How to correctly estimate the effect of advertisement

Introduction of Double Machine Lerning

About me

- Yusuke Kaneko(@coldstart_p)
- CyberAgent, Inc.
- Studying: Econometrics, Causal Inference

Main Theme

How to estimate correctly treatment effects in high-dimensional data(such as advertising data)

Original

1. Chernozhukov, Victor, et al. "Double/debiased/neyman machine learning of treatment effects." *American Economic Review* 107.5 (2017): 261-65.

2. Belloni, A., Chernozhukov, V., Fernández-Val, I., & Hansen, C. (2017). Program evaluation and causal inference with high-dimensional data. *Econometrica*, *85*(1), 233-298.

3. Chernozhukov, Victor, et al. "Double/debiased machine learning for treatment and structural parameters." *The Econometrics Journal* 21.1 (2018): C1-C68.

Problem

- Advertisement data has a lot of variables. (= High-Dimensional Data)
  example: Type of Advertisement , Time , History-data
  ➔ Under such circumstances, it is known that low-dimensional parameters such as policy effects can not be estimated well by Machine Learning Methods (The model and figure are on the slide later)

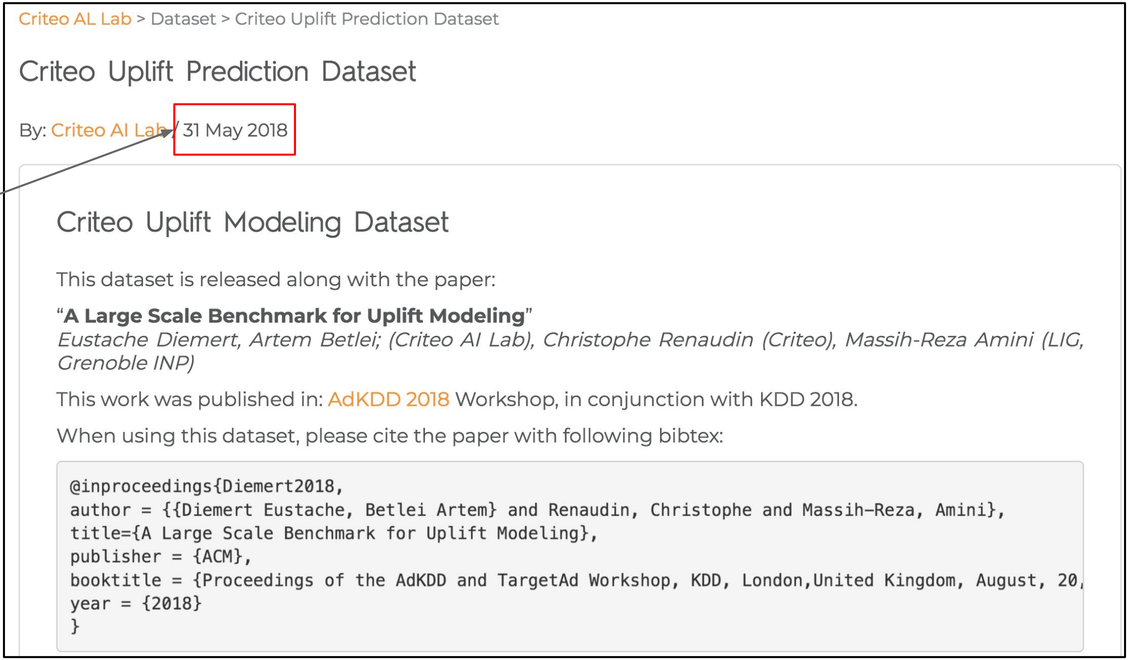
- Motivation: How to estimate policy effects correctly ?
  ➔ Use Double Machine Learning(Double ML) !
  =(FLEXIBLE and SIMPLE Method)

- Double ML Code are on github(https://github.com/VC2015/DMLonGitHub/)
  …(Double ML package does not exist yet)

Data

- [Criteo Uplift Modeling Dataset](EC Site AD Large Dataset for causal inference)

Recently Published Dataset!



Criteo AL Lab > Dataset > Criteo Uplift Prediction Dataset

Criteo Uplift Prediction Dataset

By: Criteo AI Lab / 31 May 2018

Criteo Uplift Modeling Dataset

This dataset is released along with the paper:

“**A Large Scale Benchmark for Uplift Modeling**”
Eustache Diemert, Artem Betlei; (Criteo AI Lab), Christophe Renaudin (Criteo), Massih-Reza Amini (LIG, Grenoble INP)

This work was published in: AdKDD 2018 Workshop, in conjunction with KDD 2018.

When using this dataset, please cite the paper with following bibtex:

```
@inproceedings{Diemert2018,
author = {{Diemert Eustache, Betlei Artem} and Renaudin, Christophe and Massih-Reza, Amini},
title={A Large Scale Benchmark for Uplift Modeling},
publisher = {ACM},
booktitle = {Proceedings of the AdKDD and TargetAd Workshop, KDD, London,United Kingdom, August, 20,
year = {2018}
}
```

Data

- Features are as follows

Fields

Here is a detailed description of the fields (they are comma-separated in the file):

- **f0, f1, f2, f3, f4, f5, f6, f7, f8, f9, f10, f11**: feature values (dense, float)
- **treatment**: treatment group (1 = treated, 0 = control)
- **conversion**: whether a conversion occured for this user (binary, label)
- **visit**: whether a visit occured for this user (binary, label)
- **exposure**: treatment effect, whether the user has been effectively exposed (binary)

- Averege Visit Rate and Treatment Ratio is on the right

Key figures

- Format: CSV
- Size: 459MB (compressed)
- Rows: 25,309,483
- Average Visit Rate: .04132
- Average Conversion Rate: .00229
- Treatment Ratio: .846

Model

- Partially Linear Model

$$Y = D\theta_0 + g_0(Z) + U, \quad E[U|Z, D] = 0$$

Y: Outcome Variable(Whether user visited = visit)

D: Policy Variable (Whether or not an ad has been delivered = treatment)

Z: Vector of covariates(such as history data = f0 f1 ...)

θ_0: “lift” parameter(Policy effect parameter)

- D is expressed by the following equation

$$D = m_0(Z) + V, \quad E[V|Z] = 0$$

D is conditionally exogenous

Bad ML Estimation

- Predict Y using D and Z, then obtain

$$D\hat{\theta}_0 + \hat{g}_0(Z)$$

(Example)

1. Run Random Forest $Y - D\hat{\theta}_0$ on Z and get

$$\hat{g}_0(Z)$$

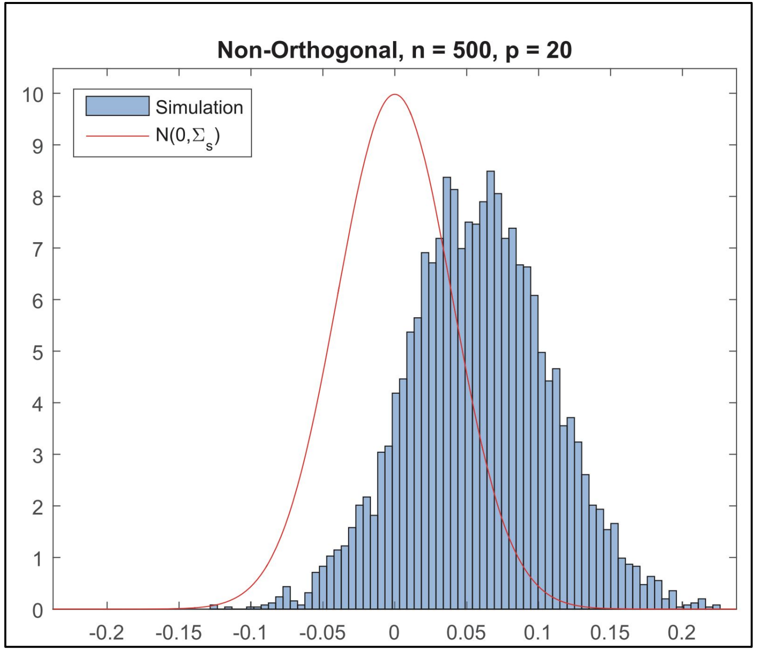
2. $\text{OLS on Y} - \hat{\text{g}}_0(\text{Z})\ \text{on Z}$ and get

$$\hat{\theta}_0$$

3. Repeat 1. and 2. until convergence

Bad ML Estimation

- Prediction performance is Good, but the distribution of $\hat{\theta}_0 - \theta_0$ looks like below.



(from paper)

Double ML

1. Predict Y and D using Z by

$$\widehat{E[Y|Z]} \text{ and } \widehat{\mathrm{E[D|Z]}}$$

obtained using ML methods(Lasso , Random Forest etc...)

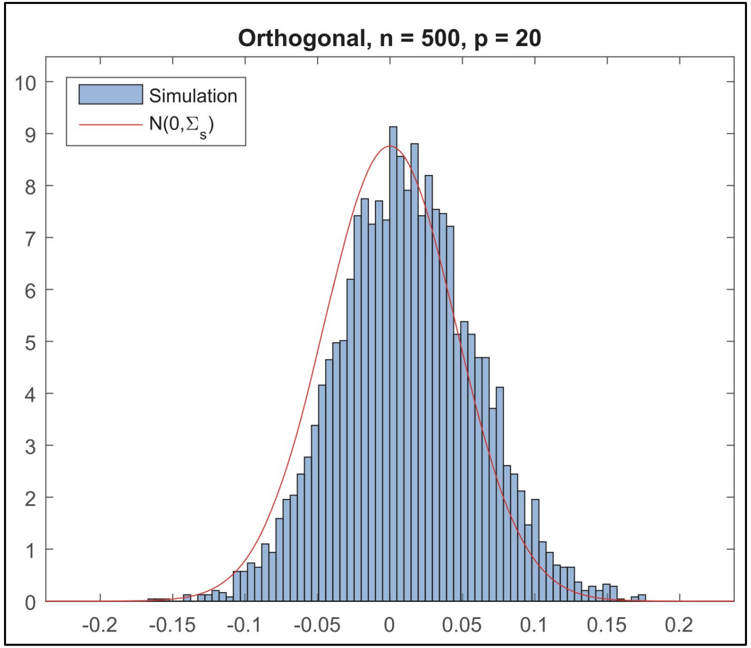
2. Residualize

$$\widehat{W} = Y - \widehat{E[Y|Z]} \text{and } \widehat{\mathrm{V}} = \mathrm{D} - \widehat{\mathrm{E[D|Z]}}$$

3. Regress $\widehat{W}$ on $\widehat{\mathrm{V}}$ and get $\tilde{\theta}_0$

Double ML

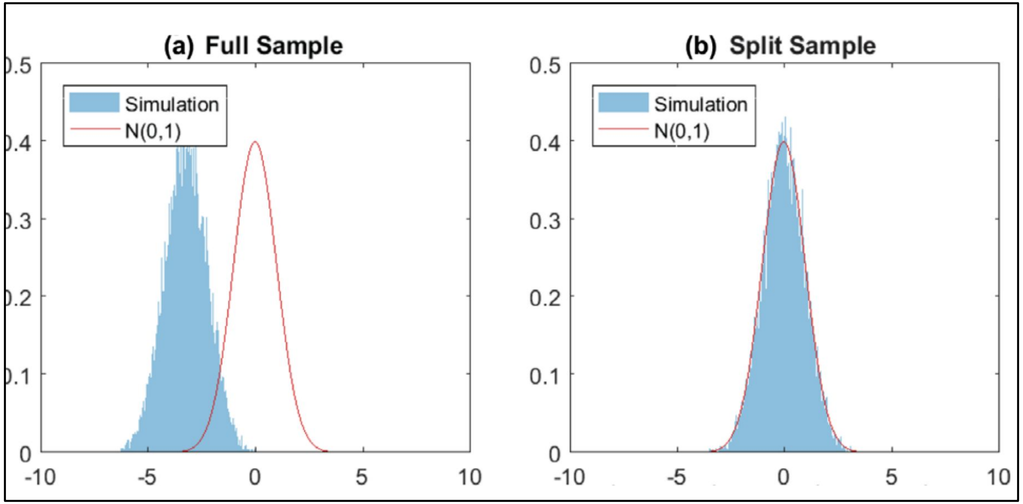
- Prediction performance is Good, and the distribution of $\tilde{\theta}_0 - \theta_0$ looks like below.



(from paper)

Sample Splitting

- In Double ML, Sample Splitting is one key Ingredient.
  ➜ Convergence rate improves with sample splitting(Please refer to the papers for the theoretical content)



(from paper)

Double ML in R

- All the code for the simulation is [on the github]

```
DML2.for.PLM <- function(x, d, y, dreg, yreg, nfold=5) {
  # this implements DML2 algorithm, where there moments are estimated via DML, before constructing
  # the pooled estimate of theta randomly split data into folds
  nobs <- nrow(x)
  foldid <- rep.int(1:nfold,times = ceiling(nobs/nfold))[sample.int(nobs)]
  I <- split(1:nobs, foldid)
  # create residualized objects to fill
  ytil <- dtil <- rep(NA, nobs)
  # obtain cross-fitted residuals
  cat("fold: ")
  for(b in 1:length(I)){
    dfit <- dreg(x[-I[[b]],], d[-I[[b]]])  #take a fold out
    yfit <- yreg(x[-I[[b]],], y[-I[[b]]])  # take a folot out
    dhat <- predict(dfit, x[I[[b]],], type="response")  #predict the fold out
    yhat <- predict(yfit, x[I[[b]],], type="response")  #predict the fold out
    dtil[I[[b]]] <- (d[I[[b]]] - dhat) #record residual
    ytil[I[[b]]] <- (y[I[[b]]] - yhat) #record residial
    cat(b," ")
  }
  rfit <- lm(ytil ~ dtil)               #estimate the main parameter by regressing one residual on the other
  coef.est <- coef(rfit)[2]             #extract coefficient
  se <- sqrt(vcovHC(rfit)[2,2])         #record standard error
  cat(sprintf("\ncoef (se) = %g (%g)\n", coef.est , se))
  low_ <- coef.est - se*1.96
  upp_ <- coef.est + se*1.96
  return(data.frame(Method = "Double ML", ATE = coef.est, lower_ci = low_, upper_ci = upp_))
}
```

Simulation

- Since Criteo Dataset is too large , I made overall sampling first (about 2%)

- Introduce Sampling Bias(Since we are using data coming from a randomized experiment)

```r
#Introduce Sampling bias
pt <- .80 # Drop p% of users who satisfy the following condition
pc <- .95

drop_from_treat <-  (df[,"f6"] < 2 | df[,"f0"] > 0.5)

drop_from_control <-(df[,"f6"] > 2 | df[,"f0"] < 0.5)

drop_treat_idx <- which(df[,"W"] == 1 & drop_from_treat)
drop_control_idx <- which(df[,"W"] == 0 & drop_from_control)

drop_idx <- unique(c(drop_treat_idx[1:round(pt*length(drop_treat_idx))],
                     drop_control_idx[1:round(pc*length(drop_control_idx))]))

print(length(drop_idx))
df_mod <- df[-drop_idx,]
df_mod <- df_mod[sample(nrow(df_mod)),]
rownames(df_mod) <- NULL
```

Naive ATE Method

- Unbiased estimation is possible with this technique even if there is no sampling bias

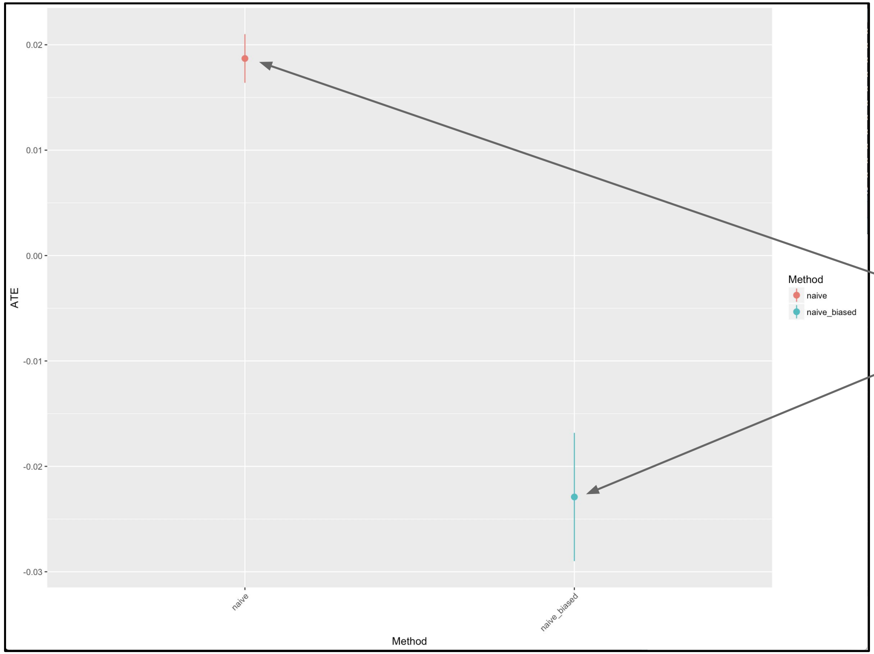
```
mean_df <- dataset %>%
  group_by_(W_var) %>%
  summarise_(y = paste("mean(", outcome_var, ")"),
             y_var = paste("var(", outcome_var, ")"),
             count = "n()") %>%
  mutate(y_var_weight = y_var/(count - 1))

E_y0 = mean_df$y[mean_df$W == 0]
E_y1 = mean_df$y[mean_df$W == 1]

tau_hat <- E_y1 - E_y0
```

Simulation

- Average treatment effect without sampling bias on the left
- Average treatment effect with sampling bias on the right



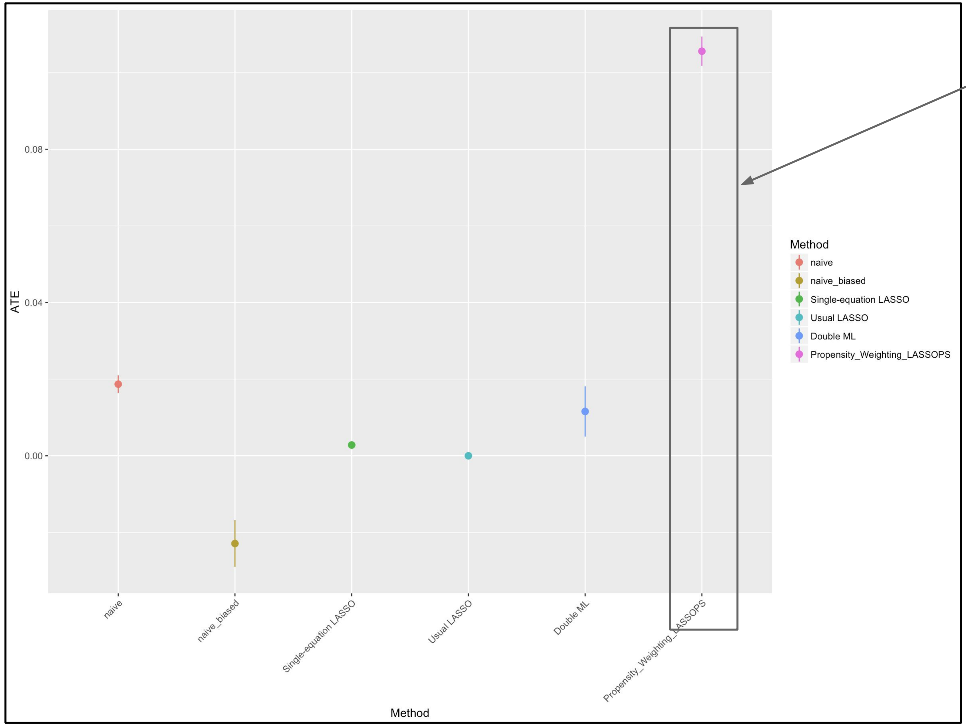
Highly Biased

Simulation

- Use approach: Double ML with Lasso(You can use RandomForest instead of Lasso)
  ➡ Counter approach: Naive ATE, Single equation Lasso , Usual Lasso , Lasso with Propensity Score Weighting

Simulation

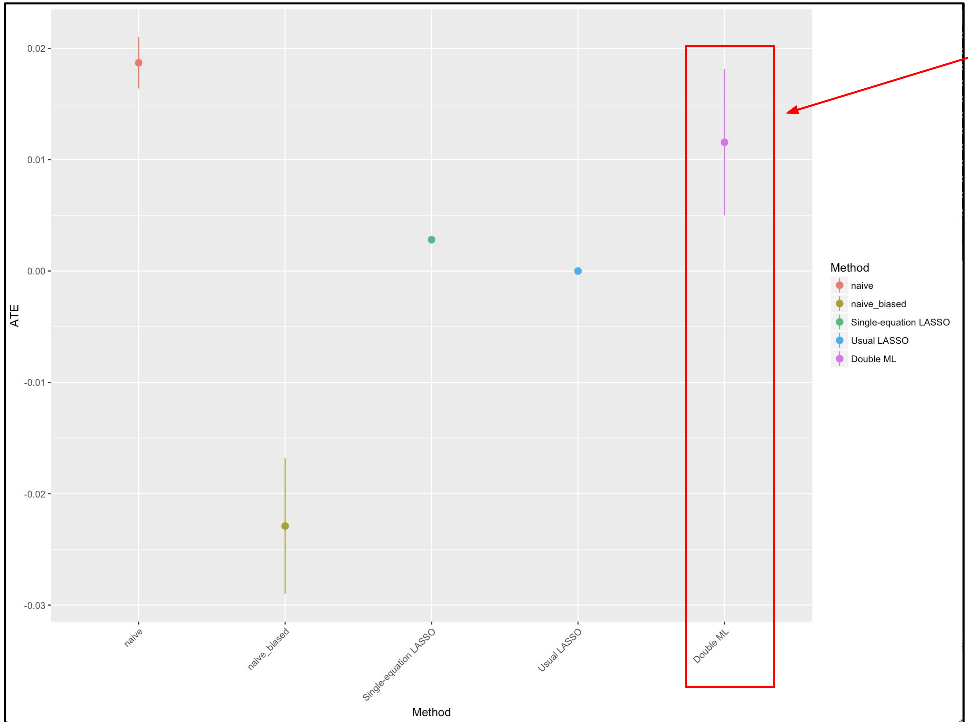
- Result(with Lasso with Propensity Score Weighting)



Lasso with PSW's performance is very bad , so remove this

Simulation

- Result(without Lasso with Propensity Score Weighting)



Double ML’s performance is very good!

Conclusion

- Double ML is Flexible and Simple Method for causal inference
- Double ML’s performance seems very good with real data