Causal Inference Algorithms Evaluation

Depeng Kong, Henan Xu, Wannian Lou, Yiran Lin, Zihan Chen

12/02/2020

GR5243 Fall 2020 Applied Data Science

Project 4 Causal Inference Algorithms Evaluation

We are going to calculate ATE using three main methods, that is Propensity Matching, Inverse Propensity Weighting and Doubly Robust Estimation.

1. Setup

```r
library(Matching)
library(glmnet)
library(tidyverse)
library(ggplot2)
setwd("./")
```

2. Load Data

```r
ldim <- read.csv("../data/lowDim_dataset.csv")
hdim <- read.csv("../data/highDim_dataset.csv")

# Low Dimention
ltr <- ldim$A
ly <- ldim$Y
lx <- ldim[,-c(1,2)]

# High Dimention
htr <- hdim$A
hy <- hdim$Y
hx <- hdim[, -c(1,2)]
```

3. Calculate Propensity Score with L2 Ridge regression for Propensity Matching

```r
seed <- c(0,2,3,5)
start_time <- Sys.time()

p_score <- function(seednum){
  set.seed(seednum)
  glm1 <- cv.glmnet(as.matrix(lx), ltr, family = "binomial", alpha = 0)

  glm1.fit <- predict(glm1$glmnet.fit,
                      s = glm1$lambda.min,
                      newx = as.matrix(lx),
                      type = "response")

  set.seed(seednum)
  glm2 <- cv.glmnet(as.matrix(hx), htr, family = "binomial", alpha = 0)

  glm2.fit <- predict(glm2$glmnet.fit,
                      s = glm2$lambda.min,
                      newx = as.matrix(hx),
                      type = "response")

  return(list(l=glm1.fit,h=glm2.fit))

}

p_score_list <- lapply(seed,p_score)
```

4. Propensity Matching

4.1 Distance calculated

```r
dist_mat <- function(li){
  glm1.fit <- li$l
  glm2.fit <- li$h
  n1 <- length(glm1.fit)
  dt1 <- matrix(0,nrow = n1, ncol = n1)
  for (i in 1:(n1-1)){
    dt1[i,i] <- 1
    for (j in (i+1):n1){
      dt1[i,j] <- abs(glm1.fit[i] - glm1.fit[j])
      dt1[j,i] <- dt1[i,j]
    }
  }



  n2 <- length(glm2.fit)
  dt2 <- matrix(0,nrow = n2, ncol = n2)
  for (i in 1:(n2-1)){
    dt2[i,i] <- 1
    for (j in (i+1):n2){
      dt2[i,j] <- abs(glm2.fit[i] - glm2.fit[j])
```

```
      dt2[j,i] <- dt2[i,j]
    }
  }

  return(list(lm=dt1,hm=dt2))

}

dist_mat_list <- lapply(p_score_list,dist_mat)

end_time <- Sys.time()
tm <- end_time - start_time
cat("Time for Preparing is: ")
```

```
## Time for Preparing is:
```

```
tm
```

```
## Time difference of 33.57126 secs
```

4.2 Propensity Score Marching Function

```
cal_neighbour <- function(index,df,thresh,y,A){
  dt_vec <- df[index,]
  ind_vec <- which(dt_vec<thresh)
  ind_final=ind_vec[A[index]!=A[ind_vec]]

  if (length(ind_final)==0){
    return(NA)
  }
  else{
    return(list(mean(y[ind_final]),ind_final))
  }

}
```

4.3 Matching Low-Dim

```
seq = 10:200/10000

start_time <- Sys.time()

get_ate_pair <- function(ind){
  dt1 <- dist_mat_list[[ind]]$lm
  a <- as.vector(dt1)

  ATE_low <- vector("double")
  pairs_low <- vector("double")
  for (percentage in seq){
```

```r
    threshold <- quantile(a,percentage)


    n1_vec <- 1:nrow(dt1)
    list_1 <- lapply(n1_vec,cal_neighbour,df=dt1,thresh = threshold,y=ly,A=ltr)
    mean_list_1 <- lapply(n1_vec,function(x) unlist(list_1[[x]][1]))
    mean_cal_1 <- unlist(mean_list_1)
    neighbour_list_1 <- lapply(n1_vec,function(x) unlist(list_1[[x]][2]))

    df_1 <- (data.frame(Y=ly,A=ltr)
             %>%mutate(ind = row_number())
             %>%mutate(AAA=neighbour_list_1)
             %>%mutate(mean_cal = mean_cal_1)
             %>%filter(!is.na(mean_cal))
             %>%mutate(ATE = (Y-mean_cal)*ifelse(A==0,-1,1))

    )

    ATE_low <- append(ATE_low,mean(df_1$ATE))
    pairs_low <- append(pairs_low,sum(!is.na(unlist(neighbour_list_1)))/2)
  }
  return(list(ate=ATE_low,pair=pairs_low))

}

ind_mat <- 1:4
low_list <- lapply(ind_mat,get_ate_pair)


end_time <- Sys.time()
tm <- end_time - start_time
cat("Time for Propensity Matching Low_Dim is: ")
```

```
## Time for Propensity Matching Low_Dim is:
```

```r
tm
```

```
## Time difference of 19.671 secs
```

4.4 Matching High-Dim

```r
start_time <- Sys.time()

seq = 10:200/10000
get_ate_pair <- function(ind){
  dt2 <- dist_mat_list[[ind]]$hm
  a_h <- as.vector(dt2)

  ATE_high <- vector("double")
  pairs_high <- vector("double")
```

```
  for (percentage in seq){

    threshold <- quantile(a_h,percentage)


    n2_vec <- 1:nrow(dt2)
    list_2 <- lapply(n2_vec,cal_neighbour,df=dt2,thresh = threshold,y=hy,A=htr)
    mean_list_2 <- lapply(n2_vec,function(x) unlist(list_2[[x]][1]))
    mean_cal_2 <- unlist(mean_list_2)
    neighbour_list_2 <- lapply(n2_vec,function(x) unlist(list_2[[x]][2]))

    df_2 <- (data.frame(Y=hy,A=htr)
             %>%mutate(ind = row_number())
             %>%mutate(AAA=neighbour_list_2)
             %>%mutate(mean_cal = mean_cal_2)
             %>%filter(!is.na(mean_cal))
             %>%mutate(ATE = (Y-mean_cal)*ifelse(A==0,-1,1))

    )

    ATE_high <- append(ATE_high,mean(df_2$ATE))
    pairs_high <- append(pairs_high,sum(!is.na(unlist(neighbour_list_2)))/2)
  }

  return(list(ate=ATE_high,pair=pairs_high))

}

ind_mat <- 1:4
high_list <- lapply(ind_mat,get_ate_pair)

end_time <- Sys.time()
tm <- end_time - start_time
cat("Time for Propensity Matching High_Dim is: ")
```

```
## Time for Propensity Matching High_Dim is:
```

```
tm
```

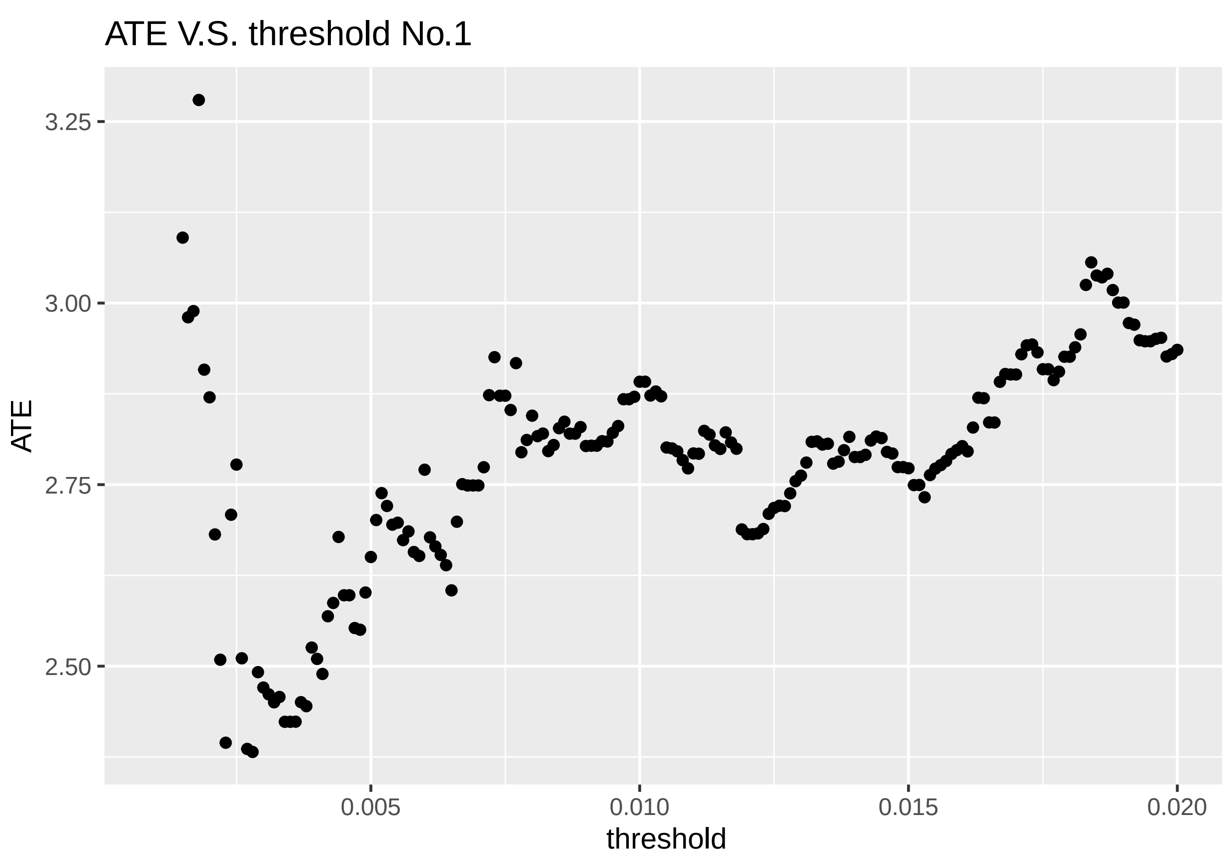
```
## Time difference of 2.726467 mins
```

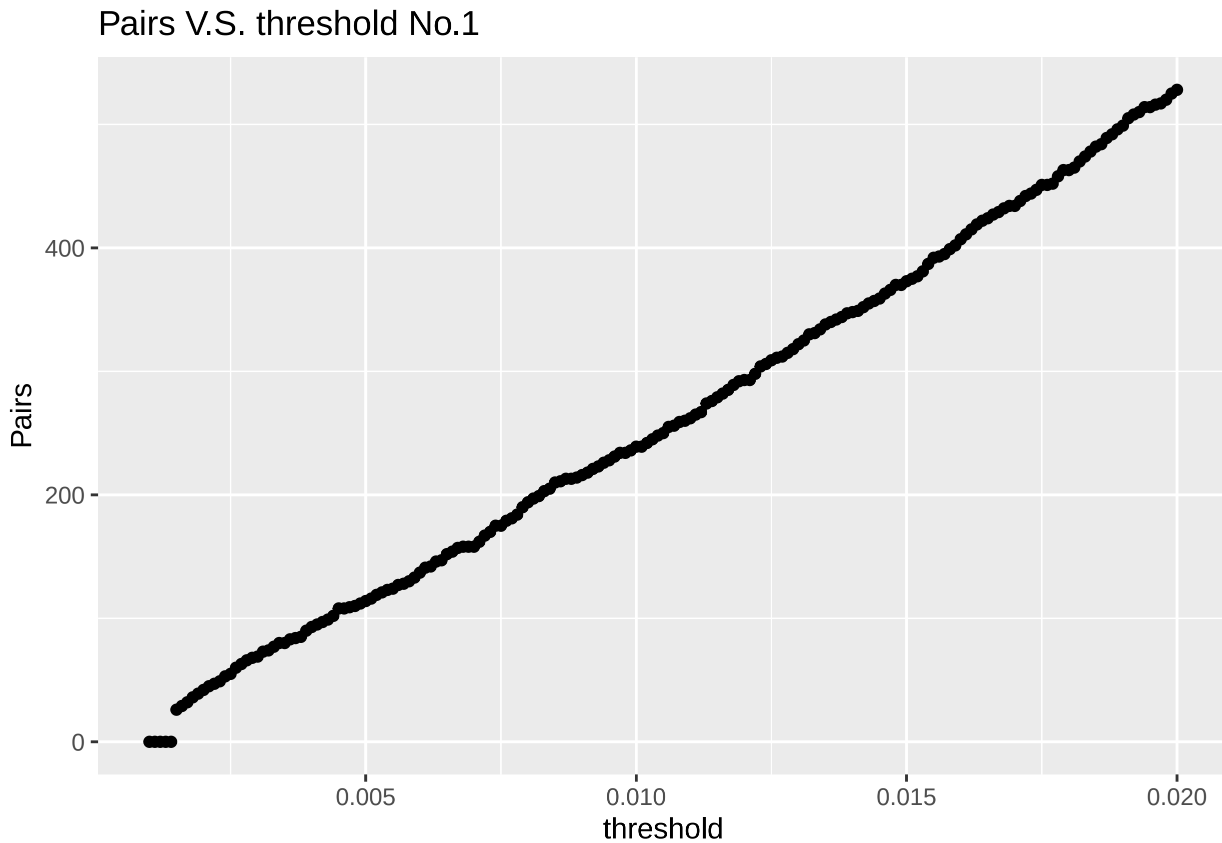
4.5 Plotting Part for Low-Dim

```
for (i in 1:4){
  ATE_low <- low_list[[i]]$ate
  pairs_low <- low_list[[i]]$pair
  plot_low <- data.frame(x=seq,ATE=ATE_low,pairs=pairs_low)
```

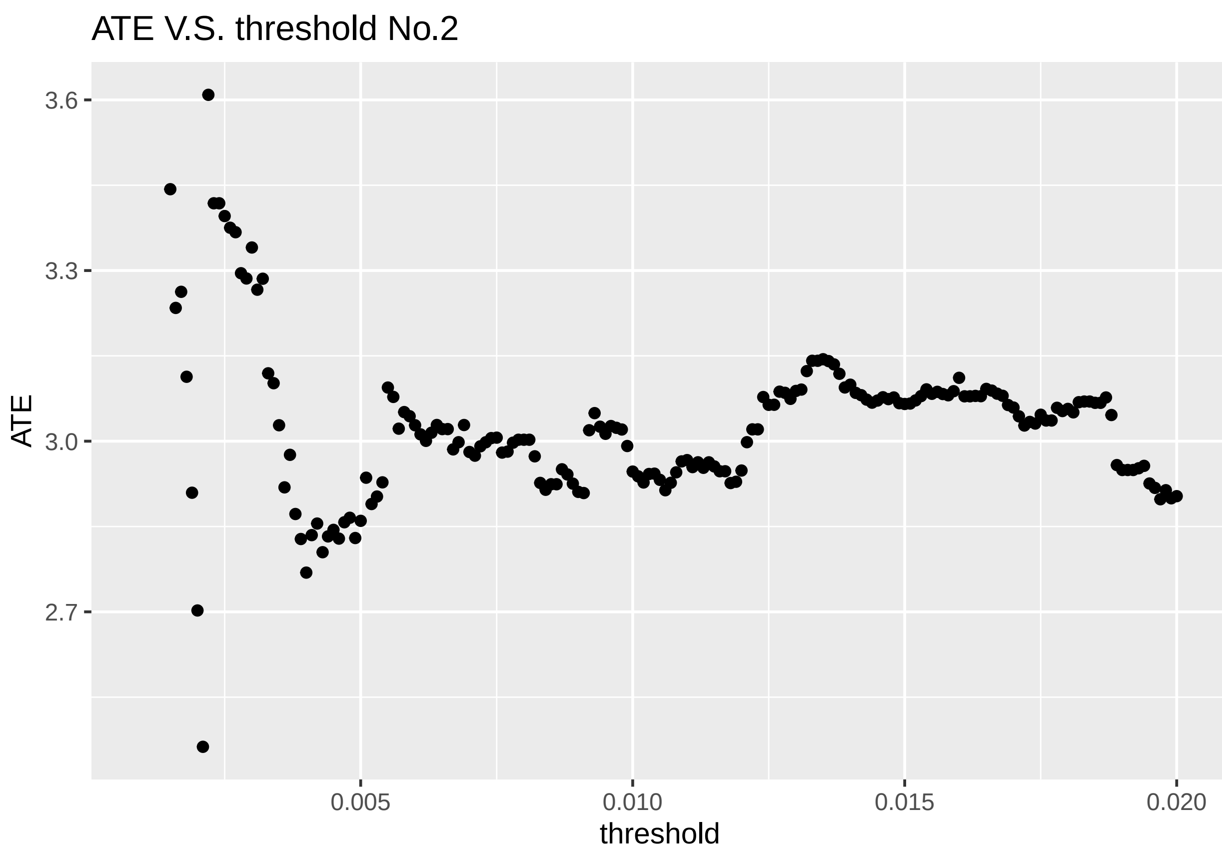
```r
  g_low <- ggplot(plot_low)+
    geom_point(aes(x,ATE))+
    labs(
      title = paste0("ATE V.S. threshold No.",i),
      x = "threshold",
      y = "ATE"
    )
  print(g_low)

  match_low <- ggplot(plot_low)+
    geom_point(aes(x,pairs))+
    labs(
      title = paste0("Pairs V.S. threshold No.",i),
      x = "threshold",
      y = "Pairs"
    )
  print(match_low)
}
```

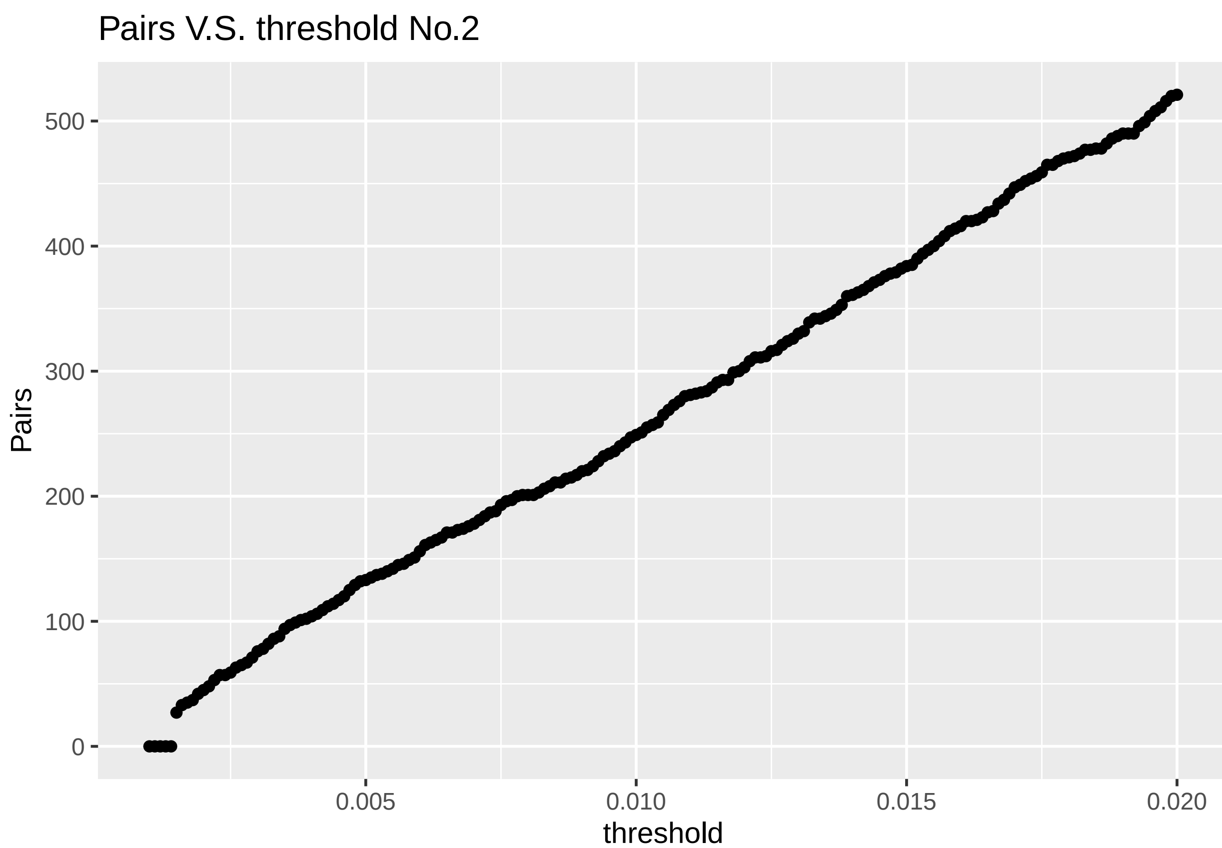
```
## Warning: Removed 5 rows containing missing values (geom_point).
```



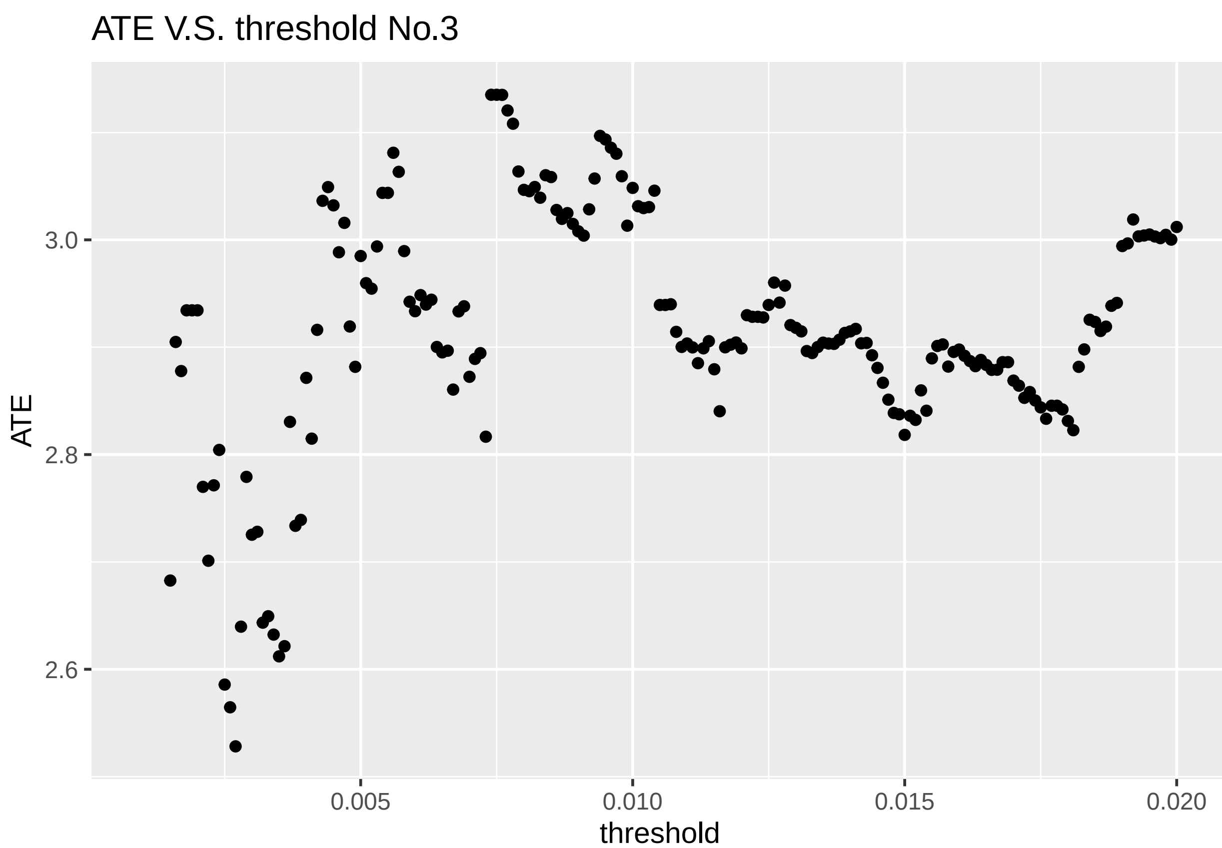


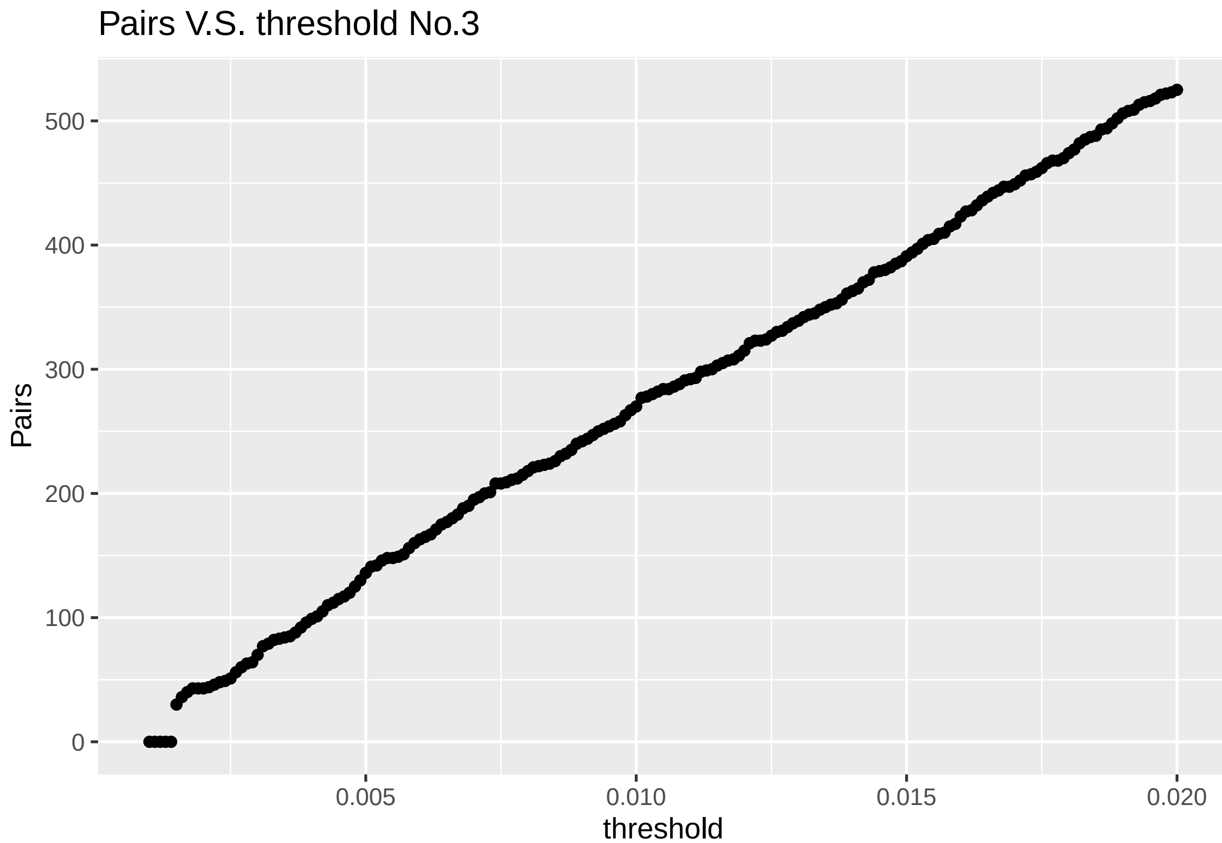

```
## Warning: Removed 5 rows containing missing values (geom_point).
```


ATE V.S. threshold No.2
3.6
3.3
3.0
2.7
ATE
0.005
0.010
0.015
0.020
threshold

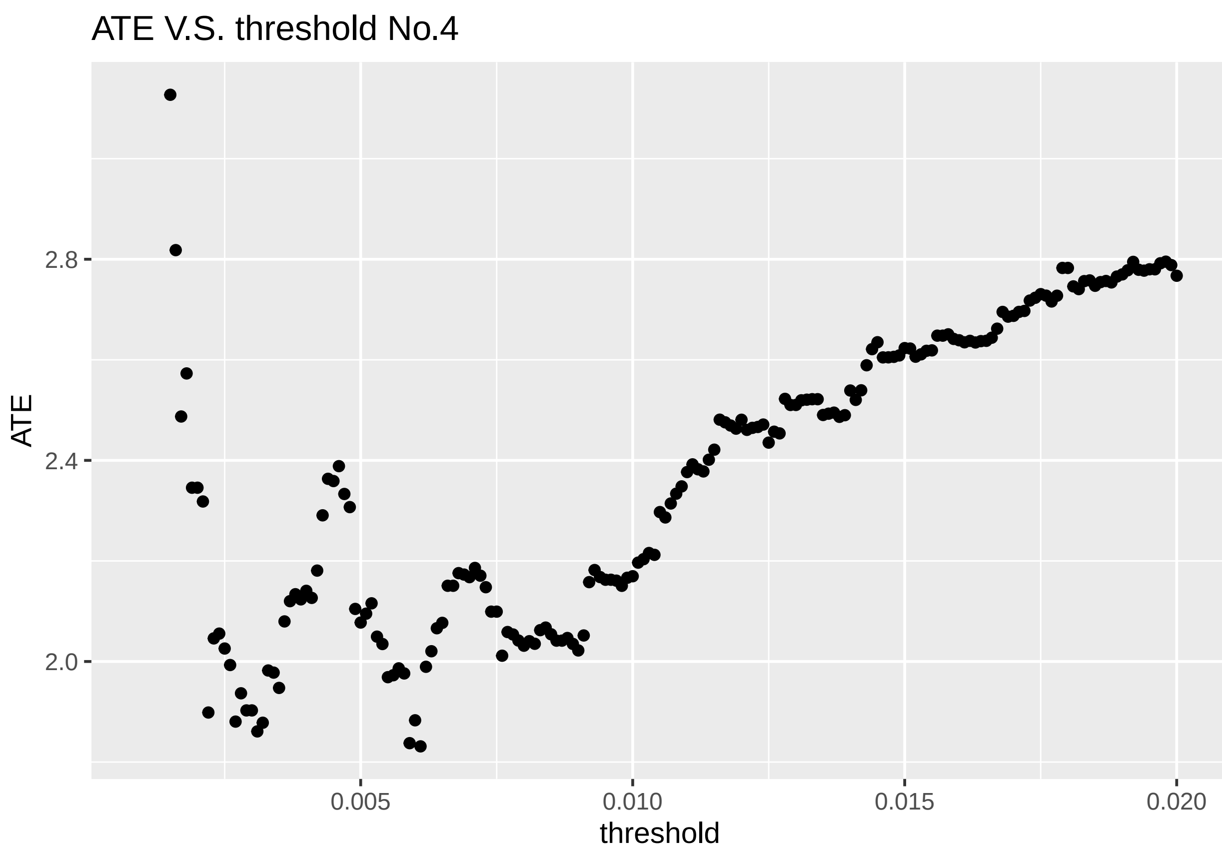


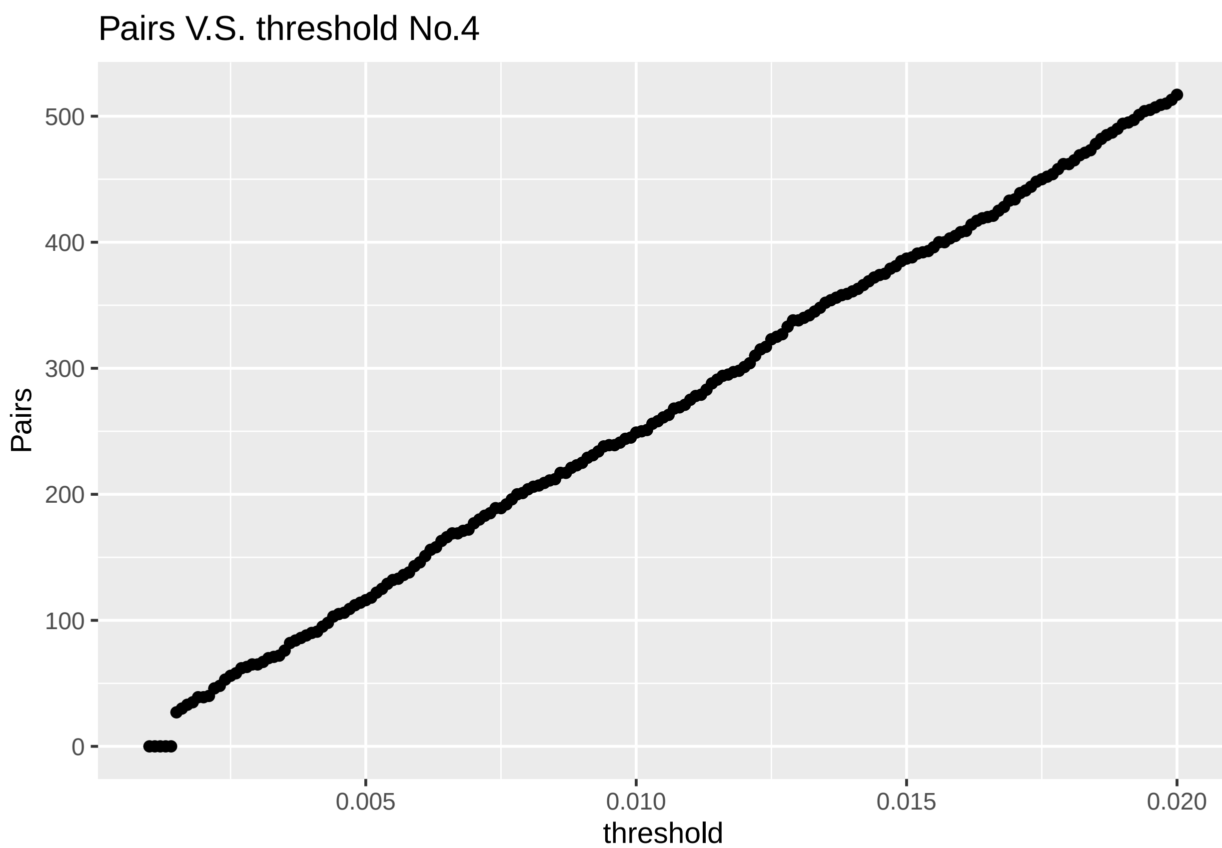

```
## Warning: Removed 5 rows containing missing values (geom_point).
```


ATE V.S. threshold No.3
ATE
3.0
2.8
2.6
0.005
0.010
0.015
0.020
threshold




```
## Warning: Removed 5 rows containing missing values (geom_point).
```

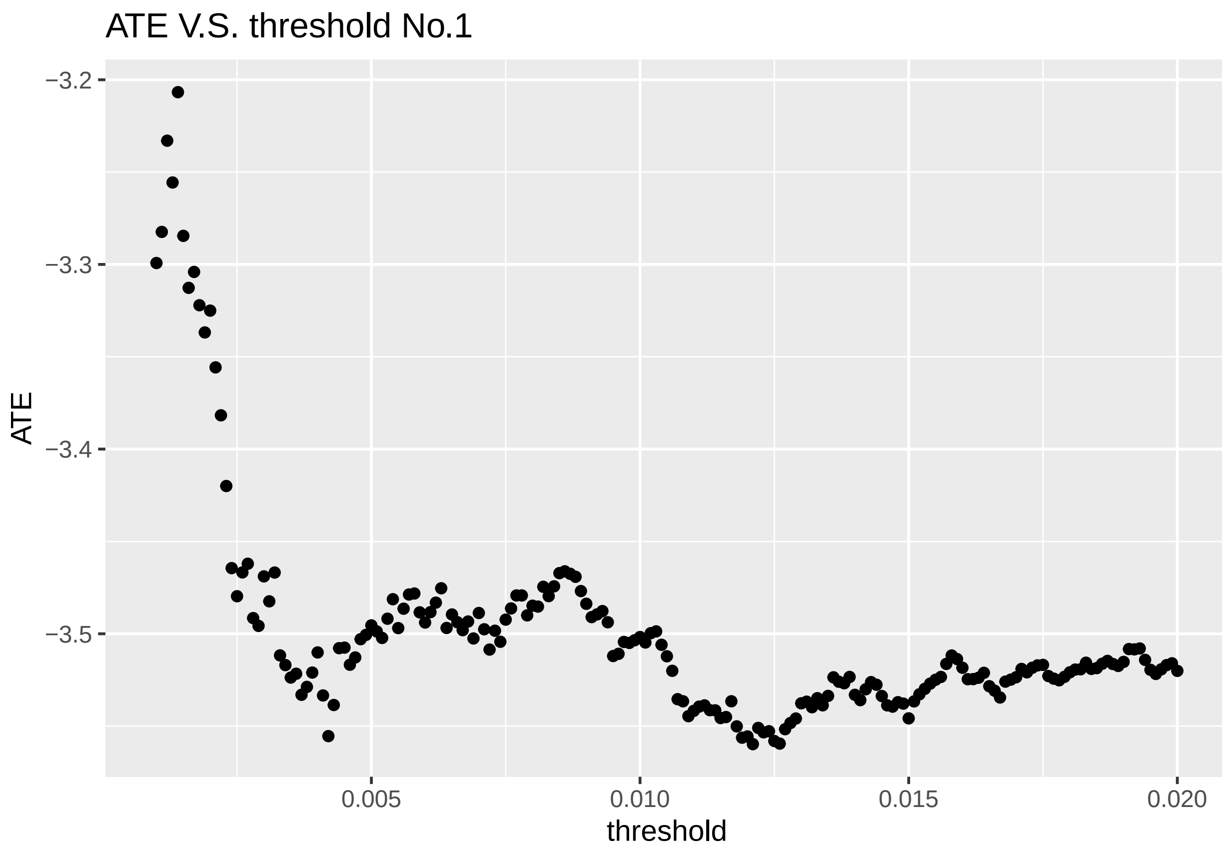

ATE V.S. threshold No.4
ATE
2.8
2.4
2.0
0.005
0.010
0.015
0.020
threshold

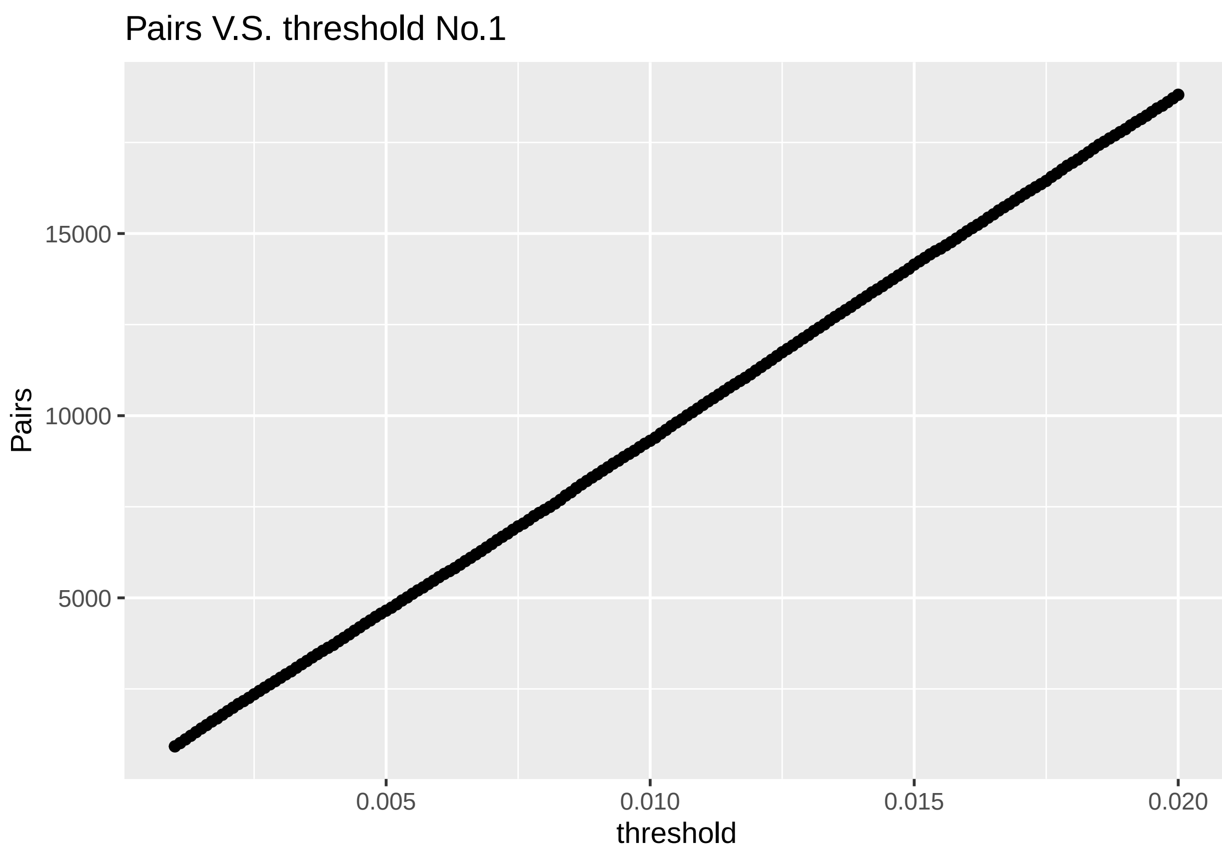


4.6 Plotting Part for High-Dim

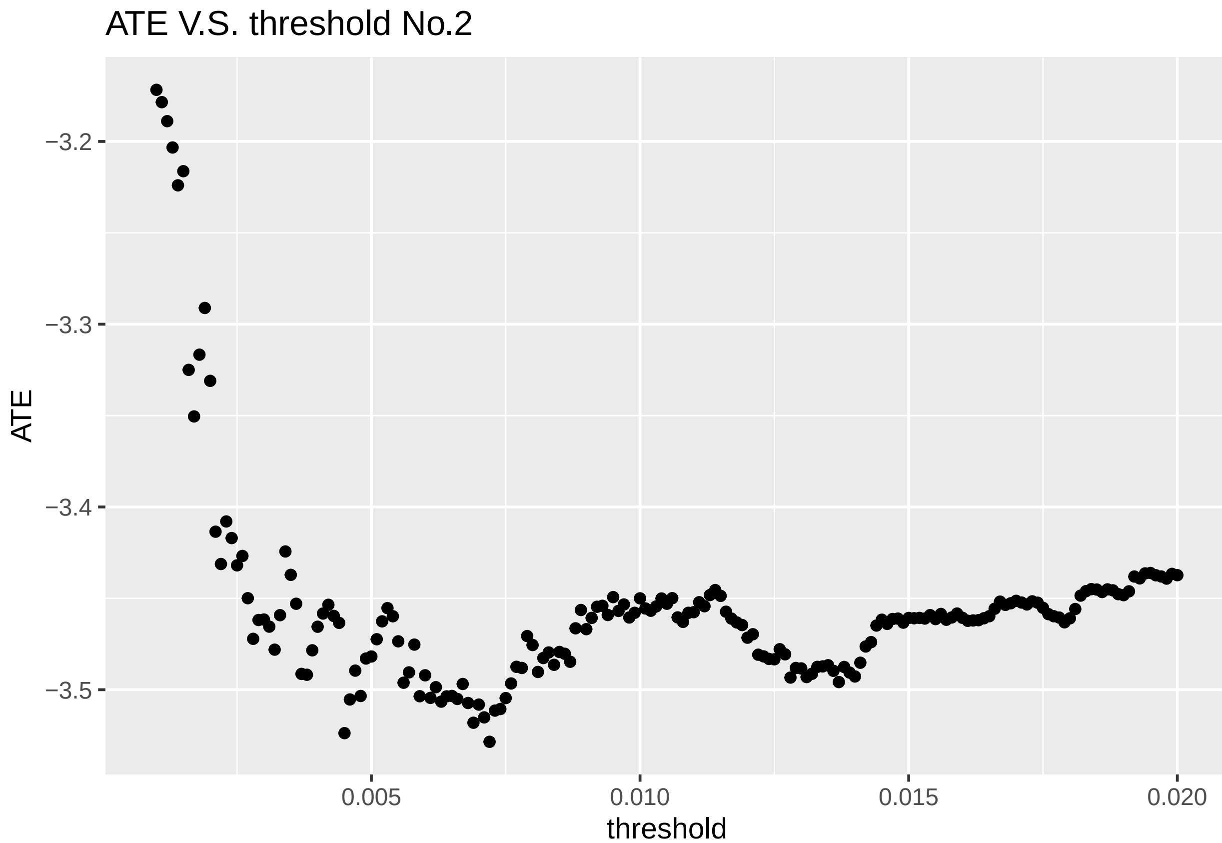
```
for (i in ind_mat){
  ATE_high <- high_list[[i]]$ate
  pairs_high <- high_list[[i]]$pair

  plot_high <- data.frame(x=seq,ATE=ATE_high,pairs=pairs_high)

  g_high <- ggplot(plot_high)+
    geom_point(aes(x,ATE))+
    labs(
      title = paste0("ATE V.S. threshold No.",i),
      x = "threshold",
      y = "ATE"
    )
  print(g_high)

  match_high <- ggplot(plot_high)+
    geom_point(aes(x,pairs))+
    labs(
      title = paste0("Pairs V.S. threshold No.",i),
      x = "threshold",
      y = "Pairs"
    )
```

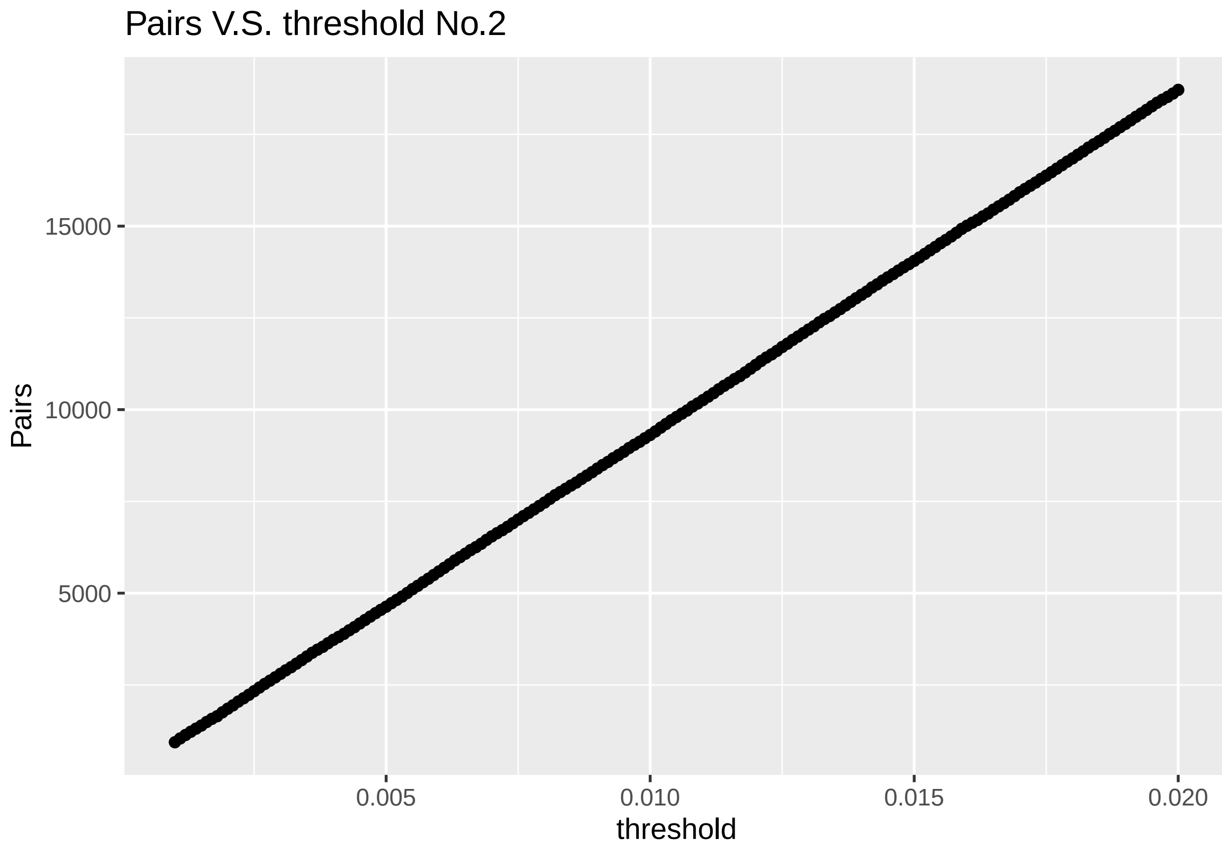
```
  print(match_high)

}
```

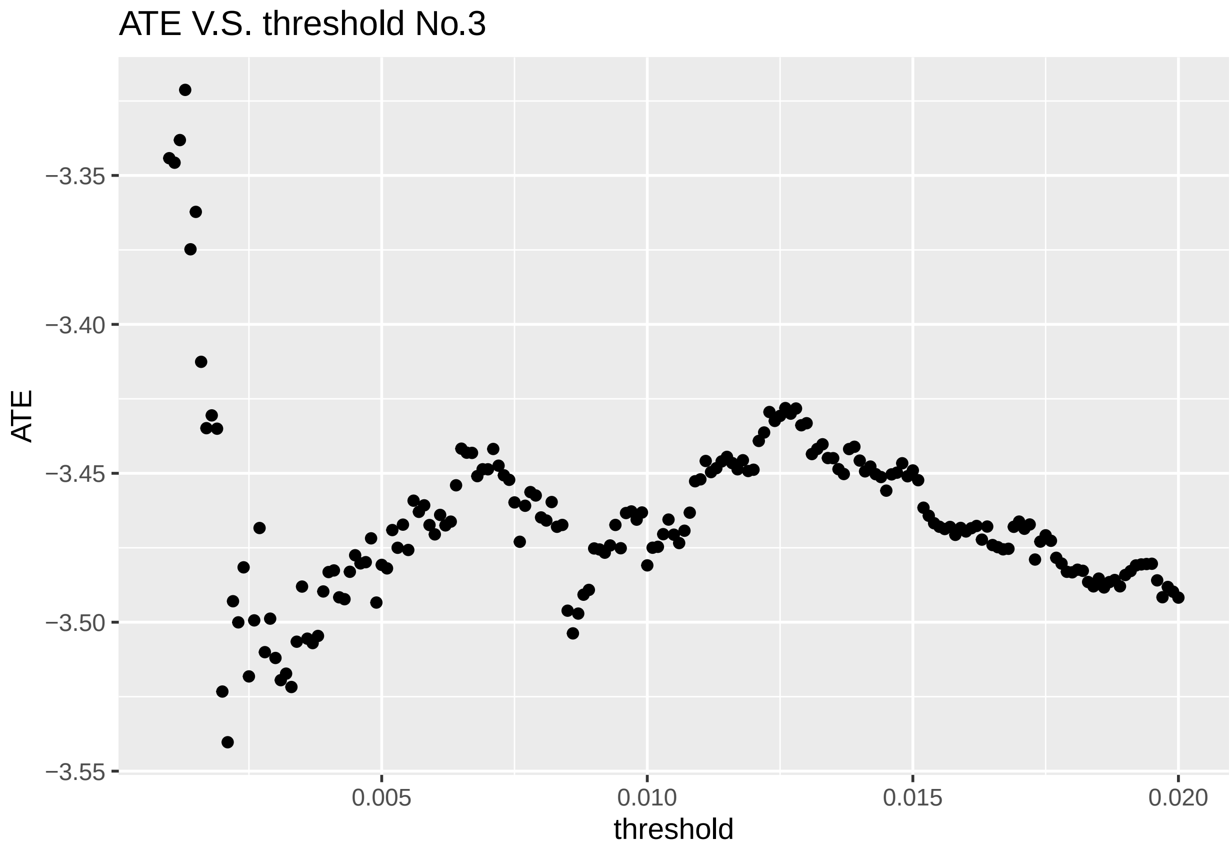


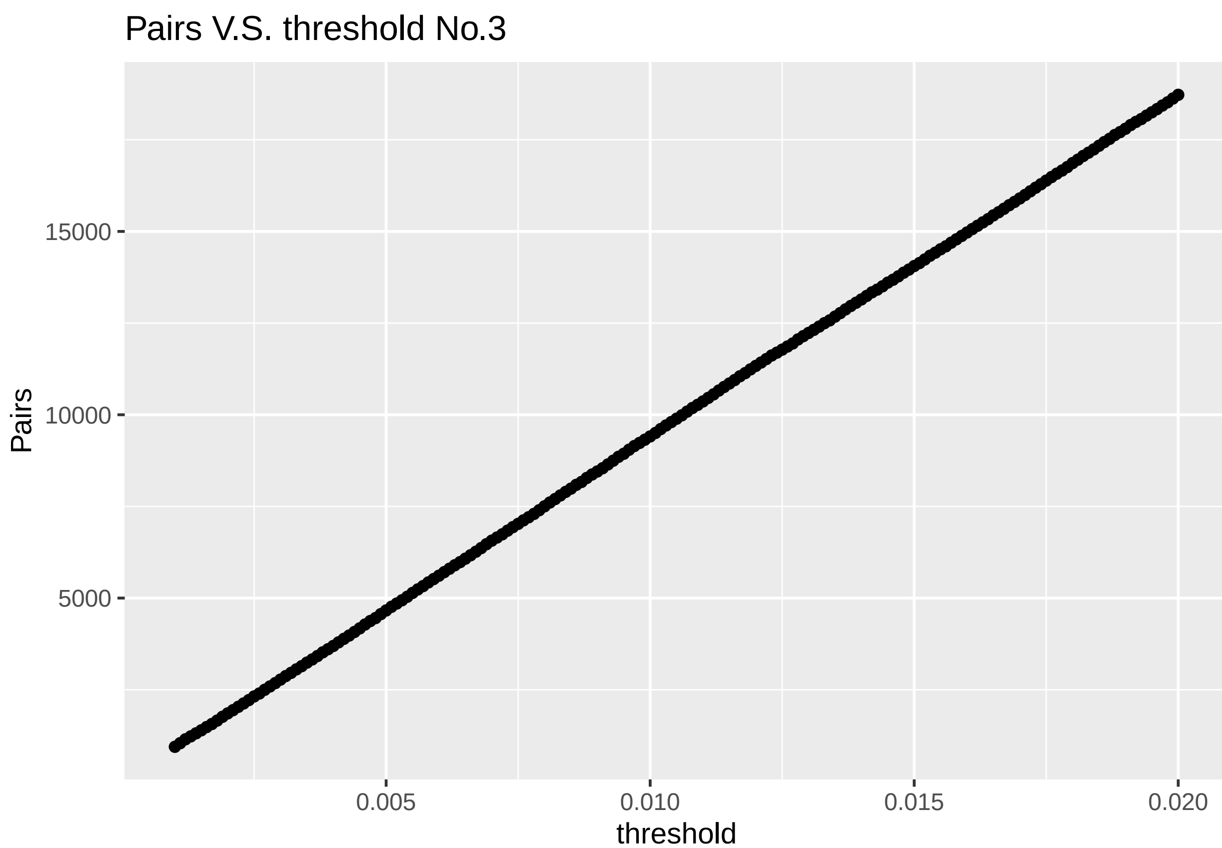


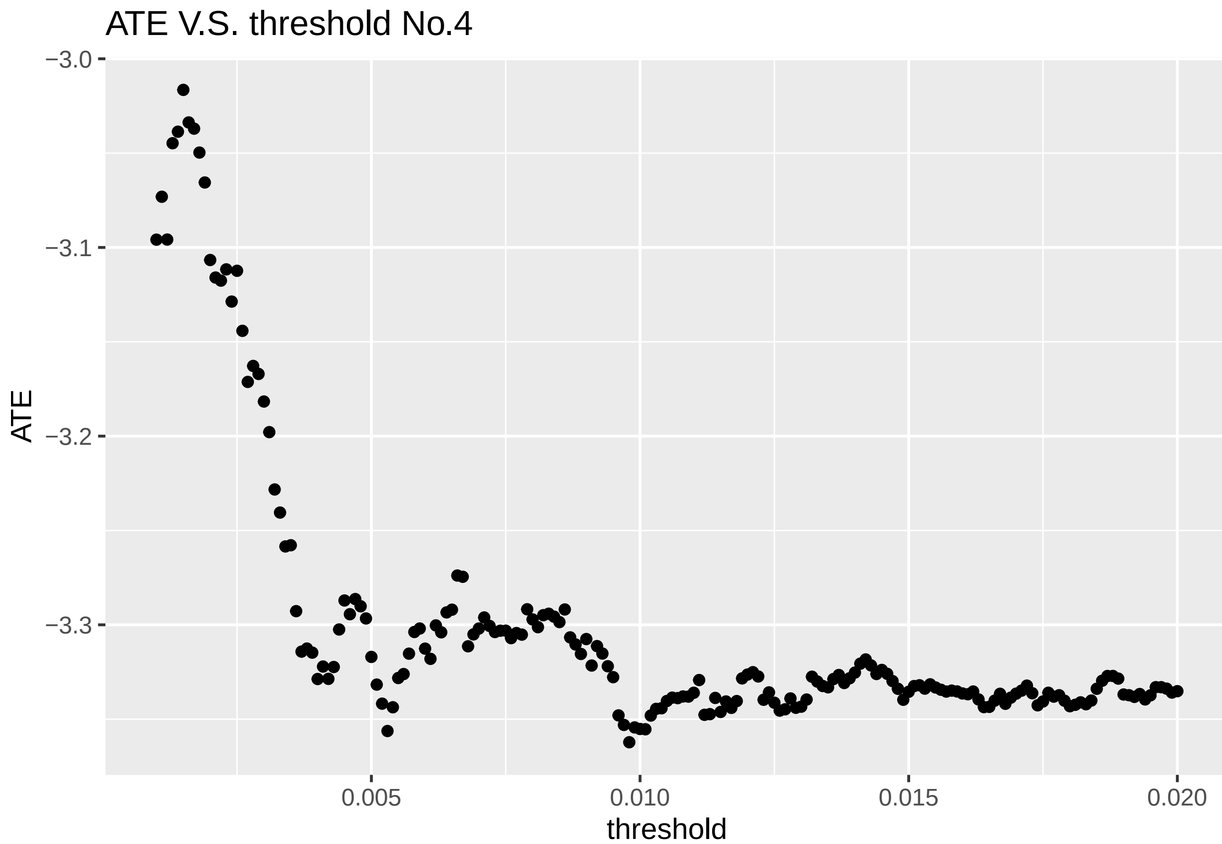
Pairs V.S. threshold No.1
Pairs
15000
10000
5000
0.005
0.010
0.015
0.020
threshold

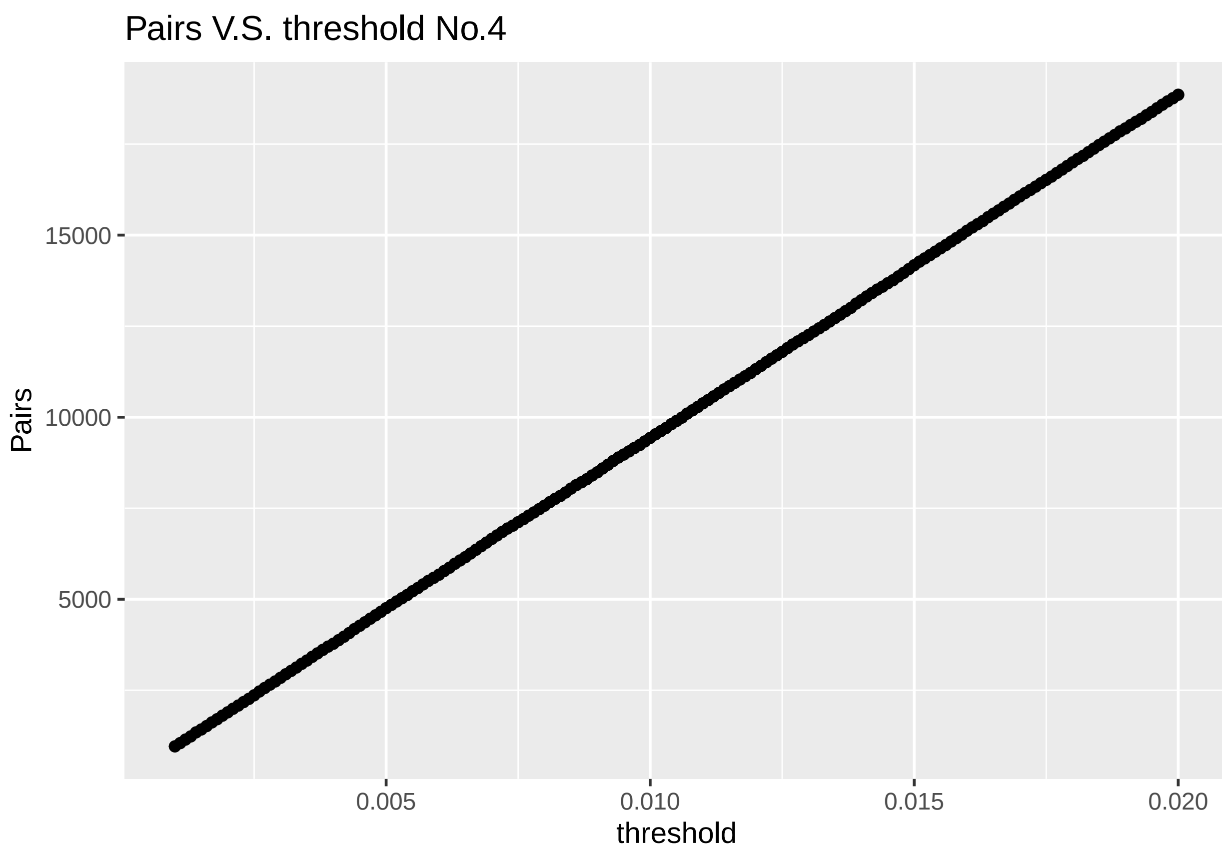

ATE V.S. threshold No.2
−3.2
−3.3
−3.4
−3.5
ATE
0.005
0.010
0.015
0.020
threshold


Pairs V.S. threshold No.2
15000
10000
5000
Pairs
0.005
0.010
0.015
0.020
threshold


ATE V.S. threshold No.3
ATE
−3.35
−3.40
−3.45
−3.50
−3.55
0.005
0.010
0.015
0.020
threshold


Pairs V.S. threshold No.3
15000
10000
5000
Pairs
0.005
0.010
0.015
0.020
threshold


ATE V.S. threshold No.4
−3.0
−3.1
−3.2
−3.3
ATE
0.005
0.010
0.015
0.020
threshold



5. Inverse Propensity Weighting

5.1 Introduction of Inverse Propensity Weighting

Propensity score weighting is an alternative to propensity score matching in casual inference. Its idea, in the context of this project, is to directly use propensity scores as inverse weights in calculating the ATE.

Individuals from the treatment group are weighted as $\frac{1}{\hat{e}_i}$, whereas individuals from the control group are weighted as $\frac{1}{1-\hat{e}_i}$, where $\hat{e}_i$ is the estimated propensity score for individual i.

Such an approach addresses some of the disadvantages inherent to propensity score matching. First, it may be impossible to pair each treatment to a different control, or even to any control. Secondly, grouping may include too few controls at high propensities to yield reliable group mean differences. Lastly, groups can be so coarse that the controls and treatments in a group are not well-matched.

```
data_low <- ldim
data_high <- hdim

# Low Dimension
treatment_low <- data_low$A
y_low <- data_low$Y
x_low <- data_low[, -c(1,2)]

# High Dimension
treatment_high <- data_high$A
```

```
y_high <- data_high$Y
x_high <- data_high[, -c(1,2)]
```

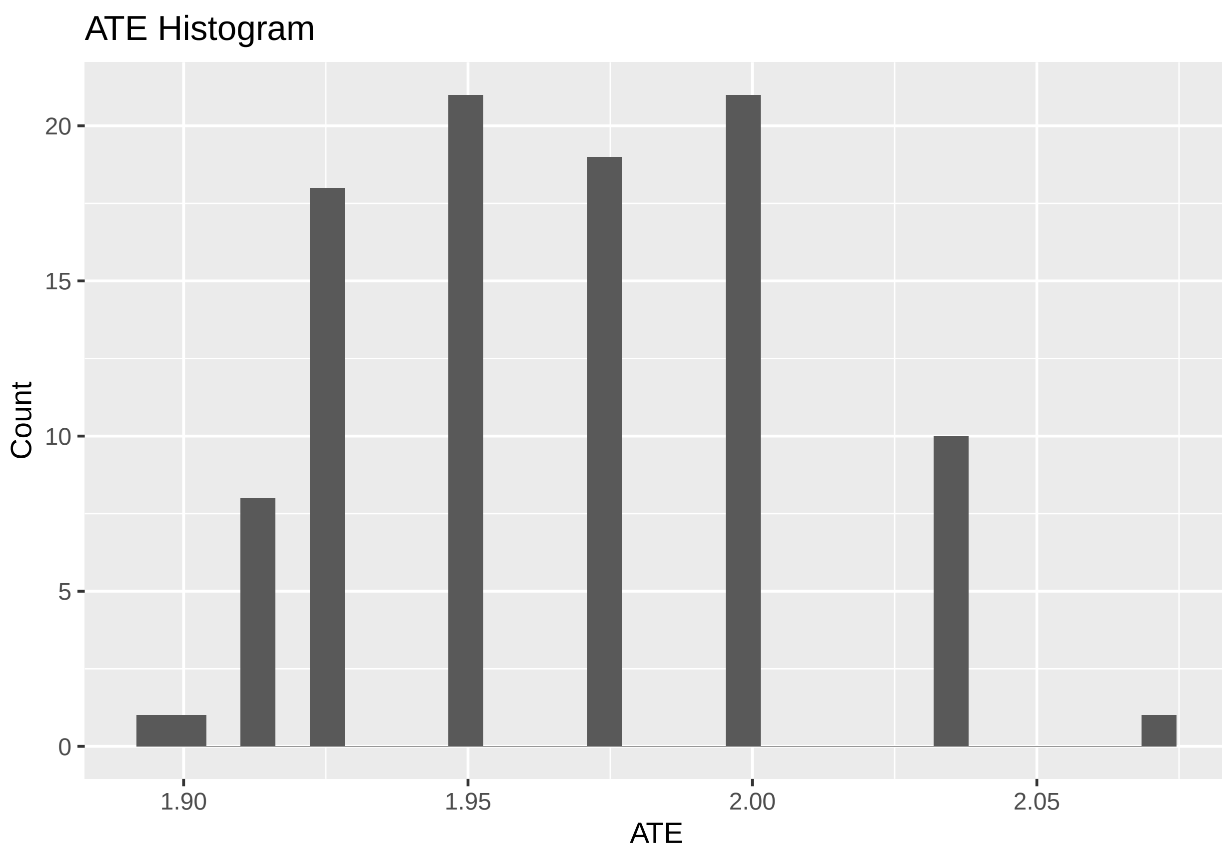
5.2 Calculating ATE for Low-Dim with bootstrap

```
start_time <- Sys.time()

set.seed(0)
seed <- sample(1:10000,100)

ate_ipw_vec <- vector("double")
for (seednum in seed){
  set.seed(seednum)
  glm_low <- cv.glmnet(as.matrix(x_low), treatment_low, family = "binomial", alpha = 0)

  ps_low <- predict(glm_low$glmnet.fit,
                    s = glm_low$lambda.min,
                    newx = as.matrix(x_low),
                    type = "response")
  data_low$ps <- ps_low


  data_low$inv_prop_weight <- ifelse(data_low$A == 1, 1/data_low$ps,
                                     1/(1 - data_low$ps))


  data_low_treatment <- data_low[which(data_low$A == 1), ]
  data_low_control <- data_low[which(data_low$A == 0), ]
  ATE_low <- (sum(data_low_treatment$inv_prop_weight * data_low_treatment$Y) -
                sum(data_low_control$inv_prop_weight * data_low_control$Y)) /
    dim(data_low)[1]

  ate_ipw_vec <- append(ate_ipw_vec,ATE_low)
}
```

5.3 ATE Hist Plot for Low-Dim

```
plot_low_ipw <- data.frame(x=1:length(seed),ATE=ate_ipw_vec)

g_low <- ggplot(plot_low_ipw)+
  geom_histogram(aes(ATE))+
  labs(
    title = paste0("ATE Histogram"),
    x = "ATE",
    y = "Count"
  )
print(g_low)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
end_time <- Sys.time()


cat("mean ATE = ",mean(ate_ipw_vec))
```

```
## mean ATE =  1.965807
```

```
cat("std ATE = ",sd(ate_ipw_vec))
```

```
## std ATE =  0.03833074
```

```
tm <- end_time - start_time
cat("Time for IPW Low-Dim is: ")
```

```
## Time for IPW Low-Dim is:
```

```
tm
```

```
## Time difference of 30.55518 secs
```

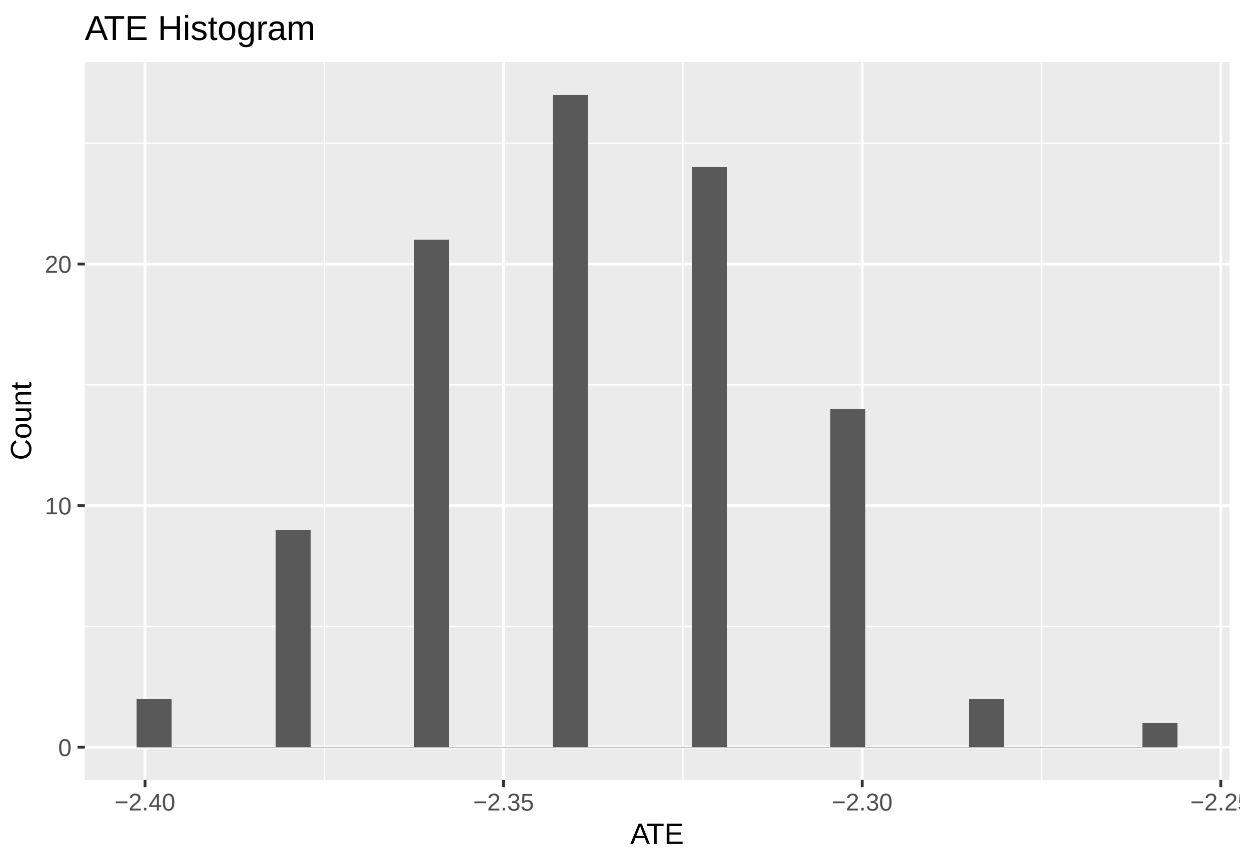
5.4 Calculating ATE for High-Dim with bootstrap

```
start_time <- Sys.time()

set.seed(0)
seed <- sample(1:10000,100)

ate_ipw_vec <- vector("double")
for (seednum in seed){
  set.seed(seednum)

  glm_high <- cv.glmnet(as.matrix(x_high), treatment_high, family = "binomial", alpha = 0)

  ps_high <- predict(glm_high$glmnet.fit,
                     s = glm_high$lambda.min,
                     newx = as.matrix(x_high),
                     type = "response")
  data_high$ps <- ps_high

  data_high$inv_prop_weight <- ifelse(data_high$A == 1,
                                      1/data_high$ps,
                                      1/(1 - data_high$ps))

  data_high_treatment <- data_high[which(data_high$A == 1), ]
  data_high_control <- data_high[which(data_high$A == 0), ]
  ATE_high <- (sum(data_high_treatment$inv_prop_weight * data_high_treatment$Y) -
               sum(data_high_control$inv_prop_weight * data_high_control$Y)) /
    dim(data_high)[1]

  ate_ipw_vec <- append(ate_ipw_vec,ATE_high)
}
```

5.5 ATE Hist Plot for High-Dim

```
plot_low_ipw <- data.frame(x=1:length(seed),ATE=ate_ipw_vec)

g_high <- ggplot(plot_low_ipw)+
  geom_histogram(aes(ATE))+
  labs(
    title = paste0("ATE Histogram"),
    x = "ATE",
    y = "Count"
  )
print(g_high)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```r
cat("mean ATE = ",mean(ate_ipw_vec))
```

```
## mean ATE =  -2.338619
```

```r
cat("std ATE = ",sd(ate_ipw_vec))
```

```
## std ATE =  0.02712537
```

```r
tm <- end_time - start_time
cat("Time for IPW High-Dim is: ")
```

```
## Time for IPW High-Dim is:
```

```r
tm
```

```
## Time difference of -0.08164406 secs
```

6. Doubly Robust Estimation

6.1 Introduction of Doubly Robust Estimation

Doubly Robust Estimation combines the predicted outcome from linear regression with propensity score to estimate the causal effect. Without combining, these two methods can be easily biased. Doubly Robust

Estimation reduces the likelihood to be biased since only one of the 2 models need to be correctly specified to obtain unbiased estimator.

```
low <- read.csv("../data/lowDim_dataset.csv")
high <- read.csv("../data/highDim_dataset.csv")
# Low Dimension
lowA <- low$A
lowY <- low$Y
lowData <- low[,-c(1,2)]

# High Dimension
highA <- high$A
highY <- high$Y
highData <- high[, -c(1,2)]
```

6.2 ATE calculated with Doubly Robust Estimation Low-Dim

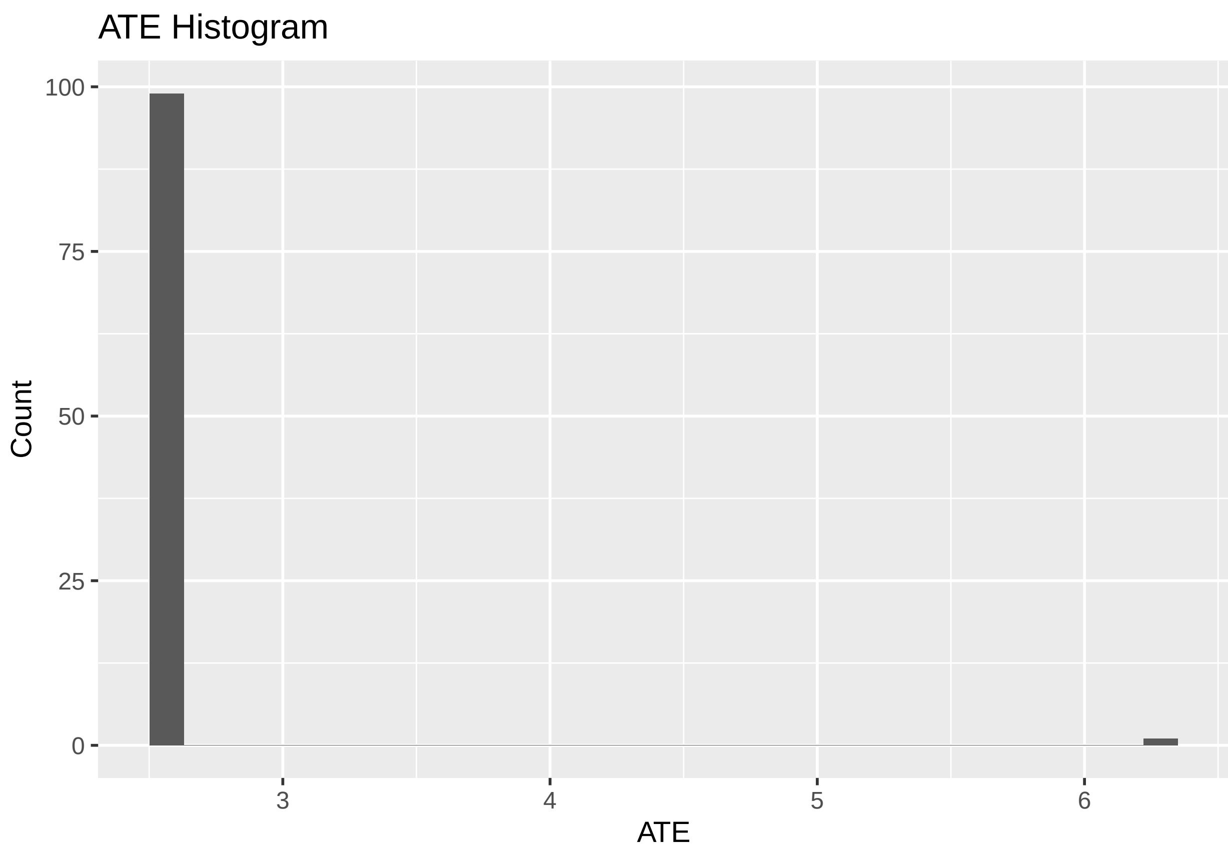
```
start_time <- Sys.time()

set.seed(0)
seed <- sample(1:10000,100)

ate_ipw_vec <- vector("double")
for (seednum in seed){
  set.seed(seednum)

  glm_low <- cv.glmnet(as.matrix(lowData), lowA, family = "binomial", alpha = 0)

  psLow <- predict(glm_low$glmnet.fit,
                   s = glm_low$lambda.min,
                   newx = as.matrix(lowData),
                   type = "response")
  low$ps <- psLow

  low1 <- low[which(low$A=='1'),]
  low0 <- low[which(low$A=='0'),]
  lr_low1 <- glm(formula=Y~ .,data = low1)
  lr_low0 <- glm(formula=Y~ .,data = low0)
  low$m1 <- predict(lr_low1,low[,-c(1)])
  low$m0 <- predict(lr_low0,low[,-c(1)])
  # Calculate
  ATE_low <- sum((low$A*low$Y-(low$A-low$ps)*low$m1)/low$ps)/dim(low)[1]-sum(((1-low$A)*low$Y+(low$A-lo



  ate_ipw_vec <- append(ate_ipw_vec,ATE_low)
}
```

The first chunk of this code is for building linear regression models based on output(Y) and features(low1) for different treatment(A) values. The following part uses all the value we obtain before to calculate ATE.

6.3 ATE Hist Plot for Low-Dim

```r
plot_low_ipw <- data.frame(x=1:length(seed),ATE=ate_ipw_vec)

g_low <- ggplot(plot_low_ipw)+
  geom_histogram(aes(ATE))+
  labs(
    title = paste0("ATE Histogram"),
    x = "ATE",
    y = "Count"
  )
print(g_low)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```r
end_time <- Sys.time()


cat("mean ATE = ",mean(ate_ipw_vec))
```

```
## mean ATE =  2.597958
```

```r
cat("std ATE = ",sd(ate_ipw_vec))
```

```
## std ATE =  0.3710094
```

```r
tm <- end_time - start_time
cat("Time for IPW Low-Dim is: ")
```

```
## Time for IPW Low-Dim is:
```

```r
tm
```

```
## Time difference of 31.51484 secs
```

6.4 ATE calculated with Doubly Robust Estimation High-Dim

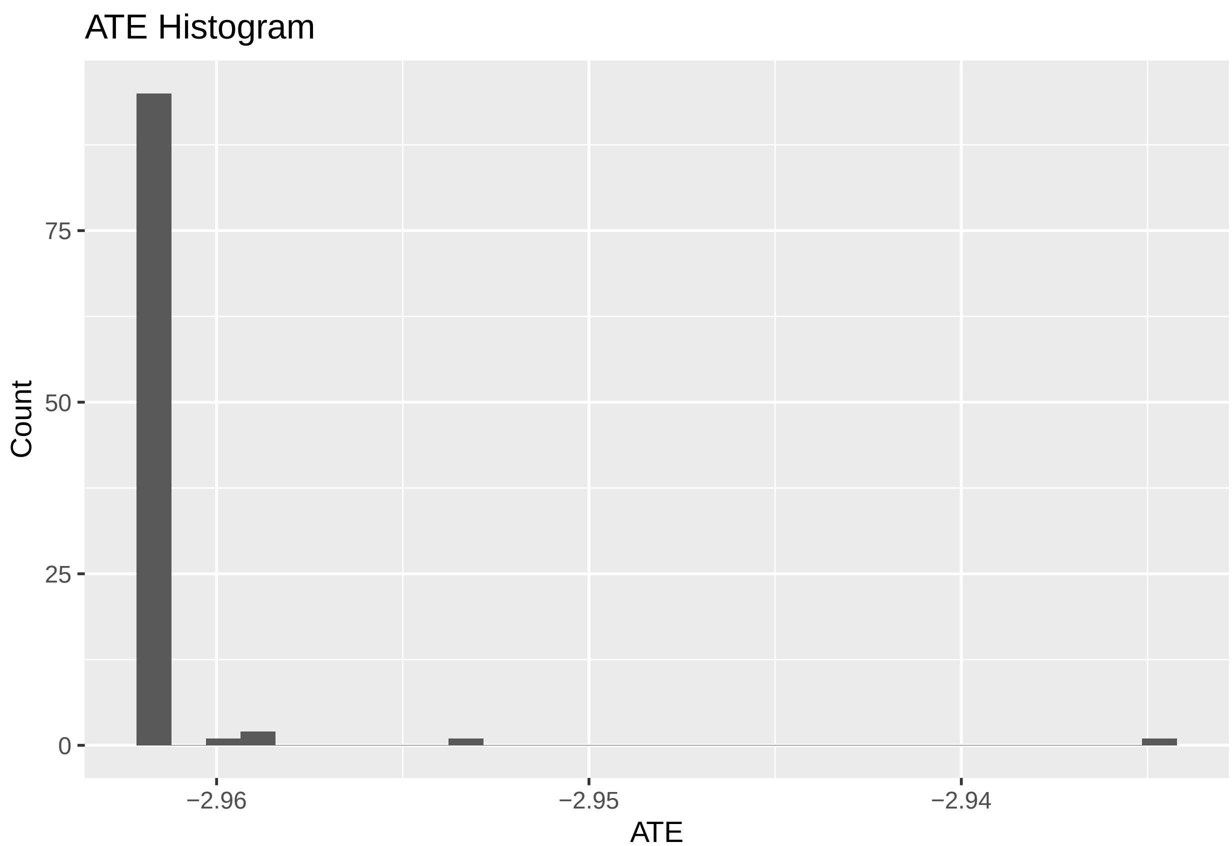
```r
start_time <- Sys.time()

set.seed(0)
seed <- sample(1:10000,100)

ate_ipw_vec <- vector("double")
for (seednum in seed){
  set.seed(seednum)

  glm_high <- cv.glmnet(as.matrix(highData), highA, family = "binomial", alpha = 0)

  psHigh <- predict(glm_high$glmnet.fit,
                    s = glm_high$lambda.min,
                    newx = as.matrix(highData),
                    type = "response")
  high$ps <- psHigh

  high1 <- high[which(high$A=='1'),]
  high0 <- high[which(high$A=='0'),]
  lr_high1 <- glm(formula=Y~ .,data = high1)
  lr_high0 <- glm(formula=Y~ .,data = high0)
  high$m1 <- predict(lr_high1,high[,-c(1)])
  high$m0 <- predict(lr_high0,high[,-c(1)])
  # Calculate
  ATE_high <- sum((high$A*high$Y-(high$A-high$ps)*high$m1)/high$ps)/dim(high)[1]-sum(((1-high$A)*high$Y



  ate_ipw_vec <- append(ate_ipw_vec,ATE_high)
}
```

The steps to calculate the ATE for high dimension data is exactly the same. The first part is for building linear regression models to predict outcomes and the second part is to calculate ATE.

6.3 ATE Hist Plot for High-Dim

```
plot_low_ipw <- data.frame(x=1:length(seed),ATE=ate_ipw_vec)

g_low <- ggplot(plot_low_ipw)+
  geom_histogram(aes(ATE))+
  labs(
    title = paste0("ATE Histogram"),
    x = "ATE",
    y = "Count"
  )
print(g_low)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
end_time <- Sys.time()

cat("mean ATE = ",mean(ate_ipw_vec))
```

```
## mean ATE =  -2.960964
```

```r
cat("std ATE = ",sd(ate_ipw_vec))
```

```
## std ATE =  0.002820911
```

```r
tm <- end_time - start_time
cat("Time for IPW Low-Dim is: ")
```

```
## Time for IPW Low-Dim is:
```

```r
tm
```

```
## Time difference of 10.75147 mins
```