0)
tau.hat2 = predict(train.forest2,X= test_data[,covariateNames], estimate.variance = T)
CATE2_causalForest = tau.hat2$predictions

cor(CATE_causalForest, CATE2_causalForest)
```

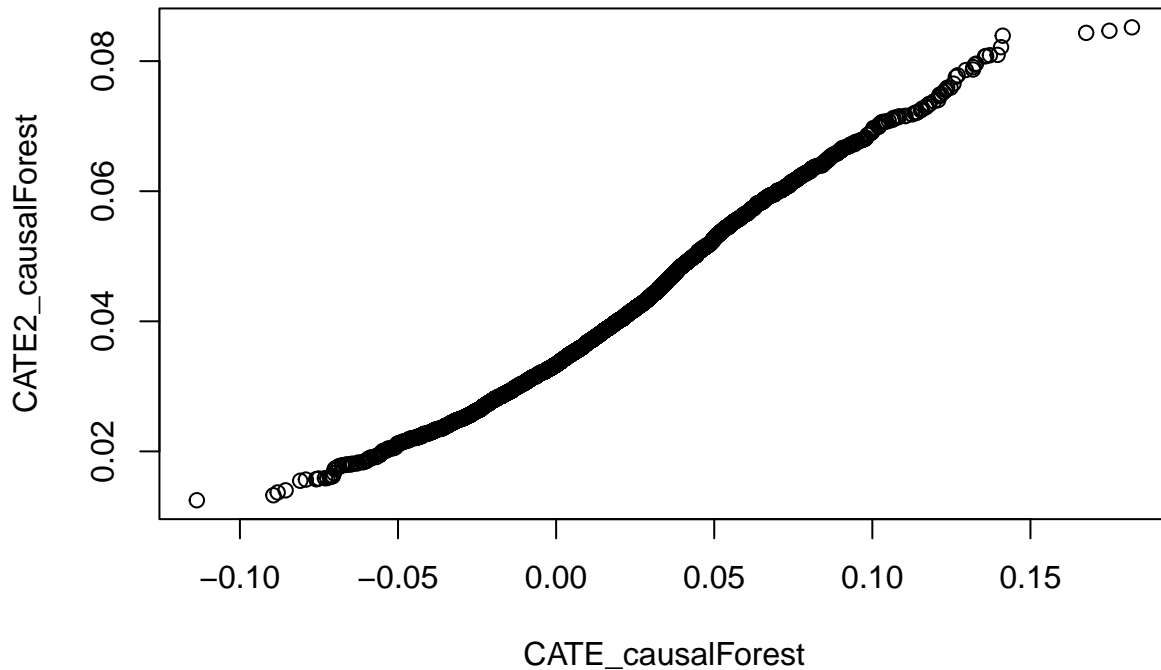
```
## [1] 0.5487647
```

*#2.box plot*

```
boxplot(CATE_causalForest, CATE2_causalForest)
```



```
#3.QQ plot  
qqplot(CATE_causalForest, CATE2_causalForest)
```



### Determine Best Linear Projection of CATE/Variable Importance

```
#Causal Forests
#Step 1: Subset important variables
importance_cf = variable_importance(train.forest)
rownames(importance_cf) = names(train_data[,covariateNames])

#select variables above the median of importance of the aggregated importances
#across imputed datasets
important.var_cf = rownames(importance_cf)[importance_cf>median(importance_cf)]

#Step 2
#run test forest
test.forest = causal_forest(X = test_data[,important.var_cf],
                           Y = test_data$X2MTHETK1,
                           W = as.numeric(as.character(test_data$treated)),
                           W.hat = test_data$ps,
                           mtry = 16, num.trees=5000,
                           min.node.size = 1, seed = 0)

#Step 3: Predict the conditional average treatment effect (CATE)
tau.hat = predict(test.forest,X= test_data[,important.var_cf], estimate.variance = T)
CATE_test = tau.hat$predictions
summary(CATE_test)
```

```
##      Min.    1st Qu.    Median      Mean   3rd Qu.      Max.
## -0.289183 -0.038450 -0.004933 -0.003709  0.028862  0.567660
```

*#When using the test data set and fitting the important variables,  
#the conditional ATE was small, which ranges from -0.401 to 0.254 with a mean of 0.018.*

```
test_calibration(test.forest)
```

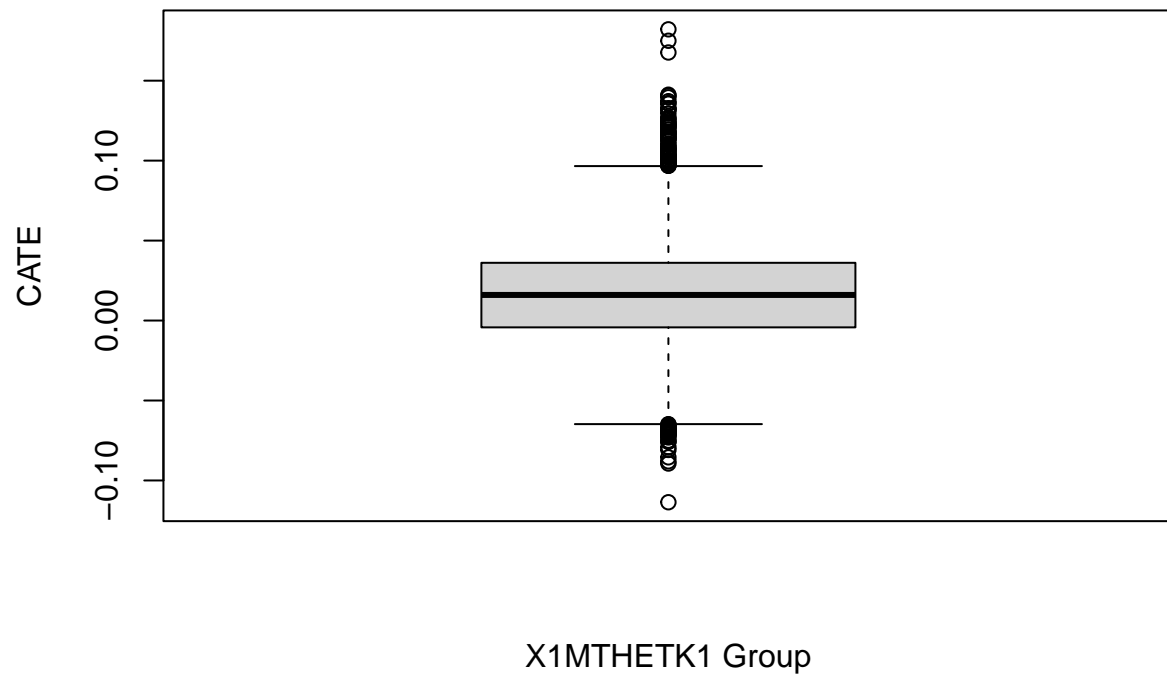
```
##
## Best linear fit using forest predictions (on held-out data)
## as well as the mean forest prediction as regressors, along
## with one-sided heteroskedasticity-robust (HC3) SEs:
##
##              Estimate Std. Error t value Pr(>t)
## mean.forest.prediction      1.18209    4.24958  0.2782 0.3904
## differential.forest.prediction -0.27651    0.39743 -0.6957 0.7567
```

*#The outcomes showed that the coefficient of the mean forest prediction was 1  
#which indicated the mean forest prediction was correct.  
#Also, the results indicated no heterogeneity been detected.*

**Two most important CATE predictors (For each method), plot CATE/predictor relationship**

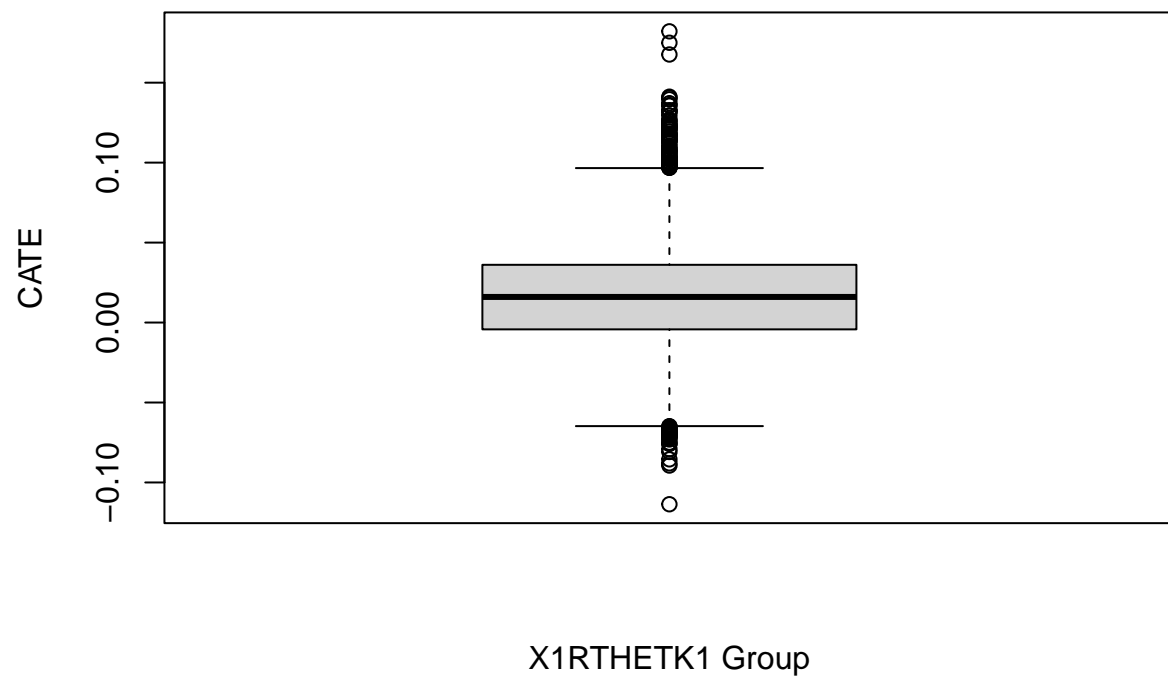
```
#Causal Forests
#The most two predictors are
Top2predictors <- importance_cf[order(importance_cf, decreasing = T),][1:2]

#Predictor 1
groupX1MTHETK1 <- quantile(test_data[, names(Top2predictors)[1]])
test_data$groupX1M <- ifelse(test_data$X1MTHETK1 >= groupX1MTHETK1[4], 4,
                             ifelse(test_data$X1MTHETK1 >= groupX1MTHETK1[3] & test_data$X1MTHETK1 < groupX1MTHETK1[4], 3,
                                     ifelse(test_data$X1MTHETK1 >= groupX1MTHETK1[2] & test_data$X1MTHETK1 < groupX1MTHETK1[3], 2,
                                             1)))
test_data$groupX1M <- factor(test_data$groupX1M)
boxplot(CATE_causalForest ~ test_data$groupX1M, xlab = "X1MTHETK1 Group", ylab = "CATE")
```



```
#Predictor 2
groupX1R <- quantile(test_data[, names(Top2predictors)[2]])
test_data$groupX1R <- ifelse(test_data$X1RTHETK1 >= groupX1R[4], 4,
                             ifelse(test_data$X1RTHETK1>=groupX1R[3] & test_data$X1RTHETK1<groupX1R[4],
                                     ifelse(test_data$X1RTHETK1>=groupX1R[2] & test_data$X1RTHETK1<groupX1R[3],
                                             1)))
test_data$groupX1R <- factor(test_data$groupX1R)
boxplot(CATE_causalForest ~ test_data$groupX1R, xlab = "X1RTHETK1 Group", ylab = "CATE")
```





## Method Comparisons and Conclusion

Insert BART conclusion

Insert GenericML conclusion

Insert Causal Forest conclusion

Comparison

In summary,