Part 4: Comparing causal model evaluations on the assistance question

Austin Murphy, Fred Lu

6/3/2020


```
# Downloading script
# df <- readr::read_csv(file = "https://raw.githubusercontent.com/gsbDBI/ExperimentData/master/Welfare/ProcessedDat

# 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"))
df <-
  df %>%
  dplyr::left_join(black_assistance)

# Specify outcome, treatment, and covariate variable names to use
outcome_variable_name <- "y_race" # or y_race for the other outcome variable
treatment_variable_name <- "w"
covariate_names <- c("partyid", "polviews", "age", "educ", "year", "attblack")

# Combine all names
all_variables_names <- c(outcome_variable_name, treatment_variable_name, covariate_names)
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...

dim(df)
```

```
## [1] 36501     8
```

```
names(df)
```

```
## [1] "year"     "age"      "educ"     "partyid"  "polviews" "W"        "attblack"
## [8] "Y"
```

```
head(df)
```

```
## # A tibble: 6 x 8
##    year   age  educ partyid polviews     W attblack     Y
##   <int> <int> <int>   <int>    <dbl> <int>    <dbl> <int>
## 1  1986    28    14       3        4     0    0.333     0
## 2  1986    54    16       6        6     0    0.5       1
## 3  1986    44    16       0        2     0    0.75      0
## 4  1986    77    14       0        4     0    0.5       0
## 5  1986    44    14       0        4     0    0.5       0
## 6  1986    47    10       1        1     0    0.5       0
```

```
# Filter out missing data
df <-
  df %>%
  filter_all(all_vars(. != -999)) %>%
  filter(partyid != 7)

# 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),]
head(df)
```

```
## # A tibble: 6 x 8
##    year   age  educ partyid polviews     W attblack     Y
##   <int> <int> <int>   <int>    <dbl> <int>    <dbl> <int>
## 1  1986    28    14       3        4     0    0.333     0
## 2  1986    54    16       6        6     0    0.5       1
## 3  1986    44    16       0        2     0    0.75      0
## 4  1986    77    14       0        4     0    0.5       0
## 5  1986    44    14       0        4     0    0.5       0
## 6  1986    47    10       1        1     0    0.5       0
```

Recall that the ATE is 0.06 on this dataset.

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

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

Models for estimating CATE

X-learner

Before fitting the X-learner, first generate, as simple comparisons, model fits with the S-learner and T-learner strategies. We would anticipate that the X-learner would give similar results to BART, both of which would be more effective than S or T learner.

```
testidx <- base::sample(nrow(df), 5000)
df_train <- df[-testidx, ]
df_test <- df[testidx, ]

X <- df_train[,covariate_names]
W <- df_train$W
Y <- df_train$Y
num.trees <- 1000

# s learner
slearn <- regression_forest(df_train[,c(covariate_names,'W')], Y, num.trees=num.trees)

# t learner
n_train <- dim(df_train)[1]
tf0 <- regression_forest(X[W==0,], Y[W==0], num.trees=num.trees)
tf1 <- regression_forest(X[W==1,], Y[W==1], num.trees=num.trees)
```

Now estimate HTEs via the X-learner strategy, using regression forests for the nuisance components.

```
# Compute the 'imputed treatment effects' using the other group
D1 <- Y[W==1] - predict(tf0, X[W==1,])$predictions
D0 <- predict(tf1, X[W==0,])$predictions - Y[W==0]

# Compute the cross estimators
xf0 <- regression_forest(X[W==0,], D0, num.trees=num.trees)
xf1 <- regression_forest(X[W==1,], D1, num.trees=num.trees)

# Predict treatment effects, making sure to always use OOB predictions where appropriate
xf.preds.0 <- rep(0, n_train)
xf.preds.0[W==0] <- predict(xf0)$predictions
xf.preds.0[W==1] <- predict(xf0, X[W==1,])$predictions
xf.preds.1 <- rep(0, n_train)
xf.preds.1[W==0] <- predict(xf0)$predictions
xf.preds.1[W==1] <- predict(xf0, X[W==1,])$predictions

# Estimate the propensity score
propf <- regression_forest(X, W, num.trees=num.trees)
ehat <- predict(propf)$predictions

# Finally, compute the X-learner prediction
tauhat_xl <- (1 - ehat) * xf.preds.1 + ehat * xf.preds.0
```

Causal Forest

Train a causal forest and compute (out-of-bag) CATE estimates on the training set.

```
cf <- causal_forest(
  X = df_train[, covariate_names],
  Y = df_train$Y,
  W = df_train$W,
  num.trees=1000
)

oob_pred <- predict(cf, estimate.variance=TRUE)
```

BART model

Fit the BART model again, to enable test set comparisons with the other models

```
test_w0 = df_test %>% select(-Y)
test_w1 = df_test %>% select(-Y)
test_w0$W = 0
test_w1$W = 1
x.test = as.data.frame(rbind(test_w0, test_w1))

bart = pbart(x.train = as.data.frame(df_train %>% select(-Y)),
             y.train = df_train$Y,
             x.test = x.test,
             nskip = 1000)
```

```
## *****Into main of pbart
## *****Data:
## data:n,p,np: 14738, 7, 10000
## y1,yn: 0, 0
## x1,x[n*p]: 1986.000000, 0.500000
## xp1,xp[np*p]: 2002.000000, 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: -0.881959
## *****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: 94s
## check counts
## trcnt,tecnt: 1000,1000
```

```
# Process BART predictions to get in same format (treatment effect for each entry)
pred_w0 = bart$prob.test.mean[1:5000]
pred_w1 = bart$prob.test.mean[5001:10000]
tauhat_bart_test <- pred_w1 - pred_w0
```

```
X_test <- df_test[,covariate_names]
tauhat_s_test <- predict(slearn, df_test[,c(covariate_names,'W')])

tauhat_t_test <- numeric(nrow(df_test))
tauhat_t_test[which(df_test$W==1)] <- predict(tf1, X_test[df_test$W==1, ])$predictions
tauhat_t_test[which(df_test$W==0)] <- predict(tf1, X_test[df_test$W==0, ])$predictions

X_test <- df_test[,covariate_names]
ehat.test <- predict(propf, X_test)$predictions
xf.preds.1.test <- predict(xf1, X_test)$predictions
xf.preds.0.test <- predict(xf0, X_test)$predictions
tauhat_xl_test <- (1 - ehat.test) * xf.preds.1.test + ehat.test * xf.preds.0.test

test_pred <- predict(cf, as.matrix(df_test[,covariate_names]), estimate.variance=TRUE)
tauhat_cf_test <- test_pred$predictions


# Compute Y-star
p <- mean(df_test$W)
Y_star <- ((df_test$W - p)/(p*(1-p)))*df_test$Y

# Compute the sample average treatment effect to use as a baseline comparison
tauhat_sample_ate <- with(df_train, mean(Y[W==1]) - mean(Y[W==0]))

# Compute test mse for all methods
mse <- data.frame(
  Sample_ATE_Loss = (Y_star - tauhat_sample_ate)^2,
  BART_Loss = (Y_star - tauhat_bart_test)^2,
  Causal_Forest_Loss = (Y_star - tauhat_cf_test)^2,
  X_Learner_Loss = (Y_star - tauhat_xl_test)^2,
  S_Learner_Loss = (Y_star - tauhat_s_test$predictions)^2,
  T_Learner_Loss = (Y_star - tauhat_t_test)^2)

print('Mean')
```

```
## [1] "Mean"
```

```
colMeans(mse)
```

```
##    Sample_ATE_Loss          BART_Loss Causal_Forest_Loss     X_Learner_Loss
##          0.7482886          0.7477187          0.7485652          0.7486786
##     S_Learner_Loss     T_Learner_Loss
##          0.7506756          0.7726409
```

```
print('SD')
```

```
## [1] "SD"
```

```
apply(mse, 2, sd)
```

```
##    Sample_ATE_Loss          BART_Loss Causal_Forest_Loss     X_Learner_Loss
##           1.551426           1.552584           1.552684           1.553003
##     S_Learner_Loss     T_Learner_Loss
##           1.564302           1.602983
```

Recall that in the welfare question analysis, BART, Causal Forest, and X-learner perform similarly, while achieving significantly better metrics than S-learner and T-learner. However, on this assistance question, all methods perform similarly. This may be attributed to the lack of a strong treatment effect in the dataset (compare 0.06 vs 0.33 previously). If there is not a strong or consistent pattern of heterogeneous treatment effect, any improved methods will not outperform the baseline treatment effect estimate. Supporting that assumption, we see here that the sample ATE loss baseline performs just as well as any other method.

```
Y.forest.test = regression_forest(X = as.matrix(df_test[covariate_names]), Y = df_test$Y)
Y.hat.test = predict(Y.forest.test)$predictions
W.forest.test = regression_forest(X = as.matrix(df_test[covariate_names]), Y = df_test$W)
W.hat.test = predict(W.forest.test)$predictions

mse_rloss <- data.frame(
  Sample_ATE_Loss = (df_test$Y - Y.hat.test - (df_test$W - W.hat.test) * tauhat_sample_ate)^2,
  BART_Loss = (df_test$Y - Y.hat.test - (df_test$W - W.hat.test) * tauhat_bart_test)^2,
  Causal_Forest_Loss = (df_test$Y - Y.hat.test - (df_test$W - W.hat.test) * tauhat_cf_test)^2,
  X_Learner_Loss = (df_test$Y - Y.hat.test - (df_test$W - W.hat.test) * tauhat_xl_test)^2,
  S_Learner_Loss = (df_test$Y - Y.hat.test - (df_test$W - W.hat.test) * tauhat_s_test$predictions)^2,
  T_Learner_Loss = (df_test$Y - Y.hat.test - (df_test$W - W.hat.test) * tauhat_xl_test)^2
)

print('Mean')
```

```
## [1] "Mean"
```

```
colMeans(mse_rloss)
```

```
##    Sample_ATE_Loss          BART_Loss Causal_Forest_Loss     X_Learner_Loss
##          0.1452209          0.1450260          0.1451812          0.1452226
##     S_Learner_Loss     T_Learner_Loss
##          0.1489455          0.1452226
```

```
print('SD')
```

```
## [1] "SD"
```

```
apply(mse_rloss, 2, sd)
```

```
##    Sample_ATE_Loss          BART_Loss Causal_Forest_Loss     X_Learner_Loss
##          0.2272428          0.2276221          0.2273643          0.2274689
##     S_Learner_Loss     T_Learner_Loss
##          0.2314391          0.2274689
```

The results here demonstrate the same effects as previously.