# Project 4: Causal Inference Algorithms Evaluation

Group 4

Jingbin Cao, Pin-Chun Chen, Aurore Gosmant, Weiwei Song, Zikun Zhuang

## Basic Setup

```
#Test Branch created
if(!require("readr")){
  install.packages("readr")
}
if(!require("tidyverse")){
  install.packages("tidyverse")
}
if(!require("glmnet")){
  install.packages("glmnet")
}
if(!require("pryr")){
  install.packages("pryr")
}

library(readr)
library(tidyverse)
library(glmnet)
library(pryr)

lowDim_raw <- read_csv('../data/lowDim_dataset.csv')
highDim_raw <- read_csv('../data/highDim_dataset.csv')

lowDim <- lowDim_raw
highDim <- highDim_raw
```

## Project Overview

We will evaluate four inference algorithm in this project, Inverse Propensity Weighting (IPW) + Logistic Regression, Regression Estimate, Stratification + Logistic Regression, and Regression Adjustment + Logistic Regression. We will compute the average treatment effect (ATE) using the four algorithms on two distinct datasets (low dimension & high dimension), and we will compare their performance and computational efficiency.

## Getting Propensity Scores

We estimate the propensity scores using logistic regression:

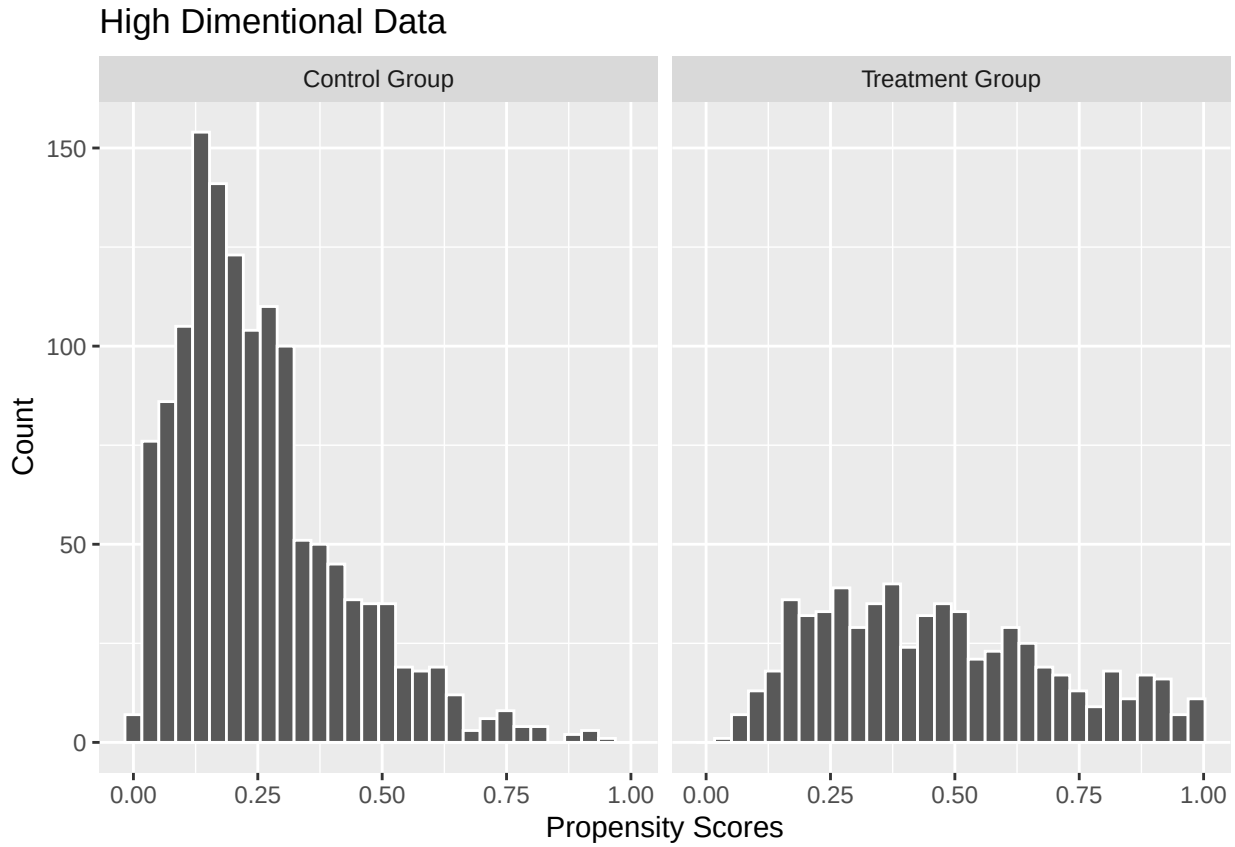
$$logit[Pr(T=1|X)] = \beta_0 + \beta_1 x_1 + ... + \beta_p x_p$$

$$Pr(T=1|X) = \frac{1}{1+e^{-(\beta_0+\beta_1 x_1+...+\beta_p x_p)}}$$

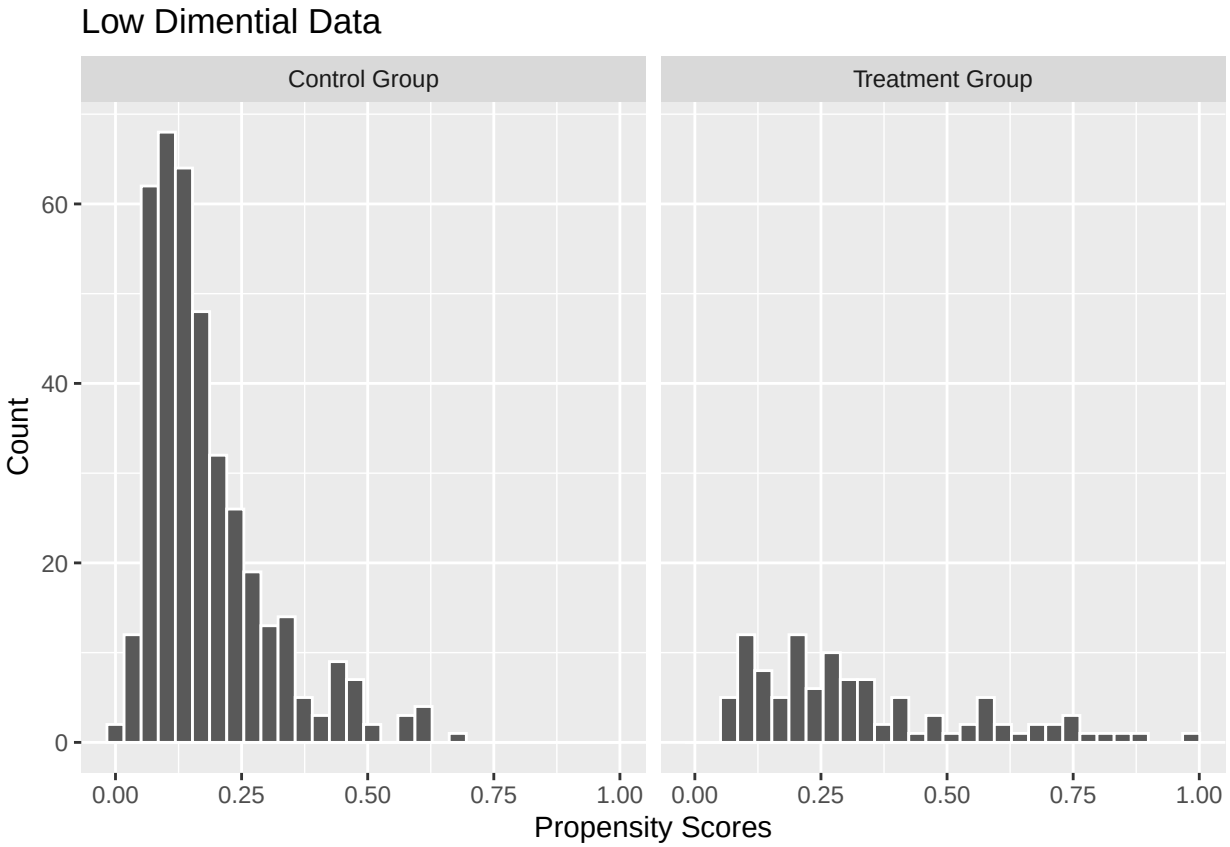
```
# High Dimentional Data
ps_high_estimate <- glm(data = highDim,
                        formula = A~ . -Y,
                        family=binomial())

ps_high_data <- data.frame(ps = predict(ps_high_estimate,type="response"),
                           treatment = ps_high_estimate$model$A)


# Low Dimentional Data
ps_low_estimate <- glm(data = lowDim,
                       formula = A~ . -Y,
                       family=binomial())

ps_low_data <- data.frame(ps = predict(ps_low_estimate,type="response"),
                          treatment = ps_low_estimate$model$A)

# Visualize Data
## Show PS means
(ps_means <- data.frame(high_treat_ps_mean = mean(ps_high_data[ps_high_data$treatment==1,]$ps),
                        high_control_ps_mean = mean(ps_high_data[ps_high_data$treatment==0,]$ps),
                        low_treat_ps_mean = mean(ps_low_data[ps_low_data$treatment==1,]$ps),
                        low_control_ps_mean = mean(ps_low_data[ps_low_data$treatment==0,]$ps)))
```

```
##   high_treat_ps_mean high_control_ps_mean low_treat_ps_mean
## 1          0.4663911            0.2528449         0.3290442
##   low_control_ps_mean
## 1           0.1805109
```

```
## Plot Counts
ps_high_data %>%
  mutate(treatment = ifelse(treatment == 0, "Control Group", "Treatment Group")) %>%
  ggplot(aes(x=ps))+
  ggtitle("High Dimentional Data")+
  xlab("Propensity Scores")+
  ylab("Count")+
  geom_histogram(color="white")+
  facet_wrap(~treatment)
```




```
ps_low_data %>%
  mutate(treatment = ifelse(treatment == 0, "Control Group", "Treatment Group")) %>%
  ggplot(aes(x=ps))+
  ggtitle("Low Dimential Data")+
  xlab("Propensity Scores")+
  ylab("Count")+
  geom_histogram(color="white")+
  facet_wrap(~treatment)
```




## Model 1: Inverse Propensity Weighting and Logistic Regression

$$w_i = \frac{T_i}{\hat{e_i}} + \frac{1 - T_i}{1 - \hat{e_i}}$$

where $\hat{e_i}$ is the estimated propensity score for individual $i$.
Estimate ATE:

$$\hat{\triangle}_{IPW} = N^{-1}(\sum_{i \in Treated} w_i Y_i - \sum_{i \in Controlled} w_i Y_i)$$

where the first summation is from the treated group, and the second summation is from the controlled group.

```
set.seed(0)
# Define Data

# Write Algorithm
IPW <- function(df,ps){
  start <- Sys.time()
  ps['weights'] <- df$A/ps$ps+(1-df$A)/(1-ps$ps)
  treatment <- sum(ps[ps$treatment==1,]$weights*df$Y[df$A==1])
  controll <- sum(ps[ps$treatment==0,]$weights*df$Y[df$A==0])
  ATE <- (treatment-controll)/nrow(df)
  end <- Sys.time()
  runtime = end - start
```

```
  return(list(ATE=ATE,runtime=runtime))
}
# Output Performance
matrix(c(IPW(highDim,ps_high_data)$ATE,
         IPW(lowDim,ps_low_data)$ATE,
         IPW(highDim,ps_high_data)$runtime,
         IPW(lowDim,ps_low_data)$runtime),
       nrow = 2,byrow = TRUE,
       dimnames = list(c("ATE","Running Time (secs)"), c("High Dimension","Low Dimension")))
```

```
##                       High Dimension Low Dimension
## ATE                        -57.57303      1.689329
## Running Time (secs)          0.00000      0.000000
```

## Model 2: Regression Estimate

$$\hat{\triangle}_{reg} = N^{-1} \sum_{i=1}^{N} (\hat{m}_1 - \hat{m}_0(X_i))$$

No need for Propensity Score.

```
set.seed(0)
# Write Algorithm
Regression_Estimate <- function(df){
  df_X <- df%>% select(-Y,-A)
  start <- Sys.time()
  model_0 <- glm(Y ~ ., data = subset(df[df$A==0,], select = -A))
  model_1 <- glm(Y ~ ., data = subset(df[df$A==1,], select = -A))
  Y0_pred <- predict(model_0, newdata = df%>% select(-Y,-A))
  Y1_pred <- predict(model_1, newdata = df%>% select(-Y,-A))
  ATE = 1/nrow(df) * sum(Y1_pred - Y0_pred)
  end <- Sys.time()
  runtime = end - start
  return(list(ATE = ATE, runtime = runtime))
}

# Output Performance
matrix(c(Regression_Estimate(highDim)$ATE,
         Regression_Estimate(lowDim)$ATE,
         Regression_Estimate(highDim)$runtime,
         Regression_Estimate(lowDim)$runtime),
         nrow = 2,
         byrow = TRUE,
         dimnames = list(c("ATE","Running Time (secs)"), c("High Dimension","Low Dimension")))
```

```
##                       High Dimension Low Dimension
## ATE                      -57.4265873    2.12513807
## Running Time (secs)        0.1918159    0.01695395
```

## Model 3: Stratification and Logistic Regression

$$\hat{\triangle}_S = \sum_{j=1}^{K} \frac{N_j}{N} (N_{1j}^{-1} \sum_{i=1}^{N} T_i Y_i I(\hat{e}_i \in \hat{Q}_j) - N_{0j}^{-1} \sum_{i=1}^{N} (1 - T_i) Y_i I(\hat{e}_i \in \hat{Q}_j))$$

where $K$ is the number of strata (K=5).
$N_j$ is the number of individuals in stratum $j$.
$N_{1j}$ is the number of "treated" individuals in stratum $j$, and $N_{0j}$ is the number of "controlled" individuals in stratum $j$.
$\hat{Q}_j = (\hat{q}_{j-1}, \hat{q}_j)$ where $\hat{q}_j$ is the jth sample quantile of the estimated propensity scores.

Some brief introduction...

```
set.seed(0)
# Define Data

# Write Algorithm
Strat <- function(df, ps){
  start = Sys.time()
  ATE <- 0
  prev <- 0
  cancatdata <- cbind(df, ps)
  for (i in 1:4) {
    ps_str = as.numeric(ps$ps)
    str_q <- quantile(ps_str, 0.25*i)
    str_data <- cancatdata[cancatdata$ps>=prev,]
    str_data <- str_data[str_q>=str_data$ps,]
    prev = str_q
    temp_t <- sum(str_data$Y[str_data$A==1])
    temp_t <- 1/nrow(str_data[str_data$A==1,])*temp_t
    temp_c <- sum(str_data$Y[str_data$A==0])
    temp_c <- 1/nrow(str_data[str_data$A==0,])*temp_c
    str_e <- temp_t - temp_c
    ATE <- ATE + str_e*nrow(str_data)/nrow(df)
  }
  end = Sys.time()
  runtime = end - start
  return(list(ATE = ATE, runtime = runtime))
}
# Output Performance
matrix(c(Strat(highDim,ps_high_data)$ATE,
         Strat(lowDim,ps_low_data)$ATE,
         Strat(highDim,ps_high_data)$runtime,
         Strat(lowDim,ps_low_data)$runtime),
       nrow = 2,
       byrow = TRUE,
       dimnames = list(c("ATE","Running Time (secs)"), c("High Dimension","Low Dimension")))
```

```
##                      High Dimension Low Dimension
## ATE                     -60.38900682   2.667086580
## Running Time (secs)       0.04787111   0.007977962
```

## Model 4: Regression Adjustment and Logistic Regression

Regress the outcome variable Y on treatment indicator variable T and the estimated propensity score. Then, the estimated coefficient on the treatment indicator variable would be an estimate of ATE.
In this method, we regress the response variable ($Y$) with the treatment variable ($A$) and the propensity scores estimated using our model above, in this case, logistic regression. The estimated coefficient of the treatment variable ($A$) is then an estimate of the ATE.

D'Agostino (1998) and Austin (2011) compare regression adjustment with more traditional propensity score methods. One of the main advantages of the regression adjustment is in its simplicity in execution, in which one performs a somewhat basic linear regression model on two covariates and one response variable.

However, depending on the size of the dataset, this may run into computation issues as linear regression involves finding the inverse of a matrix. Additionally, regression adjustment may also not be helpful in cases where there is a strong separation between the two groups.

No such issues were present in this setup given that both datasets had a relatively small number of observations and there is no clear separation between the two groups

```
set.seed(0)
# Define Data

# Write Algorithm
Reg_adj <- function(df,ps_data){
  df<- data.frame(cbind(Y=df$Y,A=df$A,ps=ps_data$ps))
  start <- Sys.time()
  m<- lm(Y ~ A+ ps, data = df)
  ATE = m$coefficients[2]
  end <- Sys.time()
  runtime = end - start
  return(list(ATE = ATE, runtime = runtime))
}
# Output Performance
matrix(c(Reg_adj(highDim,ps_high_data)$ATE,
         Reg_adj(lowDim,ps_low_data)$ATE,
         Reg_adj(highDim,ps_high_data)$runtime,
         Reg_adj(lowDim,ps_high_data)$runtime),
       nrow = 2,
       byrow = TRUE,
       dimnames = list(c("ATE","Running Time (secs)"), c("High Dimension","Low Dimension")))
```

```
##                      High Dimension Low Dimension
## ATE                   -59.416700238     2.4180865
## Running Time (secs)     0.002053022     0.0029881
```

## Model Comparisons

Performance is measured by the squared difference of true ATE and estimated ATE Run times are in seconds

### Low Dimension Dataset

| Model | ATE | Run_Time | Performance |
|---|---|---|---|
| True ATE | 2.090100 | N.A | N.A. |
| IPW + LR | 1.689329 | 0 | 0.633064880516225 |
| Regression Estimate | 2.125138 | 0.0139620304107666 | 0.187184601273571 |
| Stratification + LR | 2.667087 | 0.00399112701416016 | 0.75959632724015 |
| Regression Adjustment + LR | 2.418086 | 0 | 0.572701016363988 |

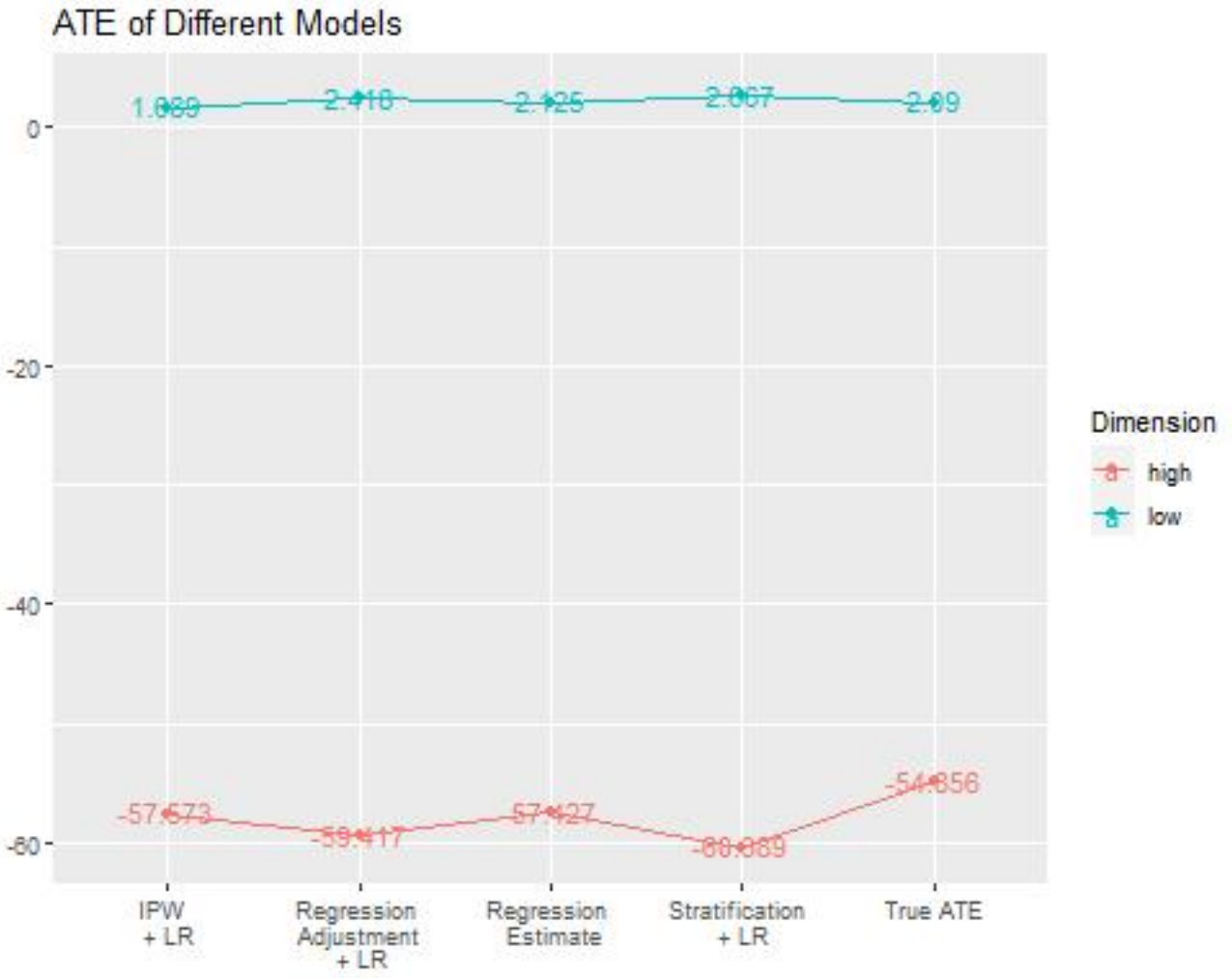
### High Dimension Dataset

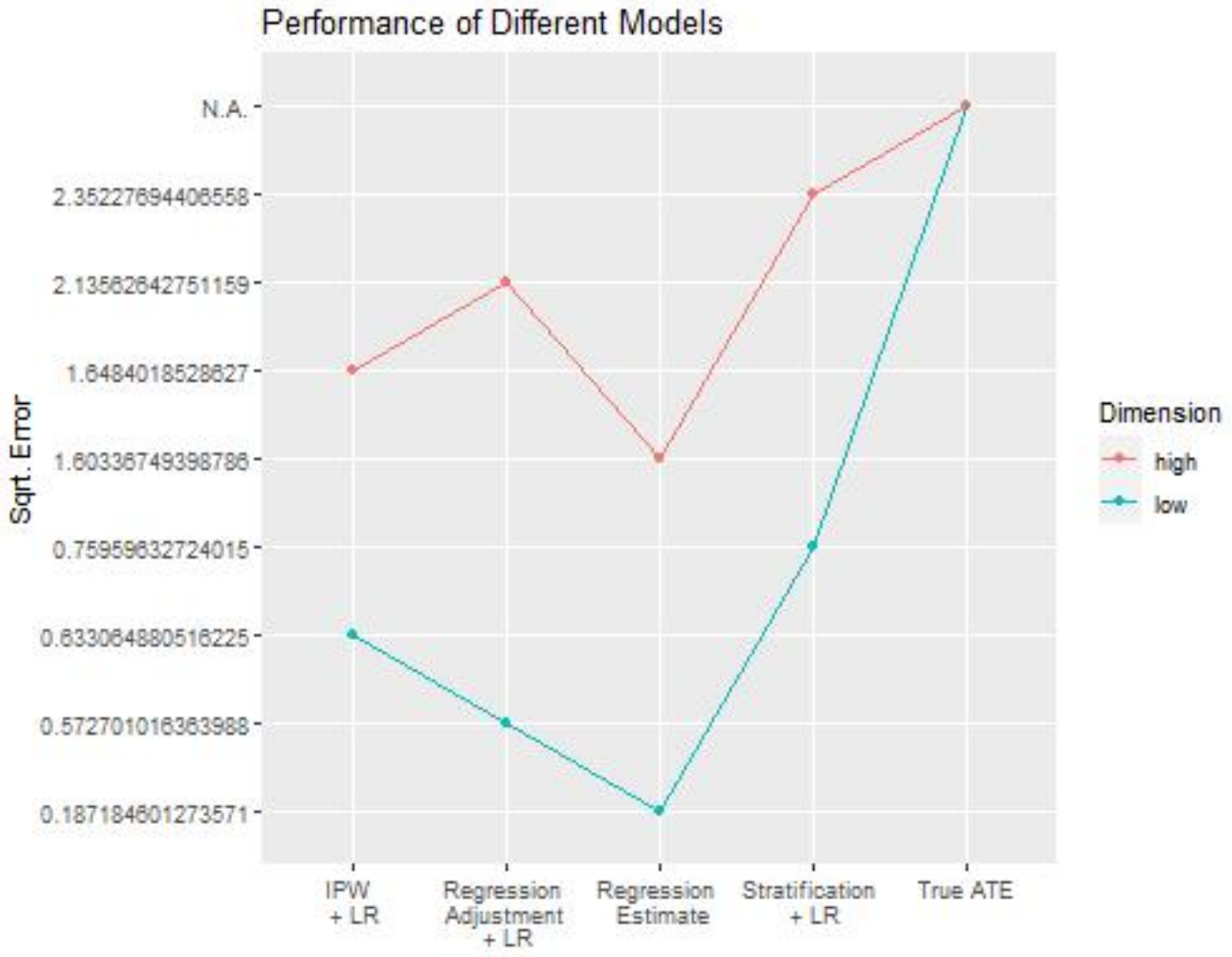
| Model | ATE | Run_Time | Performance |
|---|---|---|---|
| True ATE | -54.85580 | N.A | N.A. |
| IPW + LR | -57.57303 | 0 | 1.6484018528627 |
| Regression Estimate | -57.42659 | 0.206448078155518 | 1.60336749398786 |
| Stratification + LR | -60.38901 | 0.0339100360870361 | 2.35227694406558 |
| Regression Adjustment + LR | -59.41670 | 0 | 2.13562642751159 |

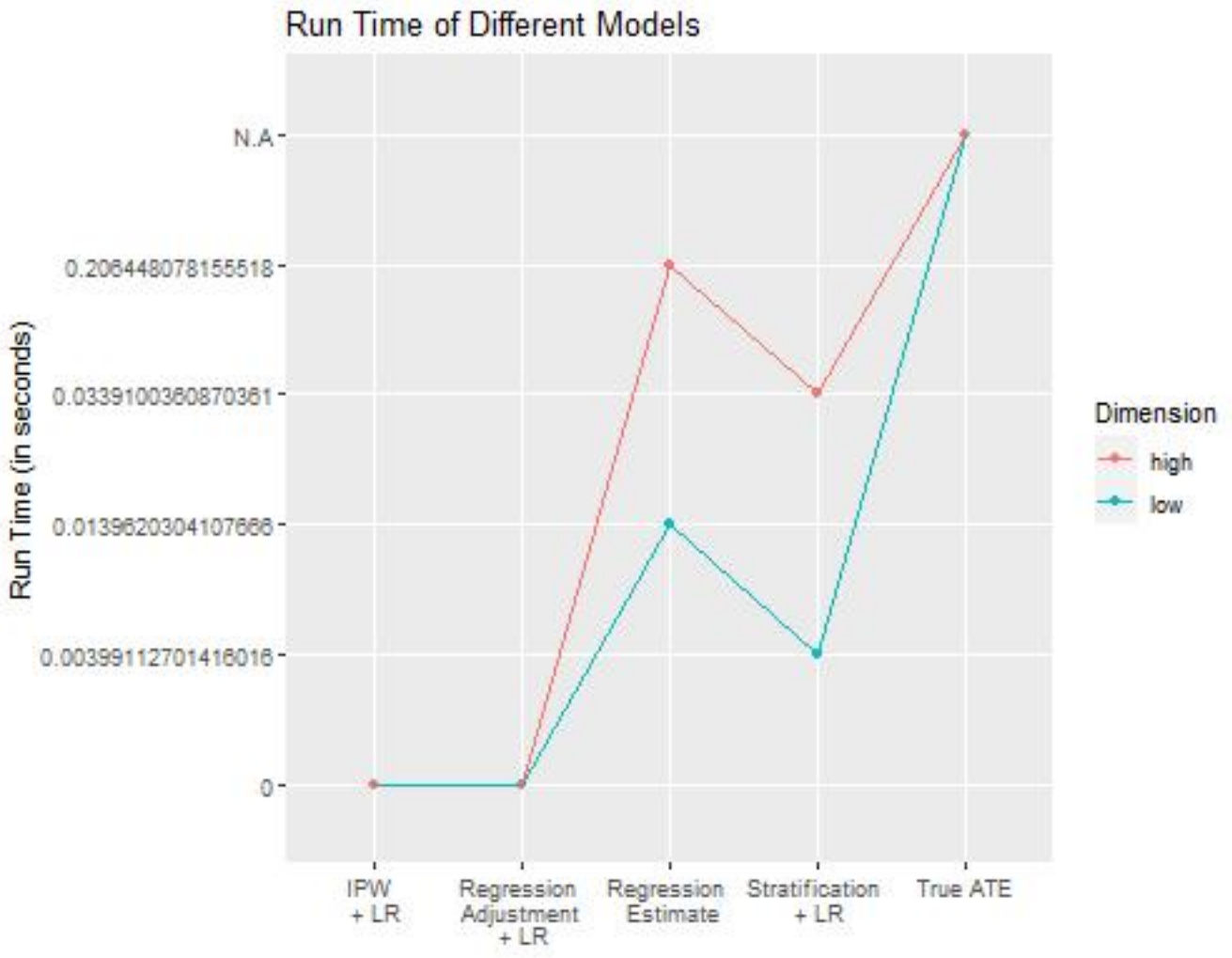
Regression Estimate has the best performance on Low Dimension Dataset; Regression Estimate has the best performance on High Dimension Dataset.

## Ploting the Result

```
FALSE pdf
FALSE   2

FALSE pdf
FALSE   2

FALSE pdf
FALSE   2
```




Performance of Different Models
Sqrt. Error
N.A.
2.35227694406558
2.13562642751159
1.6484018528627
1.60336749398786
0.75959632724015
0.633064880516225
0.572701016363988
0.187184601273571
IPW + LR
Regression Adjustment + LR
Regression Estimate
Stratification + LR
True ATE
Dimension
high
low




## Reference

Austin, Peter C. 2011. "An Introduction to Propensity Score Methods for Reducing the Effects of Confounding in Observational Studies." Multivariate Behavioral Research 46 (3): 399–424.

Lunceford, Jared K, and Marie Davidian. 2004. "Stratification and Weighting via the Propensity Score in Estimation of Causal Treatment Effects a Comparative Study." Statistics in Medicine 23 (19): 2937–60.

Stuart, Elizabeth A. 2010. "Matching Methods for Causal Inference: A Review and a Look Forward." Statistical Science: A Review Journal of the Institute of Mathematical Statistics 25 (1): 1.