Project 4 Group 6 Report

Kristen Akey, Citina Liang, Rui Liang, Mubai Liu, Wen Yin

December 2, 2020

Contents

Causal Inference Algorithms Evaluation

```
# Load libraries
pack <- c("readr", "tidyverse", "glmnet", "pryr", "flextable")

# if package not already installed, install, and load packages
if (!require("pacman")) install.packages("pacman")
pacman::p_load(pack)

for (package in pack) {
  pacman::p_load(package, character.only = TRUE, dependence=TRUE)
}
```

```
# Load data
lowDim <- read_csv('../data/lowDim_dataset.csv')
highDim <- read_csv('../data/highDim_dataset.csv')
```

Project Overview

In this project, we evaluate three causal inference algorithms. The three models include inverse propensity weighting (IPW) + L1 penalized logistic regression, regression estimate, and weighted regression + L1 penalized logistic regression. We compute the average treatment effect (ATE) using these algorithms on two distinct datasets and compare their performance and computational efficiency.

This report includes a description of each of the algorithms, code to reproduce our results, and a final comparison of each of these models.

Model 1: Inverse Propensity Weighting and L1 Penalized Logistic Regression

The propensity score is predicted through a lasso logistic regression model. Weights for the propensity score are then calculated using the inverse probability of treatment weighting (IPTW) formula. The average treatment effect is then calculated using inverse propensity weighting.

```
# Split into x, A and y
hY<-highDim$Y
hA<-highDim$A
hX<-highDim%>% select(-Y, -A) %>% as.matrix

lY<-lowDim$Y
lA<-lowDim$A
lX<-lowDim%>% select(-Y, -A) %>% as.matrix
```

```
# Setting alpha = 1 implements lasso regression
set.seed(0)
lasso_hd <- cv.glmnet(hX, hA,family = "binomial", alpha = 1)
lasso_ld <- cv.glmnet(lX, lA, family = "binomial",alpha = 1)

IPW<-function(x,A,model,data){
      start_time <- Sys.time()
      # Calculate the propensity score
      lasso_model <- glmnet(x, A, alpha = 1, family = "binomial",lambda = model$lambda.min)
      propensity <- predict(lasso_model, x, type = "response")

      # Calculate the weights
      weight <- 1 / propensity * A + 1 / (1 - propensity) * (1 - A)

      resampled_data <- data %>%
            mutate(propensity = propensity,
                   weight = weight,
                   Y_Weight = Y*weight)

      ATE<-1/nrow(resampled_data)*(sum(resampled_data[resampled_data$A==1,"Y_Weight"])
                                   -sum(resampled_data[resampled_data$A==0,"Y_Weight"]))
      end_time <- Sys.time()
      return(list(ATE=ATE,running_time = end_time - start_time))
}
ATE_highDim<-IPW(hX,hA,lasso_hd,highDim) #ATE: -2.21809, runtime:0.02583098
ATE_lowDim<-IPW(lX,lA,lasso_ld,lowDim) #ATE: 2.21036, runtime:0.005035877

matrix(c(ATE_highDim$ATE,ATE_lowDim$ATE,
         ATE_highDim$running_time,ATE_lowDim$running_time),
         nrow = 2,byrow = TRUE,
         dimnames = list(c("ATE","running_time (secs)"), c("highDim","lowDim")))
```

```
##                         highDim      lowDim
## ATE                  -2.218090 2.210363996
## running_time (secs)   0.050704 0.007995844
```

Model 2: Regression Estimate

The regression estimate does not use the propensity score. The model creates predictions for the control and treatment groups, and calculates the average treatment effect using these predications.

```
df_ld <- lowDim %>% mutate(A = factor(A))
df_hd <- highDim %>% mutate(A = factor(A))
```

```
RE <- function(df){
  # simple regression estimate
  # separate X and Y, will be used in predict function
  df_X <- df %>% select(-Y, -A)

  start <- Sys.time()
  # m0
  m0 <- glm(Y ~ ., data = subset(df[df$A==0,], select = -A))
  # m1
  m1 <- glm(Y ~ ., data = subset(df[df$A==1,], select = -A))
  # prediction using non-treatment model params
  Y_pred_0 <- predict(m0, newdata = df_X)
  # prediction using treatment model params
  Y_pred_1 <- predict(m1, newdata = df_X)
  # add predicted y to the dataframe
  df <- df %>% mutate(Y_pred1 = Y_pred_1, Y_pred0 = Y_pred_0)

  # calculate ATE
  n <- nrow(df)
  ATE = 1/n * sum(df$Y_pred1 - df$Y_pred0)
  end <- Sys.time()
  runtime = end - start
  return(list(ATE = ATE,
              runtime = runtime))
}

matrix(c(RE(df_hd)$ATE,RE(df_ld)$ATE,
         RE(df_hd)$runtime,RE(df_ld)$runtime),
         nrow = 2,byrow = TRUE,
         dimnames = list(c("ATE","running_time (secs)"), c("highDim","lowDim")))
```

```
##                       highDim      lowDim
## ATE                -2.959780 2.526943984
## running_time (secs)  0.192153 0.005686045
```

Model 3: Weighted Regression and L1 Penalized Logistic Regression

The propensity score is predicted through a lasso logistic regression model. Weights for the propensity score are then calculated using the inverse probability of treatment weighting (IPTW) formula. The average treatment effect is then calculated using a weighted regression model.

```
set.seed(0)
start_time <- Sys.time()
X_low <- df_ld %>% select(-Y, -A) %>% as.matrix
A_low <- df_ld %>% select(A) %>% as.matrix
cv_l1 <- cv.glmnet(X_low, A_low, family = "binomial", alpha = 1)
l1_low <- glmnet(X_low, A_low, family = "binomial",
                 alpha = 1, lambda = cv_l1$lambda.min)
propen_score_low <- predict(l1_low, X_low, type = "response")

# Finding weights
weight_low <- cbind(as.numeric(A_low), propen_score_low) %>%
  as_tibble %>%
  mutate(weights = (V1/s0 + (1-V1)/(1-s0))) %>%
  select(weights)

# Linear regression for selecting covarites
filter_low <- summary(lm(Y~., data = df_ld))$coef[,4][3:24]<0.05
Z_low <- cbind(A_low, X_low[,filter_low])

Z_low <- Z_low %>% apply(2, as.numeric)

# Final Regression for ATE
Y_low <- df_ld$Y

weighted_low <- lm(Y_low ~ Z_low, weights = as.numeric(unlist(weight_low)))
ATE_low <- coef(weighted_low)[2]

end_time <- Sys.time()
running_time_low = end_time - start_time
```

```
set.seed(0)
start_time <- Sys.time()
X_high <- df_hd %>% select(-Y, -A) %>% as.matrix
A_high <- df_hd %>% select(A) %>% as.matrix
cv_l1_high <- cv.glmnet(X_high, A_high, family = "binomial", alpha = 1)
l1_high <- glmnet(X_high, A_high, family = "binomial",
                  alpha = 1, lambda = cv_l1_high$lambda.min)
propen_score_high <- predict(l1_high, X_high, type = "response")

weight_high <- cbind(as.numeric(A_high), propen_score_high) %>%
  as_tibble %>%
  mutate(weights = (V1/s0) + (1-V1)/(1-s0)) %>%
  select(weights)

filter_high <- summary(lm(Y~., data = df_hd))$coef[,4][3:ncol(X_high)]<0.05
Z_high <- cbind(A_high, X_high[,filter_high])

Z_high <- Z_high %>% apply(2, as.numeric)
```

```r
Y_high <- df_hd$Y

weighted_high <- lm(Y_high ~ Z_high, weights = as.numeric(unlist(weight_high)))
ATE_high <- coef(weighted_high)[2]
end_time <- Sys.time()
running_time_high = end_time - start_time
```

```r
matrix(c(ATE_high,ATE_low,
         running_time_high,running_time_low),
         nrow = 2,byrow = TRUE,
         dimnames = list(c("ATE","running_time (secs)"), c("highDim","lowDim")))
```

```
##                         highDim    lowDim
## ATE                   -2.980899 2.5193124
## running_time (secs) 22.710594 0.2006481
```

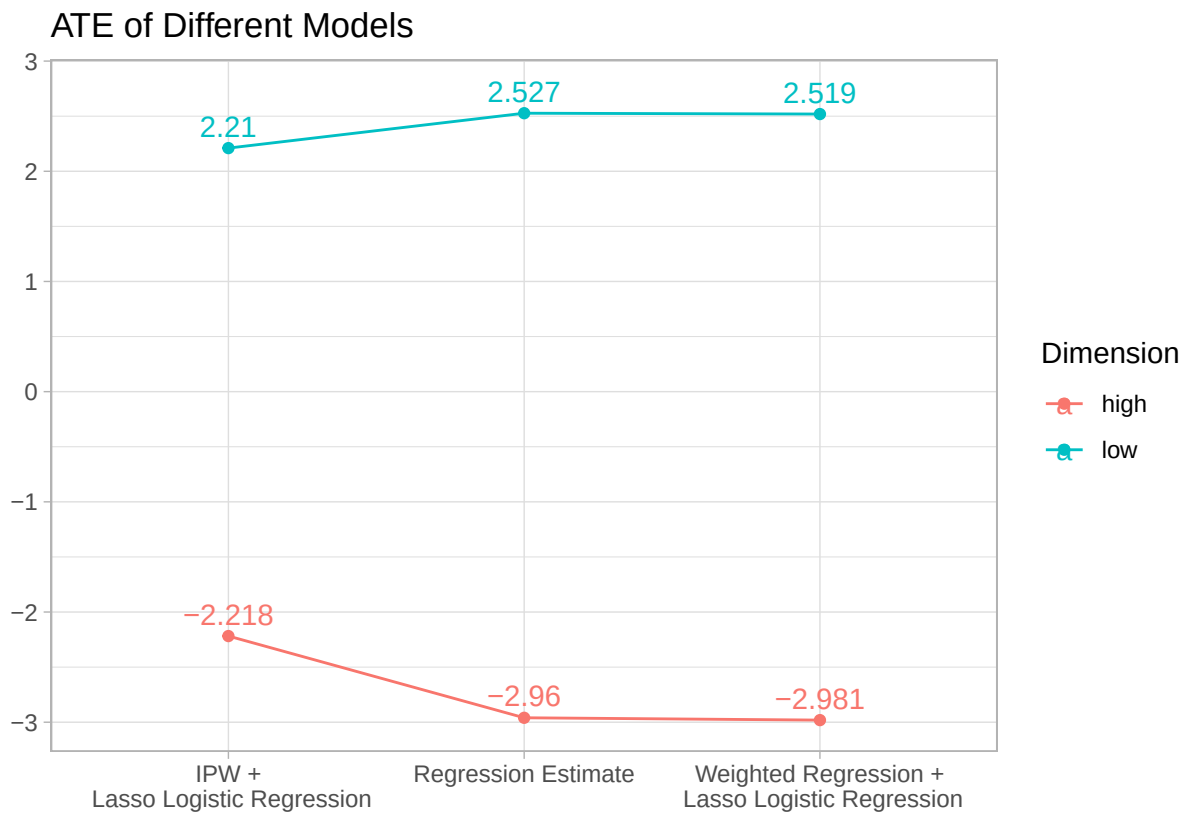
Model Comparisons

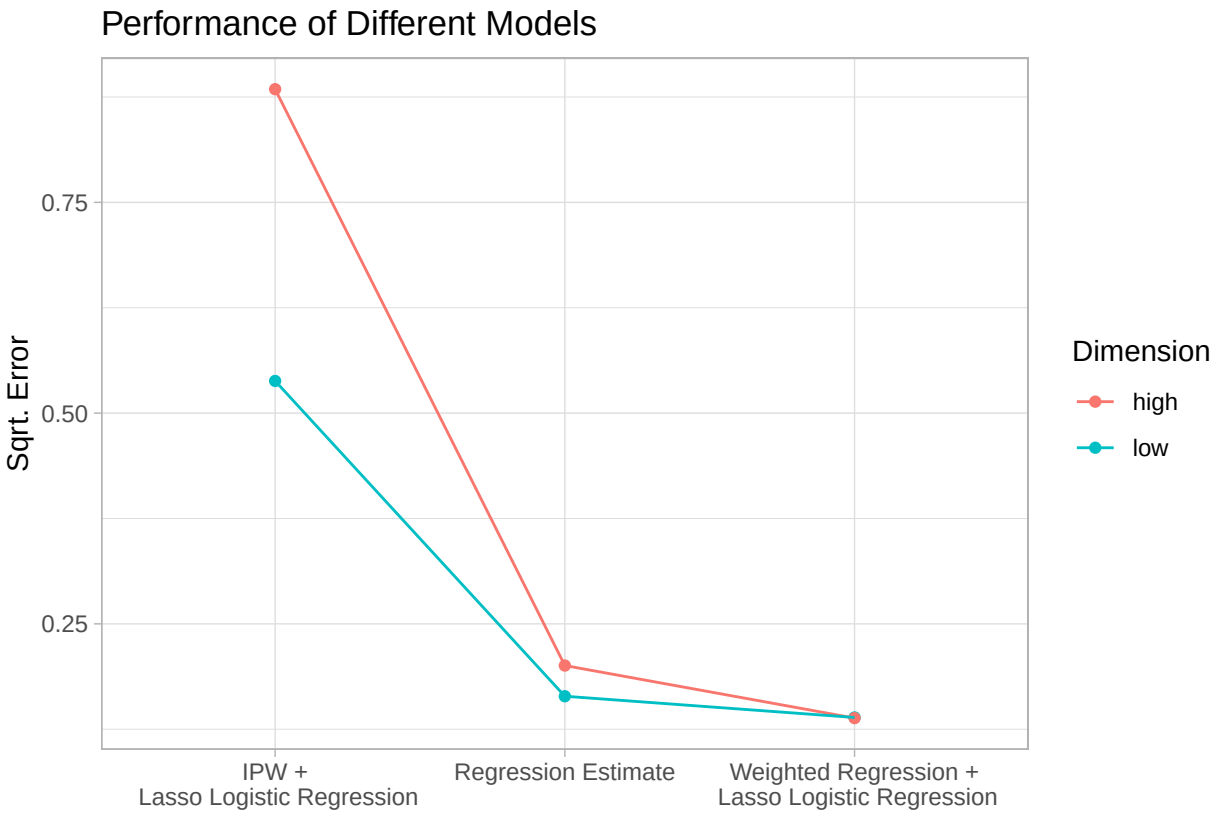
Performance = squared difference of true ATE and estimated ATE

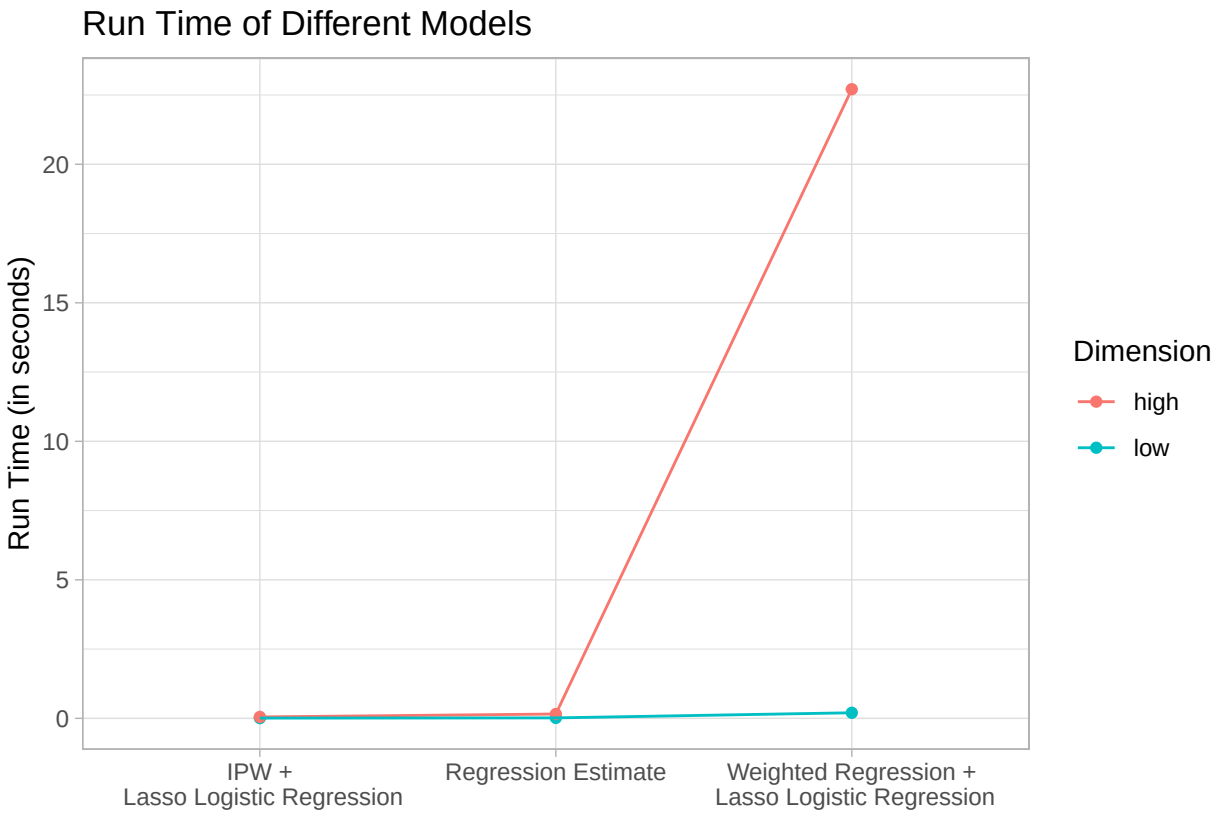
Run time (in seconds)

Model	ATE	Run.Time	Performance
IPW + Lasso Logistic Regression	2.210364	0.0079958	0.5381784
Regression Estimate	2.526944	0.0156350	0.1641462
Weighted Regression + Lasso Logistic Regression	2.519312	0.2006481	0.1389692

Model	ATE	Run.Time	Performance
IPW + Lasso Logistic Regression	-2.218090	0.0507040	0.8842567
Regression Estimate	-2.959780	0.1553719	0.2005502
Weighted Regression + Lasso Logistic Regression	-2.980899	22.7105939	0.1382057


ATE of Different Models
3
2
1
0
−1
−2
−3
2.21
2.527
2.519
−2.218
−2.96
−2.981
IPW +
Lasso Logistic Regression
Regression Estimate
Weighted Regression +
Lasso Logistic Regression
Dimension
high
low


Performance of Different Models
Sqrt. Error
0.75
0.50
0.25
IPW +
Lasso Logistic Regression
Regression Estimate
Weighted Regression +
Lasso Logistic Regression
Dimension
high
low


Run Time of Different Models
Run Time (in seconds)
0
5
10
15
20
IPW +
Lasso Logistic Regression
Regression Estimate
Weighted Regression +
Lasso Logistic Regression
Dimension
high
low