GSS Spending Survey Analysis: HTE Comparison using Bayesian Additive Regression Trees

Austin Murphy, Fred Lu

6/5/2020


```
library(here)        # File location
library(ggplot2)     # Plotting
library(BART)        # for BART models
library(fBasics)     # Summary statistics (3042.89)
library(kableExtra)  # Prettier RMarkdown (1.0.1)
library(dplyr)       # Data Manip
```

Project Description

This analysis studies the response to spending questions from the General Social Survey (GSS), influenced by the analysis done in Modeling Heterogeneous Treatment Effects in Survey Experiments with Bayesian Additive Regression Trees (http://www.donaldgreen.com/wp-content/uploads/2015/09/BART-Green-Kern-POQ-2012.pdf). In that paper, Green and Kern analyzed the well-established American dislike of programs labeled as 'welfare' by comparing how many respondents believe that the United States is spending too much money on 'welfare programs' vs how many respondents believe that the United States is spending too much money on 'assisting the poor'.

Project Question

In this analysis, we calculate heterogeneous treatment effects of respondents preferences for a government spending on assisting African Americans. The text of the question is given below, and respondents were given one of the two options:

> We are faced with many problems in this country, none of which can be solved easily or inexpensively. I'm going to name some of these problems, and for each one I'd like you to tell me whether you think we're spending too much money on it, too little money, or about the right amount... Are we spending too much, too little, or about the right amount on **Improving the conditions of Blacks** or **Assistance to Blacks**
>
> - Too little
> - About right
> - Too much
> - Don't know

We analyze the conditional average treatment effect of how many respondents selected 'Too much' is being spent on "Improving the conditions of Blacks" vs how many respondents selected 'Too much' is being spent on "Assistance to Blacks".

The Data

The data for this analysis is from the GSS Data Explorer tool (https://gssdataexplorer.norc.org/).

Part of our data comes from the Welfare dataset provided in the GSB Digital Business Initiative causal dataset GitHub (https://github.com/gsbDBI/ExperimentData). This part of the data has a number of covariates taken from the GSS website including demographic information and responses to other questions. We then pulled the response from the question described above and matched the responses to the individuals (in the GSS platform, the corresponding variables are `natrace` and `natracey`).

For more information on the questions and formatting of the questions on Trends in National Spending Priorities, 1973-2016, see this document (http://gss.norc.org/Documents/reports/social-change-reports/SC61%20GSS_Trends%20in%20Spending_1973-2016.pdf). Specific information on variables of interest can be viewed here (https://gssdataexplorer.norc.org/variables/vfilter).

We used the same dates as given in the Green and Kern paper, limiting to most surveys distributed from 1986-2010.

A description of the variables used in this analysis:

Covariates

- `partyid` : "Generally speaking, do you usually think of yourself as a Republican, Democrat, Independent, or what?" using a 7-point scale.
- `polviews` : A rating of one's political views on a 7 point scale, from extremely liberal–point 1–to extremely conservative–point 7.
- `age` : Age of respondent
- `educ` : Level of education attained by the respondent
- `attblack` : A measure of one's attitude towards African Americans. This metric was calculated by taking the average of responses to four questions (with question id given as `racdif{1|2|3|4}`), worded as follows:

> On the average (Negroes/Blacks/African-Americans) have worse jobs, income, and housing than white people. Do you think these differences are
>
> - Mainly due to discrimination?
> - Because most (Negroes/Blacks/African-Americans) have less in-born ability to learn?
> - Because most (Negroes/Blacks/African-Americans) don't have the chance for education that it takes to rise out of poverty?
> - Because most (Negroes/Blacks/African-Americans) just don't have the motivation or will power to pull themselves up out of poverty?

Treatment Indicator

- `W` : An indicator whether the respondent received the question that the government is spending too much money on "Improving the conditions of Blacks" (W==0) or "Assistance to Blacks" (W==1).

Outcome Varible

- `Y` : binary outcome, whether the respondent said the government was spending too much.

```
## Downloading script
# df <- readr::read_csv(file = "https://raw.githubusercontent.com/gsbDBI/ExperimentData/master/Welfare/ProcessedD
  ata/welfarenolabel3.csv", na = character())
# readr::write_csv(df, here::here("data/welfare.csv"))

# Read in provided welfare data
df <- readr::read_csv(file = here::here("data","welfare.csv"), na = character())
# Read in additional survey data
black_assistance <- readr::read_csv(file = here::here("data","black_assistance.csv"))
# Merge on year, id.
df <-
  df %>%
  dplyr::left_join(black_assistance)

# Specify outcome, treatment, and covariate variable names to use
outcome_variable_name <- "y_race" # removing y from welfare.csv.
treatment_variable_name <- "w"
covariate_names <- c("partyid", "polviews", "age", "educ", "year", "attblack")

# Combine all names
all_variables_names <- c(covariate_names, outcome_variable_name, treatment_variable_name)
df <- df[, which(names(df) %in% all_variables_names)]

# Rename variables
names(df)[names(df) == outcome_variable_name] <- "Y"
names(df)[names(df) == treatment_variable_name] <- "W"

# Flip Y...
```

The original dataframe has 36501 responses.

Filtering out missing data and non-responses

```
# Filter out missing data
df <-
  df %>%
  filter_all(all_vars(. != -999)) %>%
  filter(partyid != 7) # partyid==7 is equivalent to I'm not sure

# polviews has some values that are 4.1220088 for no reason... remove those
is.wholenumber <-
    function(x, tol = .Machine$double.eps^0.5)  abs(x - round(x)) < tol

df <- df[is.wholenumber(df$polviews),]
```

After filtering out missing and incorrect encodings, the data has 19738 responses.

Here's a brief description table of the variables

```
# Make a data.frame containing summary statistics of interest
summ_stats <- fBasics::basicStats(df)
summ_stats <- as.data.frame(t(summ_stats))

# Rename some of the columns for convenience
summ_stats <- summ_stats[c("Mean", "Stdev", "Minimum", "1. Quartile", "Median",  "3. Quartile", "Maximum")] %>%
  rename("Lower quartile" = '1. Quartile', "Upper quartile"= "3. Quartile")
```

	Mean	Stdev	Minimum	Lower quartile	Median	Upper quartile	Maximum
year	1997.38	7.25	1986	1991.00	1996.00	2004.0	2010
age	45.89	17.28	18	32.00	43.00	58.0	89
educ	13.25	2.99	0	12.00	13.00	16.0	20
partyid	2.78	2.01	0	1.00	3.00	5.0	6
polviews	4.13	1.38	1	3.00	4.00	5.0	7
W	0.50	0.50	0	0.00	1.00	1.0	1
attblack	0.38	0.24	0	0.25	0.33	0.5	1
Y	0.19	0.39	0	0.00	0.00	0.0	1

For a comparison of the number of resondents per year with the paper:

```
year_w_tab <-
  janitor::tabyl(df, year, W) %>%
  arrange(year) %>%
  janitor::adorn_totals()
```

Year	Improving the Conditions of Blacks	Assistance to Blacks
1986	686	694
1988	449	488
1989	450	475
1990	631	657
1991	475	485
1993	510	515
1994	926	911
1996	888	925
1998	838	882
2000	866	846
2002	411	434
2004	433	428
2006	940	933
2008	611	627
2010	679	645
Total	9793	9945

This doesn't match up with the paper's table, but the paper gives no indication of how they filtered or subset the data.

Given that the survey was a randomized experiment, we calculate the ATE by difference in means:

```
mean(df$Y[df$W == 0]) - mean(df$Y[df$W == 1])
```

```
## [1] -0.06223962
```

```
# Note:
# - W==0: Improving the conditions of Blacks
# - W==1; Assistance to Blacks
# Y[W==0] - Y[W==1]: Positive if we're spending too much on "improving the conditions of Blacks" compared with "A
  ssisting Blacks"

# Y[W==1] - Y[W==0]: Positive if we're spending too much on "Assisting Blacks" vs "improving the conditions of Bl
  acks" compared with
```

Given this, we take the negative of the response variable to offer a more easily interpretable outcome.

```
df$W <- ifelse(df$W==0, 1, 0)
mean(df$Y[df$W == 0]) - mean(df$Y[df$W == 1])
```

```
## [1] 0.06223962
```

Bayesian Additive Regression Trees (BART)

The paper used a probit BART model to estimate CATE because the response is binary.

BART wants a `x.test` matrix to be passed in when training the model. According to the paper, for each value of the covariate of interest, you create two copy of the training set. For each copy, set the covariate of interest to the value in question and set the treatment indicator to 0 for one copy and 1 to the other.

To more easily create test matrices, we'll create two copies for each level in which we're interested.:

```
create_test_matrix <- function(df, column_name, covariate_levels){
  # Get unique values
  # covariate_levels = unique(df[[column_name]][order(df[column_name])])
  num_levels = length(covariate_levels)
  treatment_values = c(0,1)
  # Create a 'design matrix', with each row representing the pair of covariate value and treatment value used in
   filling the test matrix
  # first column: covariate value, second column: treatment level
  test_design = cbind(rep(covariate_levels,each=2),
                      rep(treatment_values,times=num_levels))
  # print(test_design)

  # prep 'test' data. Large matrix of size (num values of interest * 2 * N) x ncols
  n = nrow(df)
  test_bart = NULL
  for (i in 1:nrow(test_design)){
    test0 = train %>% select(-Y)
    test0[ , column_name] = test_design[i,1]
    test0[ , 'W'] = test_design[i,2]

    test_bart = rbind(test_bart,test0)
  }
  dim(test_bart)
  return(list(test_design, test_bart))
}
```

Party Identification

We first compute the CATE estimate for all values on the 7-point scale for Party Identification.

```
train <- df
column_name = "partyid"
table(df[[column_name]])
```

```
##
##    0    1    2    3    4    5    6
## 3066 3983 2341 2861 1886 3442 2159
```

```
covariate_values = seq(0,6)
# print(covariate_values)

out <- create_test_matrix(df, column_name, covariate_levels = covariate_values)
test_design = out[[1]]; test_bart = out[[2]]

# Train BART model
set.seed(1)
bart =  BART::pbart(x.train = as.data.frame(train %>% select(-Y)),
                    y.train = train$Y,
                    x.test = as.data.frame(test_bart),
                    nskip = 1000)
```

```
## *****Into main of pbart
## *****Data:
## data:n,p,np: 19738, 7, 276332
## y1,yn: 0, 0
## x1,x[n*p]: 1986.000000, 0.500000
## xp1,xp[np*p]: 1986.000000, 0.500000
## *****Number of Trees: 50
## *****Number of Cut Points: 14 ... 6
## *****burn and ndpost: 1000, 1000
## *****Prior:mybeta,alpha,tau: 2.000000,0.950000,0.212132
## *****binaryOffset: -0.879993
## *****Dirichlet:sparse,theta,omega,a,b,rho,augment: 0,0,1,0.5,1,7,0
## *****nkeeptrain,nkeeptest,nkeeptreedraws: 1000,1000,1000
## *****printevery: 100
## *****skiptr,skipte,skiptreedraws: 1,1,1
##
## MCMC
## done 0 (out of 2000)
## done 100 (out of 2000)
## done 200 (out of 2000)
## done 300 (out of 2000)
## done 400 (out of 2000)
## done 500 (out of 2000)
## done 600 (out of 2000)
## done 700 (out of 2000)
## done 800 (out of 2000)
## done 900 (out of 2000)
## done 1000 (out of 2000)
## done 1100 (out of 2000)
## done 1200 (out of 2000)
## done 1300 (out of 2000)
## done 1400 (out of 2000)
## done 1500 (out of 2000)
## done 1600 (out of 2000)
## done 1700 (out of 2000)
## done 1800 (out of 2000)
## done 1900 (out of 2000)
## time: 148s
## check counts
## trcnt,tecnt: 1000,1000
```

A function to create the CATE estimates from the BART model output

```
calculate_cate <- function(bart, test_design){
  ## turn z-scores into probabilities
  print("Transforming to probabilities")
  bart$prob.test <- pnorm(bart$yhat.test)
  ## average over the posterior samples
  batch_indices = seq(1, to=ncol(bart$prob.test)+1, length.out=nrow(test_design)+1)
  # save memory
  bart$yhat.test <- NULL


  print("Averaging each batch")
  avg_post_draws = as.data.frame(matrix(NA, 1000, length(batch_indices)-1))
  for (i in 1:(length(batch_indices)-1)){
    # Select the data from one 1000xN dataframe
    batch = bart$prob.test[ , batch_indices[i]:(batch_indices[i+1]-1) ]
    # average over the observations to get E[CATE]
    avg_batch = apply(batch, 1, mean)
    # sanity check: should be of length 1000
    length(avg_batch) == 1000

    avg_post_draws[, i] = avg_batch
  }
  dim(avg_post_draws) # should be 1000 x 2*number of point estimates

  # save memory
  bart$prob.test <- NULL

  print("Estimating CATEs")
  ## Summary dataframe for CATE, and lower and upper bound estimates
  covariate_levels = unique(test_design[,1])
  estimates = data.frame(matrix(NA, length(covariate_levels), 4))
  estimates[,1] = covariate_levels
  names(estimates) <- c("covariate_val","mean","low","high")
  # Loop through the posterior draws and calculate the difference between them.
  # Calculate average CATE and 95% posterior bands.
  for (i in 1:dim(estimates)[1]){
    covariate_val = 2*i-1
    cate = avg_post_draws[, covariate_val] - avg_post_draws[, covariate_val+1]
    cate = cate[order(cate)]
    estimates[i,"mean"] = mean(cate)
    estimates[i,"low"] = quantile(cate, 0.025)
    estimates[i,"high"] = quantile(cate, 0.975)
  }
  return(estimates)
}
```

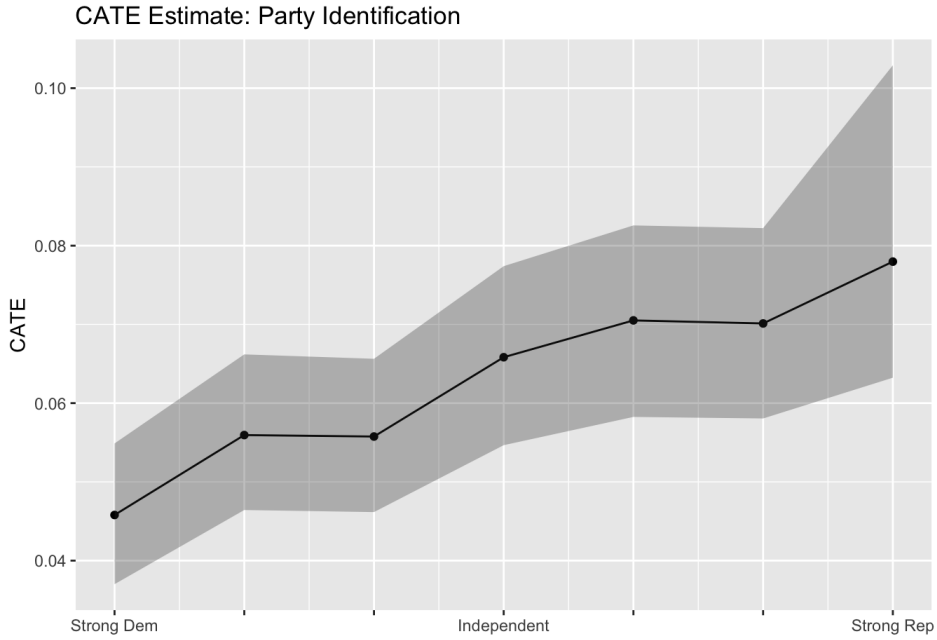
Estimate on partyid

```
estimates <- calculate_cate(bart=bart,
                            test_design=test_design)
```

```
## [1] "Transforming to probabilities"
## [1] "Averaging each batch"
## [1] "Estimating CATEs"
```

Let's see how it looks:

```
# Plot the CATE estimates
ggplot(estimates, aes(x = covariate_val, y = mean)) +
  geom_line() +
  geom_point() +
  geom_ribbon(aes(ymin=low, ymax=high), alpha=0.3) +
  ggtitle("CATE Estimate: Party Identification") +
  xlab("") +
  ylab("CATE") +
  scale_x_continuous(breaks=covariate_values,
        labels=c("Strong Dem","","","Independent","","","Strong Rep"))
```



Political Views

Now for Political Views

```
column_name = "polviews"
table(df[[column_name]])
```

```
##
##    1    2    3    4    5    6    7
##  560 2290 2518 7431 3145 3163  631
```

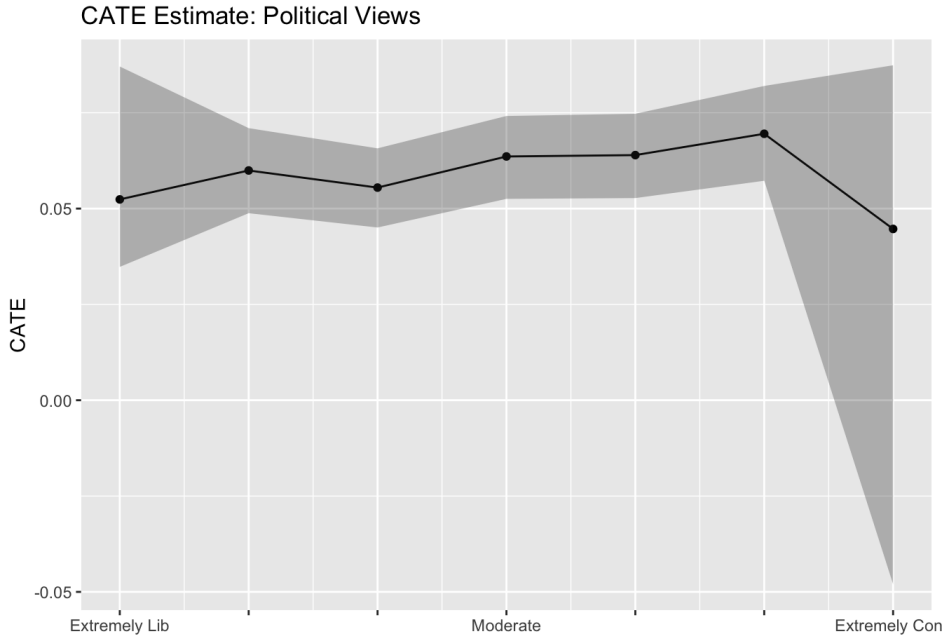
```
covariate_values = seq(1,7)
# print(covariate_values)

out <- create_test_matrix(df, column_name, covariate_levels = covariate_values)
test_design = out[[1]]; test_bart = out[[2]]

# Train BART model
rm(bart)
set.seed(1)
bart =  BART::pbart(x.train = as.data.frame(train %>% select(-Y)),
                    y.train = train$Y,
                    x.test = as.data.frame(test_bart),
                    nskip = 1000)
```

```
## *****Into main of pbart
## *****Data:
## data:n,p,np: 19738, 7, 276332
## y1,yn: 0, 0
## x1,x[n*p]: 1986.000000, 0.500000
## xp1,xp[np*p]: 1986.000000, 0.500000
## *****Number of Trees: 50
## *****Number of Cut Points: 14 ... 6
## *****burn and ndpost: 1000, 1000
## *****Prior:mybeta,alpha,tau: 2.000000,0.950000,0.212132
## *****binaryOffset: -0.879993
## *****Dirichlet:sparse,theta,omega,a,b,rho,augment: 0,0,1,0.5,1,7,0
## *****nkeeptrain,nkeeptest,nkeeptreedraws: 1000,1000,1000
## *****printevery: 100
## *****skiptr,skipte,skiptreedraws: 1,1,1
##
## MCMC
## done 0 (out of 2000)
## done 100 (out of 2000)
## done 200 (out of 2000)
## done 300 (out of 2000)
## done 400 (out of 2000)
## done 500 (out of 2000)
## done 600 (out of 2000)
## done 700 (out of 2000)
## done 800 (out of 2000)
## done 900 (out of 2000)
## done 1000 (out of 2000)
## done 1100 (out of 2000)
## done 1200 (out of 2000)
## done 1300 (out of 2000)
## done 1400 (out of 2000)
## done 1500 (out of 2000)
## done 1600 (out of 2000)
## done 1700 (out of 2000)
## done 1800 (out of 2000)
## done 1900 (out of 2000)
## time: 156s
## check counts
## trcnt,tecnt: 1000,1000
```

```
estimates <- calculate_cate(bart=bart,
                            test_design=test_design)
```

```
## [1] "Transforming to probabilities"
## [1] "Averaging each batch"
## [1] "Estimating CATEs"
```

```
# Plot the CATE estimates
ggplot(estimates, aes(x = covariate_val, y = mean)) +
  geom_line() +
  geom_point() +
  geom_ribbon(aes(ymin=low, ymax=high), alpha=0.3) +
  ggtitle("CATE Estimate: Political Views") +
  xlab("") +
  ylab("CATE") +
  scale_x_continuous(breaks=covariate_values,
        labels=c("Extremely Lib","","","Moderate","","","Extremely Cons"))
```



Age

Now for Age (age)

```
column_name = "age"
table(df[[column_name]])
```

```
##
##  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37
##  54 223 259 300 323 335 351 429 379 371 455 385 421 432 456 446 460 462 434 472
##  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57
## 461 426 426 431 428 424 407 372 336 379 360 373 342 340 315 287 281 265 269 244
##  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77
## 212 258 269 248 245 232 201 207 222 220 224 202 189 160 214 186 181 153 144 147
##  78  79  80  81  82  83  84  85  86  87  88  89
## 139 117 107  93  94  85  62  60  50  48  40 116
```

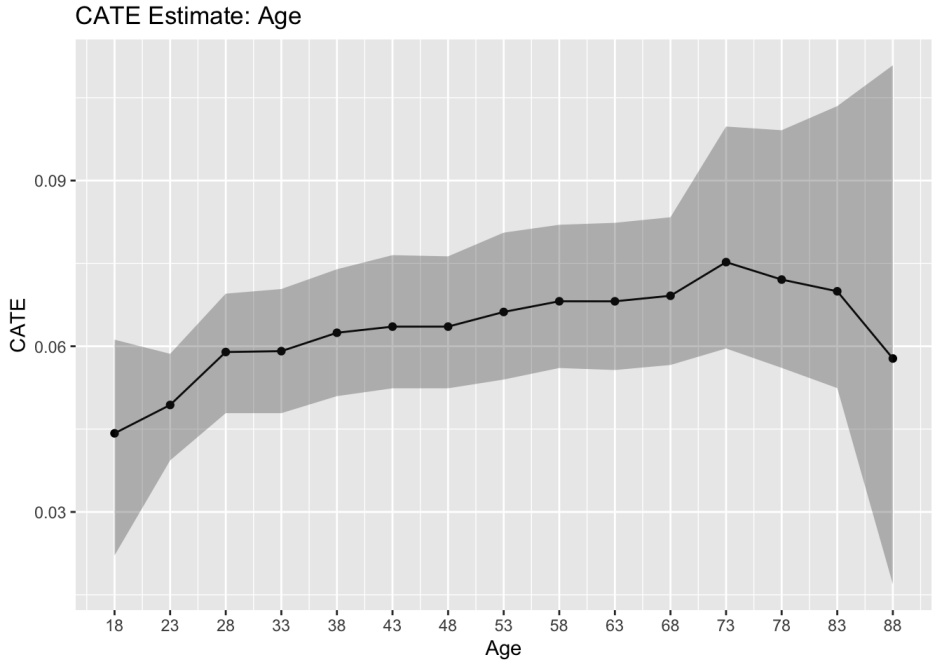
```
covariate_values = seq(18,89,by=5)
# print(covariate_values)

out <- create_test_matrix(df, column_name, covariate_levels = covariate_values)
test_design = out[[1]]; test_bart = out[[2]]

# Train BART model
rm(bart)
set.seed(1)
bart =  BART::pbart(x.train = as.data.frame(train %>% select(-Y)),
                    y.train = train$Y,
                    x.test = as.data.frame(test_bart),
                    nskip = 1000)
```

```
## *****Into main of pbart
## *****Data:
## data:n,p,np: 19738, 7, 592140
## y1,yn: 0, 0
## x1,x[n*p]: 1986.000000, 0.500000
## xp1,xp[np*p]: 1986.000000, 0.500000
## *****Number of Trees: 50
## *****Number of Cut Points: 14 ... 6
## *****burn and ndpost: 1000, 1000
## *****Prior:mybeta,alpha,tau: 2.000000,0.950000,0.212132
## *****binaryOffset: -0.879993
## *****Dirichlet:sparse,theta,omega,a,b,rho,augment: 0,0,1,0.5,1,7,0
## *****nkeeptrain,nkeeptest,nkeeptreedraws: 1000,1000,1000
## *****printevery: 100
## *****skiptr,skipte,skiptreedraws: 1,1,1
##
## MCMC
## done 0 (out of 2000)
## done 100 (out of 2000)
## done 200 (out of 2000)
## done 300 (out of 2000)
## done 400 (out of 2000)
## done 500 (out of 2000)
## done 600 (out of 2000)
## done 700 (out of 2000)
## done 800 (out of 2000)
## done 900 (out of 2000)
## done 1000 (out of 2000)
## done 1100 (out of 2000)
## done 1200 (out of 2000)
## done 1300 (out of 2000)
## done 1400 (out of 2000)
## done 1500 (out of 2000)
## done 1600 (out of 2000)
## done 1700 (out of 2000)
## done 1800 (out of 2000)
## done 1900 (out of 2000)
## time: 266s
## check counts
## trcnt,tecnt: 1000,1000
```

```
estimates <- calculate_cate(bart=bart,
                            test_design=test_design)
```

```
## [1] "Transforming to probabilities"
## [1] "Averaging each batch"
## [1] "Estimating CATEs"
```

```
# Plot the CATE estimates
ggplot(estimates, aes(x = covariate_val, y = mean)) +
  geom_line() +
  geom_point() +
  geom_ribbon(aes(ymin=low, ymax=high), alpha=0.3) +
  ggtitle("CATE Estimate: Age") +
  xlab("Age") +
  ylab("CATE") +
  scale_x_continuous(breaks=covariate_values,
        labels=covariate_values)
```



Attitude Towards Blacks

```
column_name = "attblack"
table(df[[column_name]])
```

```
##
##          0       0.25 0.33333334        0.5 0.66666669       0.75          1
##       2514       6995        714       6690        372       1810        643
```

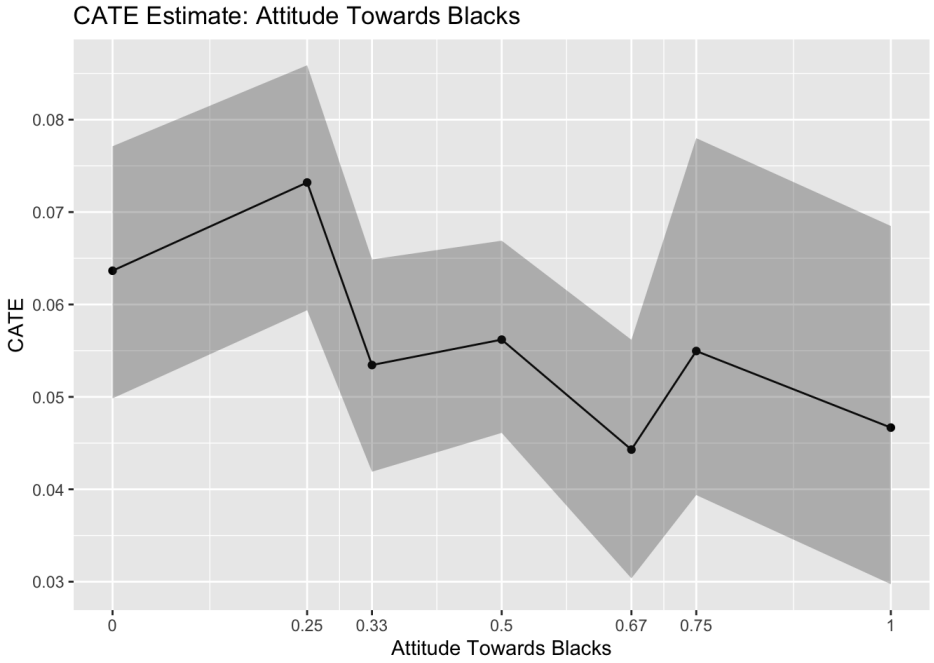
```
covariate_values = unique(df[[column_name]])[order(unique(df[[column_name]]))]
# print(covariate_values)

out <- create_test_matrix(df, column_name, covariate_levels = covariate_values)
test_design = out[[1]]; test_bart = out[[2]]

# Train BART model
rm(bart)
set.seed(1)
bart =  BART::pbart(x.train = as.data.frame(train %>% select(-Y)),
                    y.train = train$Y,
                    x.test = as.data.frame(test_bart),
                    nskip = 1000)
```

```
## *****Into main of pbart
## *****Data:
## data:n,p,np: 19738, 7, 276332
## y1,yn: 0, 0
## x1,x[n*p]: 1986.000000, 0.500000
## xp1,xp[np*p]: 1986.000000, 1.000000
## *****Number of Trees: 50
## *****Number of Cut Points: 14 ... 6
## *****burn and ndpost: 1000, 1000
## *****Prior:mybeta,alpha,tau: 2.000000,0.950000,0.212132
## *****binaryOffset: -0.879993
## *****Dirichlet:sparse,theta,omega,a,b,rho,augment: 0,0,1,0.5,1,7,0
## *****nkeeptrain,nkeeptest,nkeeptreedraws: 1000,1000,1000
## *****printevery: 100
## *****skiptr,skipte,skiptreedraws: 1,1,1
##
## MCMC
## done 0 (out of 2000)
## done 100 (out of 2000)
## done 200 (out of 2000)
## done 300 (out of 2000)
## done 400 (out of 2000)
## done 500 (out of 2000)
## done 600 (out of 2000)
## done 700 (out of 2000)
## done 800 (out of 2000)
## done 900 (out of 2000)
## done 1000 (out of 2000)
## done 1100 (out of 2000)
## done 1200 (out of 2000)
## done 1300 (out of 2000)
## done 1400 (out of 2000)
## done 1500 (out of 2000)
## done 1600 (out of 2000)
## done 1700 (out of 2000)
## done 1800 (out of 2000)
## done 1900 (out of 2000)
## time: 146s
## check counts
## trcnt,tecnt: 1000,1000
```

```
estimates <- calculate_cate(bart=bart,
                            test_design=test_design)
```

```
## [1] "Transforming to probabilities"
## [1] "Averaging each batch"
## [1] "Estimating CATEs"
```

```
# Plot the CATE estimates
ggplot(estimates, aes(x = covariate_val, y = mean)) +
  geom_line() +
  geom_point() +
  geom_ribbon(aes(ymin=low, ymax=high), alpha=0.3) +
  ggtitle("CATE Estimate: Attitude Towards Blacks") +
  xlab("Attitude Towards Blacks") +
  ylab("CATE") +
  scale_x_continuous(breaks=covariate_values,
        labels=round(covariate_values,2))
```


CATE Estimate: Attitude Towards Blacks
CATE
0.08
0.07
0.06
0.05
0.04
0.03
0
0.25
0.33
0.5
0.67
0.75
1
Attitude Towards Blacks