Project 4 Group 6 Report

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

December 2, 2020

Contents

Causal Inference Algorithms Evaluation

```r
# 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)
}
```

```r
# 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 effect using these algorithms on two distinct datasets.

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

```r
# 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
```

```r
# 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                 -2.21809011 2.210363996
## running_time (secs)  0.04725003 0.004922867
```

Model 2: Regression Estimate

```
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                 -2.9597796 2.526944
## running_time (secs)  0.1685359 0.010607
```

Model 3: Weighted Regression and L1 Penalized Logistic Regression

```
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)

Y_high <- df_hd$Y
```

```r
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
```

Model Comparisons

True ATE	
Low.Dim	**High.Dim**
2.5	-3

Performance = square difference of true ATE and estimated ATE

Run time (in seconds)

Low Dimension Dataset			
Model	**ATE**	**Run.Time**	**Performance**
IPW + Lasso Logistic Regression	2.2	0.0049	0.54
Regression Estimate	2.5	0.0127	0.16
Weighted Regression + Lasso Logistic Regression	2.5	0.1741	0.14

High Dimension Dataset			
Model	**ATE**	**Run.Time**	**Performance**
IPW + Lasso Logistic Regression	-2.2	0.047	0.88
Regression Estimate	-3.0	0.172	0.20
Weighted Regression + Lasso Logistic Regression	-3.0	19.766	0.14