DID and DML Report

19331053 纪传宇

2023 年 7 月 10 日

1 DRDID step

we argue that instead of choosing between the OR and the IPW approaches, one can combine them to form doubly robust (DR) moments/estimands for the ATT. Here, double robustness means that the resulting estimand identifies the ATT even if either (but not both) the propensity score model or the outcome regression models are misspecified. As so, the DR DID estimand for the ATT shares the strengths of each individual DID method and, at the same time, avoids some of their weaknesses.

Before describing how we exactly combine the OR and the IPW approaches to form our DR DID estimand, we need to introduce some additional notation. Let $\pi(X)$ be an arbitrary model for the true, unknown propensity score. When panel data are available, let $\Delta Y = Y_1 - Y_0$ and define $\mu^p_{d,\Delta}(X) \equiv \mu^p_{d,1}(X) - \mu^p_{d,0}(X), \mu^p_{d,t}(x)$ being a model for the true, unknown outcome regression $m^p_{d,t}(x) \equiv \mathbb{E}\left[Y_t \mid D = d, X = x\right], d, t = 0, 1$. When only repeated cross-section data are available, let $\mu^{rc}_{d,t}(x)$ be an arbitrary model for the true, unknown regression $m^{rc}_{d,t}(x) \equiv \mathbb{E}[Y \mid D = d, T = t, X = x], d, t = 0, 1$, and for, $d = 0, 1, \mu^{rc}_{d,Y}(T, X) \equiv T \cdot \mu^{rc}_{d,1}(X) + (1 - T) \cdot \mu^{rc}_{d,0}(X)$, and $\mu^{rc}_{d,\Delta}(X) \equiv \mu^{rc}_{d,1}(X) - \mu^{rc}_{d,0}(X)$.

```
#check if covariates and group are time invariant in the panel
    data case.
# matrix of covariates for pre-period and post periods
covariates_pre <- stats::model.matrix(xformla,
data=subset(dta, dta$post==0))
covariates_post <- stats::model.matrix(xformla,
data=subset(dta, dta$post==1))

d_pre <- subset(dta$D, dta$post==0)
d_post <- subset(dta$D, dta$post==1)
w_pre <- subset(dta$w, dta$post==0)
w_post <- subset(dta$w, dta$post==1)

if (panel) {
```

```
                    if (!all(covariates_pre==covariates_post)) {
                            stop("Error: covariates should be time
                               invariant, and there should be no missing
                               data.")
                    }
                    if (!all(d_pre==d_post)) {
                            stop("Error: group indicator must be time
                               invariant.")
                    }
                    if (!all(w_pre==w_post)) {
                            stop("Error: weights must be time invariant.")
                    }
            }
```

From this code snippet of DRDID, it can be inferred that covariates must be time-invariant variables in order to conduct DRDID estimation using panel data. However, treatmentD must be a time-dependent variable.

In panel data analysis, covariates can be either time-varying or time-invariant variables. However, for conducting DRDID estimation, covariates are generally assumed to be time-invariant variables. This is because one of the underlying assumptions of DRDID method based on panel data is that the trend of covariates should be parallel between the treatment and control groups over time.

On the other hand, treatmentD (the treatment variable) represents the time-dependent variable indicating the treatment status of the treatment and control groups. The treatment variable may vary over time between the treatment and control groups, which is the core of the DRDID method. By analyzing the differential effects of the time-dependent treatment variable, we can estimate the treatment effect.

Therefore, covariates are generally considered as time-invariant variables, while the treatment variable is typically a time-dependent variable. This combination allows us to utilize the DRDID method to estimate the causal effect and control for other potential confounding factors.

```
> eval_lalonde_cps$age[1:4]
[1] 42 43 51 51
> # ------------------------------------------------
> # Implement improved DR locally efficient DiD with panel data
> drdid(yname="re", tname = "year", idname = "id", dname = "experimental",
+       xformla= ~ age+ educ+ black+ married+ nodegree+ hisp+ re74,
+       data = eval_lalonde_cps, panel = TRUE)
Error in pre_process_drdid(yname = yname, tname = tname, idname = idname,  :
  Error: covariates should be time invariant, and there should be no missing data.
> #Implement "traditional" DR locally efficient DiD with panel data
> drdid(yname="re", tname = "year", idname = "id", dname = "experimental",
+       xformla= ~ age+ educ+ black+ married+ nodegree+ hisp+ re74,
+       data = eval_lalonde_cps, panel = TRUE, estMethod = "trad")
Error in pre_process_drdid(yname = yname, tname = tname, idname = idname,  :
  Error: covariates should be time invariant, and there should be no missing data.
```

图 1

For the case in which panel data are available, we consider the estimand

$$\tau^{dr,p} = \mathbb{E}\left[(w_1^p(D) - w_0^p(D, X; \pi))\left(\Delta Y - \mu_{0,\Delta}^p(X)\right)\right]$$

where, for a generic g ,

$$w_1^p(D) = \frac{D}{\mathbb{E}[D]}, \quad \text{and} \quad w_0^p(D, X; g) = \frac{g(X)(1-D)}{1-g(X)} / \mathbb{E}\left[\frac{g(X)(1-D)}{1-g(X)}\right].$$

```
#Compute the Pscore using the pscore.cal
source("pscore.cal.R")
source("loss.ps.cal.R")
source("loss.ps.IPT.R")
pscore.ipt <- pscore.cal(D, int.cov, i.weights = i.weights, n = n)
ps.fit <- as.vector(pscore.ipt$pscore)
ps.fit <- pmin(ps.fit, 1 - 1e-16)
#Compute the Outcome regression for the control group
source("wols.br.panel.R")
outcome.reg <- wols.br.panel(deltaY, D, int.cov, ps.fit, i.weights)
out.delta <-  as.vector(outcome.reg$out.reg)
#Compute Bias-Reduced Doubly Robust DiD estimators
dr.att.summand.num <- as.vector((1 - (1 - D)/(1 - ps.fit)) * (deltaY -
    out.delta))
dr.att <- mean(i.weights * dr.att.summand.num)/mean(D * i.weights)
```

2 DML

2.1 PLR model

Partially linear regression models (PLR), which encompass the standard linear regression model, play an important role in data analysis (Robinson, 1988). Partially linear regression models take the form

$$\begin{aligned} Y &= D\theta_0 + g_0(X) + \zeta, \quad \mathbb{E}(\zeta \mid D, X) = 0, \\ D &= m_0(X) + V, \quad \mathbb{E}(V \mid X) = 0, \end{aligned} \tag{1}$$

where Y is the outcome variable and D is the policy variable of interest. The high-dimensional vector $X = (X_1, \ldots, X_p)$ consists of other confounding covariates, and ζ and V are stochastic errors. Equation (1) is the equation of interest, and θ_0 is the main regression coefficient that we would like to infer. If D is conditionally exogenous (randomly assigned conditional on X), θ_0 has the interpretation of a structural or causal parameter. The causal diagram supporting such interpretation is shown in Figure 1. The second equation keeps track of confounding, namely the dependence of D on covariates/controls. The characteristics X affect the policy variable D via the function $m_0(X)$ and the outcome variable via the function $g_0(X)$. The partially linear model generalizes both linear

regression models, where functions g_0 and m_0 are linear with respect to a dictionary of basis functions with respect to X , and approximately linear models.

The two strategies rely on very different moment conditions for identifying and estimating θ_0 :

$$\begin{aligned} \mathrm{E}\left[\left(Y - D\theta_0 - g_0(X)\right) D\right] &= 0 \\ \mathrm{E}\left[\left(Y - D\theta_0\right)\left(D - E[D \mid X]\right)\right] &= 0 \\ \mathrm{E}\left[\left(\left(Y - E[Y \mid X]\right) - \left(D - E[D \mid X]\right)\theta_0\right)\left(D - E[D \mid X]\right)\right] &= 0 \end{aligned} \tag{2}$$

(1) - Regression adjustment; (2) - "propensity score adjustment" (3) - Neyman-orthogonal (semi-parametrically efficient under homoscedasticity).

Consider estimation based on (3)

$$\check{\theta}_0 = \left(\frac{1}{n}\sum_{i=1}^{n} \widehat{V}_i^2\right)^{-1} \frac{1}{n}\sum_{i=1}^{N} \widehat{V}_i \widehat{W}_i$$

$$\widehat{V} = D - \widehat{m}_0(Z), \widehat{W} = Y - \widehat{\ell}_0(Z)$$

Under mild conditions, can write

$$\begin{aligned} \sqrt{n}\left(\theta_0 - \theta_0\right) = & \underbrace{\left(\frac{1}{n}\sum_{i=1}^{n} V_i^2\right)^{-1} \frac{1}{\sqrt{n}}\sum_{i=1}^{n} V_i U_i}_{:=a^+} \\ & + \underbrace{\left(\frac{1}{n}\sum_{i=1}^{n} V_i^2\right)^{-1} \frac{1}{\sqrt{n}}\sum_{i=1}^{n} \left(m_0\left(Z_i\right) - \hat{m}_0\left(Z_i\right)\right)\left(\ell_0\left(Z_i\right) - \widehat{\ell}_0\left(Z_i\right)\right)}_{:=b^+} \end{aligned}$$

Given identification, double machine learning for a PLR proceeds as follows

(1) Estimate ℓ_0 and m_0 by $\hat{\ell}_0$ and $\hat{m}_0$, which amounts to solving the two problems of predicting Y and D using X , using any generic ML method, giving us estimated residuals

$$\hat{W} = Y - \hat{\ell}_0(X),$$

and

$$\hat{V} = D - \hat{m}_0(X).$$

The residuals should be of a cross-validated form, as explained below in Algorithm 1 or 2, to avoid biases from overfitting.

(2) Estimate θ_0 by regressing the residual $\hat{W}$ on $\hat{V}$. Use the conventional inference for this regression estimator, ignoring the estimation error in the residuals.

2.2 Interactive Regression Model (IRM)

We consider estimation of average treatment effects when treatment effects are fully heterogeneous and the treatment variable is binary, $D \in \{0, 1\}$. We consider vectors (Y, D, X) such that

$$Y = g_0(D, X) + U, \quad \mathbb{E}(U \mid X, D) = 0,$$
$$D = m_0(X) + V, \quad \mathbb{E}(V \mid X) = 0.$$

Since D is not additively separable, this model is more general than the partially linear model for the case of binary D . A common target parameter of interest in this model is the average treatment effect (ATE),

$$\theta_0 = \mathbb{E}\left[g_0(1, X) - g_0(0, X)\right]$$

Another common target parameter is the average treatment effect for the treated (ATTE),

$$\theta_0 = \mathbb{E}\left[g_0(1, X) - g_0(0, X) \mid D = 1\right]$$

IRM score: For estimation of the ATE parameter of the IRM model, we employ the score (score = "ATE")

$$\psi(W; \theta, \eta) := (g(1, X) - g(0, X)) + \tfrac{D(Y - g(1,X))}{m(X)} - \tfrac{(1-D)(Y - g(0,X))}{1 - m(X)} - \theta,$$
$$\eta = (g, m), \quad \eta_0 = (g_0, m_0)\,,$$

where W=(Y, D, X) and g and m map the support of (D, X) to $\mathbb{R}$ and the support of X to $(\epsilon, 1 - \epsilon)$, respectively, for some $\epsilon \in (0, 1/2)$, whose true values are given by

$$g_0(D, X) = \mathbb{E}[Y \mid D, X], \quad m_0(x) = \mathbb{P}[D = 1 \mid X]$$

The step to estimate the treatment effect is

DML. (Generic double machine learning with cross-fitting)

(1) Inputs: Choose a model (PLR, PLIV, IRM, IIVM), provide data $(W_i)_{i=1}^N$, a Neyman-orthogonal score function $\psi(W; \theta, \eta)$, which depends on the model being estimated, and specify machine learning methods for η .

(2) Train ML predictors on folds: Take a K -fold random partition $(I_k)_{k=1}^K$ of observation indices $[N] = \{1, \ldots, N\}$ such that the size of each fold I_k is $n = N/K$. For each $k \in [K] = \{1, \ldots, K\}$, construct a high-quality machine learning estimator

$$\hat{\eta}_{0,k} = \hat{\eta}_{0,k}\left((W_i)_{i \notin I_k}\right)$$

of η_0, where $x \mapsto \hat{\eta}_{0,k}(x)$ depends only on the subset of data $(W_i)_{i \notin I_k}$.

(3) For each $k \in [K]$, construct the estimator $\check{\theta}_{0,k}$ as the solution to the equation

$$\frac{1}{n} \sum_{i \in I_k} \psi\left(W_i; \check{\theta}_{0,k}, \hat{\eta}_{0,k}\right) = 0$$

The estimate of the causal parameter is obtained via aggregation

$$\tilde{\theta}_0 = \frac{1}{K}\sum_{k=1}^{K}\check{\theta}_{0,k}.$$

(4) Output: The estimate of the causal parameter $\tilde{\theta}_0$ as well as the values of the evaluated score function are returned.

We use Lasso as example:

```
lassoF <- function(datause, dataout, form_x, form_y, logit=FALSE, alp=
    alp, arg=arg, s=s){

        form            <- as.formula(paste(form_y, "~", form_x));

        if(logit==TRUE){
                fit             <- lm(form,  x = TRUE, y = TRUE, data=
                    datause);
                lasso           <- do.call(cv.glmnet, append(list(x=fit$x
                    [ ,-1], y=fit$y, family="binomial", alpha=alp), arg)
                    )
        }

        if(logit==FALSE){
                fit             <- lm(form,  x = TRUE, y = TRUE, data=
                    datause);
                lasso           <- do.call(cv.glmnet, append(list(x=fit$x
                    [ ,-1], y=fit$y,alpha=alp), arg))
        }

        fit.p           <- lm(form,  x = TRUE, y = TRUE, data=datause);
        yhatuse         <- predict(lasso, newx=fit.p$x[,-1], s=s)
        if(logit==TRUE){yhatuse         <- predict(lasso, newx=fit.p$x
            [,-1], s=s, type="response")}
        resuse          <- fit.p$y - yhatuse
        xuse            <- fit.p$x

        fit.p           <- lm(form,  x = TRUE, y = TRUE, data=dataout);
        yhatout         <- predict(lasso, newx=fit.p$x[,-1], s=s)
        if(logit==TRUE){ yhatout         <- predict(lasso, newx=fit.p$x
            [,-1], s=s, type="response")}
        resout          <- fit.p$y - yhatout
        xout            <- fit.p$x

        return(list(yhatuse = yhatuse, resuse=resuse, yhatout = yhatout
            , resout=resout, xuse=xuse, xuse=xout, model=lasso, yout=fit
```

```
                    .p$y, form=form));

        }
```

2.3 PLR Code

```
    if(binary==1){
       fit.z      v<- lassoF(datause=datause, dataout=dataout,  form_x, form_d
           , logit=TRUE, alp=alp, arg=arg, s=s)
       mis.z          <- error(fit.z$yhatout, dataout[,d])$mis
    }

    fit.y      <- lassoF(datause=datause, dataout=dataout,  form_x, form_y,
        alp=alp, arg=arg, s=s)

    err.z          <- error(fit.z$yhatout, dataout[,d])$err
    mz_x           <- fit.z$yhatout
    rz             <- fit.z$resout
    err.z          <- error(fit.z$yhatout, dataout[,d])$err

    ry             <- fit.y$resout
    my_x           <- fit.y$yhatout
    err.y          <- error(fit.y$yhatout, dataout[,y])$err
```

```
    if(est=="plinear"){

       lm.fit.ry              <- lm(as.matrix(cond.comp[[min1,j]]$ry) ~ as.
          matrix(cond.comp[[min2,j]]$rz)-1);
       ate                    <- lm.fit.ry$coef;
       HCV.coefs              <- vcovHC(lm.fit.ry, type = 'HC');
       STE[1,(k+1)]           <- (1/(K^2))*(diag(HCV.coefs)) +  STE[1,(k+1)]
       TE[1,(k+1)]            <- ate/K + TE[1,(k+1)] ;

    }
```

2.4 IRM Code

```
    if(plinear==0){

       fit.yz1        <- RF(datause=datause[ind_u,], dataout=dataout[ind_o,],
          form_x=form_x,  form_y=form_y, nodesize=option[["reg_nodesize"]],
          arg=arg, tune=tune)
       err.yz1        <- error(fit.yz1$yhatout, dataout[ind_o,y])$err
```

```
    my_z1x          <- predict(fit.yz1$model, dataout, type="response")

    fit.yz0         <- RF(datause=datause[-ind_u,], dataout=dataout[-ind_o
        ,],  form_x=form_x, form_y=form_y, nodesize=option[["reg_nodesize"
        ]], arg=arg, tune=tune)
    err.yz0         <- error(fit.yz0$yhatout, dataout[-ind_o,y])$err
    my_z0x          <- predict(fit.yz0$model, dataout, type="response")

}

if(binary==1){
    fit.z           <- RF(datause=datause, dataout=dataout,  form_x=form_x,
        form_y=form_d, nodesize=option[["clas_nodesize"]], arg=arg, reg=TRUE
        , tune=tune)
    mis.z           <- error(as.numeric(fit.z$yhatout), dataout[,y])$mis
}
fit.y           <- RF(datause=datause, dataout=dataout,  form_x=form_x,
    form_y=form_y, nodesize=option[["reg_nodesize"]],  arg=arg, tune=tune)
```

```
ATE <- function(y, d, my_d1x, my_d0x, md_x)
{
    return( mean( (d * (y - my_d1x) / md_x) -  ((1 - d) * (y - my_d0x) / (1
         - md_x)) + my_d1x - my_d0x ) );
}
```

```
if(est=="interactive"){

    drop                    <- which(cond.comp[[min3,j]]$mz_x>trim[1] & cond
        .comp[[min3,j]]$mz_x<trim[2])
    mz_x                    <- cond.comp[[min1,j]]$mz_x[drop]
    my_z1x                  <- cond.comp[[min2,j]]$my_z1x[drop]
    my_z0x                  <- cond.comp[[min3,j]]$my_z0x[drop]
    yout                    <- dataout[drop,y]
    dout                    <- dataout[drop,d]

    TE[1,(k+1)]             <- ATE(yout, dout, my_z1x, my_z0x, mz_x)/K + TE[
        1,(k+1)];
    STE[1,(k+1)]            <- (1/(K^2))*((SE.ATE(yout, dout, my_z1x, my_z0x
        , mz_x))^2) + STE[1,(k+1)];

}
```

3 Difference

DR (Doubly Robust) and DML (Double Machine Learning) are similar in that they both involve multiple modeling steps.

Similarities: In the first stage, both methods use machine learning models to estimate the propensity scores and the target variable Y. In the second stage, causal effects are evaluated.

Differences: The main difference lies in how the target variable Y is estimated in the first stage. In DR, both X and Treatment are used as features to estimate Y. On the other hand, DML focuses on estimating the treatment effect by using machine learning models to estimate the conditional expectation of Y given X and Treatment separately, and then combining these estimates.

While both methods involve several steps, there are significant differences. DR requires the calculation of propensity scores, whereas DML focuses on estimating the treatment effect using separate models for X and Treatment.