LASSO/Poisson DML implementation

Thomas R. Covert, Yixin Sun, and Richard Sweeney

December 12, 2020

Contents

1 Introduction

This package implements the Double Machine Learning approach from Chernozhukov et al. (2018), which constructs estimates for low-dimensional target parameters in the presence of high-dimensional nuisance parameters. We follow the set-up introduced in the paper:

$$Y = D\theta_0 + g_0(X) + \zeta, \quad E[\zeta \mid D, X] = 0, \tag{1}$$

$$D = m_0(X) + V, \qquad E[V \mid X] = 0, \tag{2}$$

where Y is the outcome variable and D is the variable of interest. The vector X consists of other confounding covariates, and ζ and V are the stochastic errors. The first equation is the equation of interest, and θ_0 is the target parameter. The goal here is to estimate the high-dimensional nuisance parameters using machine learning methods, which allow for causal estimation and valid inference of the target parameters. We implement this using lasso and regression forest methods.

The key here is constructing the "right" moment conditions to solve, such that small deviations in the nuisance functions do not invalidate the moment conditions. See paper for theoretical underpinnings of the derived moment condition, which Chernozhukov et al. (2018) refer to as the *Neyman Orthogonal Score*. The paper provides two ways to construct the Neyman Orthogonal Score, which we refer to as (1) Finite Nuisance Parameter Approach and (2) Concentrating Out Approach.

2 Finite Nuisance Parameter Approach

Let $W = (Y, D, X)$. The true values of θ and β, denoted as θ_0 and β_0, fit the data best, in the sense that

$$(\theta_0, \beta_0) = \operatorname{argmax} \underset{\theta,\beta}{E}_W [l(W, \theta, \beta)]$$

where $l(W, \theta, \beta)$ is some criterion (squared deviation, log likelihood etc).

The *Neyman Orthogonal Score* ψ is defined by:

$$\psi(W, \theta, \beta, \mu) = \frac{\partial}{\partial\theta} l(W, \theta, \beta) - \mu \frac{\partial}{\partial\beta} l(W, \theta, \beta)$$

The vector μ above is defined by the hessian of this criterion function. Let J be:

$$J = \begin{pmatrix} J_{\theta,\theta} & J_{\theta,\beta} \\ J_{\beta,\theta} & J_{\beta,\beta} \end{pmatrix} = \frac{\partial}{\partial\theta\partial\beta} E_W \left[\frac{\partial}{\partial\theta\partial\beta} l(W, \theta, \beta) \right]$$

Then we define μ as $\mu = J_{\theta,\beta} J_{\beta,\beta}^{-1}$.

2.1 The Linear Setting

In the linear regression, the function l is

$$l(W;\theta,\beta) = -\frac{(Y - D\theta - X'\beta)^2}{2}$$

and the necessary gradients needed for ψ are:

$$\partial\ell_\theta(W;\theta,\beta) = \left(Y - D\theta - X'\beta\right) D$$
$$\partial\ell_\beta(W;\theta,\beta) = \left(Y - D\theta - X'\beta\right) X$$

The entries in the Hessian matrix that we need to compute μ are:

$$J_{\theta\beta} = -E\left[DX'\right]$$
$$J_{\beta\beta} = -E\left[XX'\right]$$

yielding this expression for μ:

$$\mu = E\left[DX'\right]\left(E\left[XX'\right]\right)^{-1}$$

The Neyman orthogonal score is then given by:

$$\psi(W;\theta,\eta) = \left(Y - D\theta - X'\beta\right)(D - \mu X)$$

2.1.1 Implementation - LASSO

These steps give a single point estimate, $\hat{\theta}$ and an associated covariance matrix for a given split structure. See below for how we combine point estimates and covariance matrices across many split structures into a single point estimate/covariance matrix that should be less sensitive to the monte carlo nature of splitting.

1. Make k splits of the data into training and estimation sets. Default is $k = 5$ folds in our implementation

2. For each $k = 1, ...K$, implement the following steps:

   (a) In a **training** set k, use a linear LASSO of Y on D and X to select the covariates (making sure D is always included), $\hat{X}_k$. Then fit a linear regression of Y on $\hat{X}_k$ and D, and let $\hat{\beta}_k$ be

the estimated coefficients on $\hat{X}_k$.

(b) In the **estimation** set k, compute $s_k = \hat{X}_k \hat{\beta}_k$.

(c) In the **training** set k, compute a linear LASSO of D_j on X to select covariates $\tilde{X}_{k,j}$, for each variable of interest. Fit a linear regression of D_j on $\tilde{X}_{k,j}$, and denote this $\tilde{\mu}_{k,j}$. Collect the $\tilde{\mu}_{k,j}$ into a vector to form $\tilde{\mu}_k$.

(d) In the **estimation** set, construct

$$m_k = \tilde{X}_k \tilde{\mu}_k$$

3. Using the DML2 algorithm, for each k, construct the average of the moment:

$$\frac{1}{n} \sum_{i=1}^{n} (D_i - m_i)' \left(Y_i - s_i - D_i \theta' \right)$$

Then average over each of these folds to get the final objective function, which we use to compute $\hat{\theta}$ by minimizing squared deviations from zero (if D is univariate, we can just do root-finding). Note, this is also different from what STATA does. It seems like they compute this moment in one step using all the data, and ignore the hold out structure.

4. To get a covariance matrix for this estimate of θ, we first compute J_0 defined by:

$$\begin{aligned} J_0 &= \frac{\partial}{\partial \theta} E_W \left[\psi(W, \hat{\theta}, \hat{\beta}) \right] \\ &= -E_W \left[D'(D - m) \right] \end{aligned}$$

Next we compute Ψ :

$$\Psi = E_W \left[\psi(W, \theta, \tilde{\theta}, \hat{\beta}) \psi(W, \theta, \tilde{\theta}, \hat{\beta})' \right]$$

$$\psi(W, \hat{\theta}, \tilde{\theta}, \hat{\beta}) = (Y_i - s_i - D_i \hat{\theta}')(D_i - m_i)$$

And we can compute:

$$\hat{\text{Var}}(\hat{\theta}) = \frac{1}{n} J_0^{-1} \Psi J_0^{-1}$$

2.2 The Poisson Setting

In Poisson regression, the function l is

$$l(W; \theta, \beta) = Y(D\theta + X\beta) - \exp(D\theta + X\beta)$$

and its associated gradients needed for the definition of ψ are

$$\frac{\partial}{\partial\theta}l(W,\theta,\beta) = (Y - \exp(D\theta + X\beta))D$$
$$\frac{\partial}{\partial\beta}l(W,\theta,\beta) = (Y - \exp(D\theta + X\beta))X$$

The entries in the Hessian matrix that we need to compute μ are:

$$J_{\theta,\theta} = -E\left[D'D\exp(D\theta + X\beta)\right]$$
$$J_{\theta,\beta} = -E\left[D'X\exp(D\theta + X\beta)\right]$$
$$J_{\beta,\beta} = -E\left[X'X\exp(D\theta + X\beta)\right]$$

yielding this expression for μ:

$$\mu = E\left[D'X\exp(D\theta + X\beta)\right]\left(E\left[X'X\exp(D\theta + X\beta)\right]\right)^{-1}$$

This construction is revealing, since it looks like weighted least squares, with D as the outcome, X as the covariates, and weights equal to $\exp(D\theta + X\beta)$.

The Neyman Orthogonal moment for Poisson regression is then:

$$\psi = (Y - \exp(D\theta + X\beta))(D - X\mu)$$

2.2.1 Implementation - Poisson LASSO

1. Make k splits of the data into training and estimation sets. Default is $k = 5$ in our implementation.
2. For each $k = 1, ..., K$, implement the following steps:
   (a) In a **training** set k, use Poisson LASSO of Y on D and X to select the covariates, $\hat{X}_k$ (making sure D is always included). Then fit a Poisson regression of Y on $\hat{X}_k$ and D, and let $\hat{\beta}_k$ be the estimated coefficients on $\hat{X}_k$.
   (b) In the corresponding **estimation** set k, compute $s_k = \hat{X}_k\hat{\beta}_k$.
   (c) Back in the **training** set k, compute weights $w_k = \exp(D\hat{\theta}_k + X\hat{\beta}_k)$ using the results from step 2. Compute a linear LASSO of D_j on X using those weights to select covariates $\tilde{X}_{k,j}$ for each variable of interest. Fit a linear regression of D_j on $\tilde{X}_{k,j}$, and denote this $\tilde{\mu}_{k,j}$. Collect the $\tilde{\mu}_{k,j}$

into a vector to form $\tilde{\mu}_k$.

(d) In **estimation** set, construct

$$m_k = \tilde{X}_k \tilde{\mu}_k$$

3. Using the DML2 algorithm, for each k, construct the average of the moment:

$$\frac{1}{n}\sum_{i=1}^{n}(D_i - m_i)'\left(Y_i - \exp(s_i - D_i\theta')\right)$$

Then average over each of these folds to get the final objective function, which we use to compute $\hat{\theta}$ by minimizing squared deviations from zero (if D is univariate, we can just do root-finding).

4. To get a covariance matrix for this estimate of θ, we first compute J_0, defined by:

$$\begin{aligned} J_0 &= \frac{\partial}{\partial\theta} E_W\left[\psi(Y, D, X, \hat{\theta}, \tilde{\theta}, \tilde{\beta})\right] \\ &= -E_W\left[D' \exp(D\hat{\theta} + s)(D - X\hat{\mu})\right] \end{aligned}$$

Next we compute Ψ:

$$\begin{aligned} \Psi &= E_W\left[\psi(W, \hat{\theta}, \tilde{\theta}, \tilde{\beta})\psi(W, \hat{\theta}, \tilde{\theta}, \tilde{\beta})'\right] \\ &= E_W\left[(Y - \exp(D\hat{\theta} + s))^2 (D - X\tilde{\mu})(D - X\tilde{\mu})'\right] \end{aligned}$$

In both cases, we compute each of these as the average over points in the estimation set k, and then average over each of the estimation sets within a split structure.[1] We then get:

$$\hat{Var}(\hat{\theta}) = \frac{1}{n} J_0^{-1} \Psi J_0^{-1}$$

3 Concentrating Out Approach

Chernozhukov et al. (2018) write that the approach for constructing Neyman orthogonal scores is closely related to the "concentrating-out approach". For all $\theta \in \Theta$, let β_θ be the solution of the following

[1] For an example of this averaging, see the formula for $\hat{J}_0$ on page C27 of the original DML paper.

optimization problem:

$$\underset{\beta \in \mathcal{B}}{\operatorname{argmax}} E[\ell(W; \theta, \beta)]$$

where β_θ satisfies

$$\partial_\beta E\left[\ell\left(W; \theta, \beta_\theta\right)\right] = 0$$

Differentiating this with respect to θ and interchanging the order of differentiation gives us

$$\begin{aligned} 0 &= \partial_\theta \partial_\beta E\left[\ell\left(W; \theta, \beta_\theta\right)\right] = \partial_\beta \partial_\theta E\left[\ell\left(W; \theta, \beta_\theta\right)\right] \\ &= \partial_\beta E\left[\partial_\theta \ell\left(W; \theta, \beta_\theta\right) + \left[\partial_\theta \beta_\theta\right]' \partial_\beta \ell\left(W; \theta, \beta_\theta\right)\right] \\ &= \partial_\beta E\left[\psi\left(W; \theta, \beta, \partial_\theta \beta_\theta\right)\right]|_{\beta=\beta_\theta} \end{aligned}$$

So our score function here is:

$$\psi\left(W; \theta, \beta, \partial_\theta \beta_\theta\right) := \partial_\theta \ell(W; \theta, \beta) + \left[\partial_\theta \beta_\theta\right]' \partial_\beta \ell(W; \theta, \beta)$$

3.1 The Linear Setting

Consider again the function:

$$\ell(W; \theta, \beta) = -\frac{(Y - D\theta - \beta(X))^2}{2}$$

Taking FOC with respect to β, we get:

$$\begin{aligned} 0 &= E[Y - D\theta - \beta(X)|X] \\ 0 &= E[Y - D\theta|X] - \beta(X) \\ \Rightarrow \beta(X) &= E[Y - D\theta|X] \end{aligned}$$

Letting $g(x) = E[Y|X]$ and $m(x) = E[D|X]$, we get the following Neyman orthogonal score

$$\begin{aligned} \psi\left(W; \theta, \beta_\theta\right) &= -\frac{1}{2} \frac{d\left\{Y - D\theta - E[Y - D\theta|X]\right\}^2}{d\theta} \\ &= (D - E[D|X]) \times (Y - E[Y|X] - (D - E[D|X])\,\theta) \\ &= (D - m(X)) \times (Y - g(x) - (D - m(x))\theta) \end{aligned}$$

Note that we now use ML methods to calculate $E[D|X]$ and $E[Y|X]$, which we plug into the score

function we derived above in a Frisch-Waugh-Lovell style method. This is different from the finite-nuisance parameter approach, which uses ML to calculate the finite-nuisance parameter component, $s = X\beta$. The implication of this is that the ML steps for the concentrating-out approach are fairly uniform across ML methods and model types; as you can see below, what changes is just the score function that we are optimizing over.

3.1.1 Implementation - LASSO

1. Make k splits of the data into training and estimation sets. Default is $k = 5$ folds in our implementation.

2. For each $k = 1, ...K$, implement the following steps:

   (a) In a **training** set k, use a linear LASSO to generate a model of Y on and X to select the covariates, $\hat{X}_k$. Then fit a linear regression of Y on $\hat{X}_k$, and let $\hat{\beta}_k$ be the estimated coefficients on $\hat{X}_k$.

   (b) In the **estimation** set k, compute $l_k = E[Y|X]$ using the model generated in the previous step.

   (c) In the **training** set k, compute a linear LASSO of D_j on X to select covariates $\tilde{X}_{k,j}$ for each variable of interest. Fit a linear regression of D_j on $\tilde{X}_{k,j}$, and denote the estimated coefficients as $\tilde{\mu}_{k,j}$. Collect the $\tilde{\mu}_{k,j}$ into a vector to form $\tilde{\mu_k}$.

   (d) In the **estimation** set, compute $m_k = E[D|X]$, using the model generated from the previous step.

3. Using the DML2 algorithm, for each k, construct the average of the moment:

$$\frac{1}{n}\sum_{i=1}^{n}(D_i - m_i)'\left(Y_i - g_i - (D_i - m_i)\theta'\right)$$

   Then average over each of these folds to get the final objective function, which we use to compute $\hat{\theta}$ by minimizing squared deviations from zero.

4. To get a covariance matrix for this estimate of θ, we first compute J_0 defined by:

$$\begin{aligned} J_0 &= \frac{\partial}{\partial\theta}E_W\left[\psi(W, \hat{\theta}, \hat{\beta})\right] \\ &= -E_W\left[(D - m)'(D - m)\right] \end{aligned}$$

Next we compute Ψ :

$$\Psi = E_W \left[\psi(W, \theta, \tilde{\theta}, \hat{\beta}) \psi(W, \theta, \tilde{\theta}, \hat{\beta})' \right]$$

$$\psi(W, \hat{\theta}, \tilde{\theta}, \hat{\beta}) = (Y_i - g_i - (D_i - m_i)\hat{\theta}')(D_i - m_i)$$

And we can compute:

$$\hat{\text{Var}}(\hat{\theta}) = \frac{1}{n} J_0^{-1} \Psi J_0^{-1}$$

3.1.2 Implementation - Regression Forest

1. Make k splits of the data into training and estimation sets. Default is $k = 5$ folds in our implementation

2. For each $k = 1, ...K$, implement the following steps:

   (a) In **training** set k, use a regression forest to generate a model of Y on and X. Use this trained regression forest and apply to data in the **estimation** set to calculate $l_k = E[Y|X]$.

   (b) In **training** set k, use a regresison forest to generate a model of D_j on X. Use this trained regression forest and apply to data in the **estimation** set to calculate $m_{k,j} = E[D_j|X]$. Collect to form $m_k = (m'_{k,1}, m'_{k,2}, ..., m'_{k,J})$

3. The rest of the implementation should follow steps 3 and 4 of the lasso instructions in the previous section.

3.2 The Poisson Setting

We have only solved the orthogonal score for the case when D is a univariate binary variable. Consider again for the regression case, the function l is

$$l(W; \theta, \beta) = Y(D\theta + X\beta(X)) - \exp(D\theta + \beta(X))$$

Taking FOC with respect to β, we get:

$$\begin{aligned}
\beta_\theta(X) &= E[\exp(D\theta + \beta(X))|X] \\
&= \exp(\beta(X))E[\exp(D\theta)|X] \\
&= \exp(\beta(x))\left(Pr(D=1|X)\exp(\theta) + (1 - Pr(D=1|X))\right) \\
&= \exp(\beta(x))\left(E[D|X]\exp(D\theta) + (1 - E[D|X])\right)
\end{aligned}$$

As before, let $g(X) = E[Y|X]$ and $m(X) = E[D|X]$ and rearrange to get:

$$\exp\left(\beta_\theta(X)\right) = \frac{g(X)}{\exp(\theta)m(X) + 1 - m(X)}$$

From this, we have the following:

$$\begin{aligned}
\beta_\theta(X) &= \log\left(\frac{g(X)}{\exp(\theta)m(X) + 1 - m(X)}\right) \\
\partial_\theta\beta_\theta &= -\frac{m(X)\exp(\theta)}{m(X)\exp(\theta) + (1 - m(x))}
\end{aligned}$$

Putting this altogether and plugging into the score function, we get:

$$\psi(W;\theta, g(X), m(X)) = \left(Y - \frac{\exp(D\theta)g(X)}{\exp(\theta)m(X) + 1 - m(X)}\right)\left(D - \frac{\exp(\theta)m(X)}{\exp(\theta)m(X) + 1 - m(X)}\right)$$

3.2.1 Implementation - Poisson

Steps 1 and 2 are the same as the linear model in the previous section.

3. Using the DML2 algorithm, for each k, construct the average of the moment:

$$\frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \frac{\exp(D_i\theta)g_i}{\exp(\theta)m_i + 1 - m_i}\right)\left(D_i - \frac{\exp(\theta)m_i}{\exp(\theta)m_i + 1 - m_i}\right)$$

4. To get a covariance matrix for this estimate of θ, we first compute J_0 defined by

$$J_0 = \frac{\partial}{\partial\theta}E_W\left[\psi(W, \hat{\theta}, \hat{\beta})\right]$$

To break down this calculation, let

$$P_i = \exp(\theta)m_i + 1 - m_i$$
$$A_i = \frac{\exp(D_i\theta)g_i}{P}$$
$$B_i = \frac{\exp(\theta)m_i}{P}$$

So we have

$$J_0 = E_W\left[A \times \left(\frac{\exp(\theta)m}{P} + \frac{\exp(\theta)^2 g^2}{P^2}\right) + B \times \left(\frac{D\exp(D\theta)g}{P} + \frac{\exp(D\theta)g}{P^2}\right)\right]$$

As always, we have

$$\Psi = E_W\left[\psi(W, \theta, \tilde{\theta}, \hat{\beta})\psi(W, \theta, \tilde{\theta}, \hat{\beta})'\right]$$

And we can compute

$$\hat{\text{Var}}(\hat{\theta}) = \frac{1}{n}J_0^{-1}\Psi J_0^{-1}$$

3.2.2 Implementation - Regression Forest

Steps 1 and 2 follow the same steps as the instructions in section 2.1.2, while steps 3 and 4 follow the Poisson LASSO in the previous section.

References

Chernozhukov, Victor, Denis Chetverikov, Mert Demirer, Esther Duflo, Christian Hansen, Whitney Newey, and James Robins, "Double/debiased machine learning for treatment and structural parameters," *The Econometrics Journal*, February 2018, *21* (1), C1–C68.