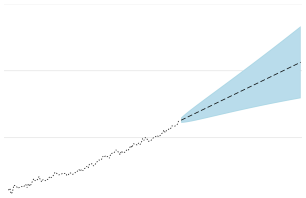
友万科技



机器学习因果推断与Stata应用

王群勇（南开大学数量经济研究所）

数量经济学国际讲习班暨第九届Stata中国用户大会

长春，2025年7月11日

主要内容：

- 机器学习与因果推断
- 局部线性模型（双重机器学习）
- 交互模型（双重稳健学习、元学习器）
- 机器学习因果推断（DML-IV、DML-RD、DML-DID、DML-SS）
- 政策学习

机器学习的目标在于得到更准确的预测，而回归模型的目标在于得到因变量y与自变量X之间的因果关系。因此，在计量经济模型为了得到真实的因果关系往往以牺牲模型的拟合优度为代价[@AtheyImbens2019]，而标准的机器学习也以牺牲参数估计量的一致性为代价。

"It is very common to see that a predictive model (e.g. least squares regression) might have very high explanatory power (e.g. high R2), while the causal model (e.g. instrumental variables regression) might have very low explanatory power (in terms of predicting outcomes). In other words, economists typically abandon the goal of accurate prediction of outcomes in pursuit of an unbiased estimate of a causal parameter of interest." (Susan, 2018)

计量模型得到因果关系的必要条件是充分控制混淆因素：既要把影响因变量的因素全部加入到模型中，还要正确地设定模型的形式。有时变量的个数比较多，甚至比样本量还要大，如果再加入平方项或交互项，那么维度变得更高。即使非混淆性成立，但模型不能充分体现变量的影响，那么也会导致估计量的不一致性。

- 维度过高：变量选择（逐步回归、贝叶斯选择等）
- 非线性：为了捕捉变量的各种可能的非线性关系，经常采用非参方法。但非参方法存在维度诅咒的问题。

机器学习方法在处理高维问题和非线性问题上表现出突出的优势。将机器学习与因果推断结合起来得到迅速发展和应用[@Athey2019;@Chernozhukov2018; @Wager2018]。

"I believe that regularization and systematic model selection have many advantages over traditional approaches, and for this reason will become a standard part of empirical practice in economics. " (Susan, 2018)

主要内容：

- 机器学习在经济学中的应用概述
- **局部线性模型（双重机器学习）**
  - 异质处理效应
  - 双重机器学习（因果森林）
- 交互模型（双重稳健学习、元学习器）
- 机器学习因果推断（DML-IV、DML-RD、DML-DID、DML-SS）
- 政策学习

异质处理效应

当处理效应在总体中为常数时，平均处理效应（ATE）描述了处理效应的整体分布情况。然而，当处理效应存在异质性，且不同个体对相同处理的反应不同时，仅估计处理效应的均值可能会掩盖处理如何影响不同个体的潜在机制。例如，当某些群体经历正向效应，而其他群体经历负向效应时，估计得到的ATE可能接近于零。

CATE即是用于刻画异质处理效应。与ATE不同，CATE是在总体的子群体上计算得出的。CATE可以帮助我们理解处理效应是否是异质的，处理效应如何随着一些变量发生变化，不同群体的处理效应有什么差别，处理效应不同的群体有什么不同的特征等等。

同时，CATE也是优化分配规则的重要基础。

令$y(1)$表示个体在处理组的潜在结果，$y(0)$表示个体在控制组的潜在结果。处理效应的反事实定义：$\theta = E[y(1) - y(0)]$。CATE函数为

$$\theta(X) = E[y(1) - y(0)|X]$$

CATE:

$$\theta(t_0, t_1, x) = E[\theta(X)|X = x] \cdot (t_1 - t_0)$$

- 个体平均处理效应（Individual ATE，简写为IATE）：处理效应随每个个体是变化的。
- 组平均处理效应（Group ATE，简写为GATE）：处理效应对于不同的分组是不同的。
- 排序组平均处理效应（Sorted Group ATE，简写为GATES）是指按照IATE的分位数分组，ATE对于不同的分组是不同的。

元学习器（Meta-Learner）

`S-Learner`（Single Learner）：将处理变量 T 作为特征加入回归模型，直接拟合 $\mathbb{E}[Y|T,X]$。

设$y=f(T,X)+U$。如果T为二值变量：将所有观测值的T赋值为1，计算$\hat{f}(T=1,X)$；然后将所有观测值的T赋值为0，计算$\hat{f}(T=0,X)$。处理效应为

$$\theta(X)=\hat{f}(T=1,X)-\hat{f}(T=0,X)$$

如果T为连续变量：将所有观测值的T赋值为a，计算$\hat{f}(T=a,X)$；然后将所有观测值的T赋值为b，计算$\hat{f}(T=b,X)$。T从a变化到b的处理效应为

$$\theta(X)_{a\to b}=\hat{f}(T=b,X)-\hat{f}(T=a,X)$$

`T-Learner`（Two Learner）[@Kunzel2019]: 分别用处理组和对照组数据训练两个独立的回归模型：$\mu_0(X)$ 和 $\mu_1(X)$，再计算差值（适用于非参数模型，如随机森林）。

第1步：

$$\mu_0(x) = E[Y|T=0, X=x]$$
$$\mu_1(x) = E[Y|T=1, X=x]$$

第2步：CATE估计量为

$$\hat{\theta}(x) = \hat{\mu}_1(x) - \hat{\mu}_0(x)$$

`X-Learner`（Cross-Learner）:将数据集随机分为两个互不重叠的子样本S_1和S_2（通常按 50% 比例分割），用于交叉拟合以降低偏差。

第1步：估计结果模型（Outcome Model）。对于处理组$(T=1)$，拟合 $E[Y|T=1,X]$，记为 $\hat{\mu}_1(X)$。对于对照组$(T=0)$，拟合 $E[Y|T=0,X]$，记为 $\hat{\mu}_0(X)$。

第2步：分别计算$T=\{0,1\}$的潜在结果：

$$D_i^1 = Y_i^1 - \hat{\mu}_0(X_i^1)$$
$$D_i^0 = \hat{\mu}_1(X_i^0) - Y_i^0$$

第3步：对$T=\{0,1\}$分别估计

$$\tau_1(x) = E[D^1|X=x], \tau_0(x) = E[D^0|X=x]$$

第4步：设$g(x)$为倾向得分，CATE估计量为：

$$\theta(x) = g(x)\tau_0(x) + (1-g(x))\tau_1(x)$$

最后，可以用$\theta(x)$对某些变量回归，得到$\theta(x)$的函数形式。

`R-Learner` （Residual-Learner） [@Nie2021]:

第1步：利用交互校正的方法预测$E(y|X)$，得到$\hat{g}^{(-i)}(x_i)$，表示用其他观测值估计模型来预测第i组样本的预测值，整个样本的预测表示为$\hat{g}(x)$。类似地， $E(T|X)$的预测值表示为$\hat{m}^{(-i)}(x_i)$和$\hat{m}(X)$。

第2步：CATE估计量$\theta(X)$为最小化下面的目标函数：

$$\hat{L}_n(\theta(x)) = \frac{1}{n}\sum_{i=1}^{n}\left[(Y_i - \hat{g}(X_i)) - (T_i - \hat{m}(X_i))\theta(X_i)\right]^2$$

领域自适应学习器（Domain Adaptation Learner）通过领域自适应技术来估计结果模型$\mu_0(X)$（对照组）和$\mu_1(X)$（处理组）。该方法的核心假设是：处理组与对照组的数据分布存在差异（即$P(X|A=1) \neq P(X|A=0)$），因此需要通过样本权重调整分布差异，使模型在跨领域（处理组与对照组）预测时保持无偏性。

$$\hat{\mu}_0 = M_1\left(Y^0 \sim X^0, \text{weights} = \frac{g(X^0)}{1-g(X^0)}\right)$$

$$\hat{\mu}_1 = M_2\left(Y^1 \sim X^1, \text{weights} = \frac{1-g(X^1)}{g(X^1)}\right)$$

$$\hat{T}^1 = Y^1 - \hat{\mu}_0(X^1)$$

$$\hat{T}^0 = \hat{\mu}_1(X^0) - Y^0$$

$$\hat{\theta} = M_3(\hat{T}^0|\hat{T}^1 \sim X^0|X^1)$$

```
. use dmlirm, clear
. econmlcate y X1-X10, treat(d) tdisc(True) learner(SLearner) modely("rfr") modelt("logit") nrepbs(50)

Learner    =              SLearner        No. of obs.   =          500
E(Y|T,X)   =                   rfr        No. bootstrap =           50
------------------------------------------------------------------------------
             |              Bootstrap
           y | Coefficient  std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
           d |   .6151163   .4189633     1.47   0.142    -.2060366    1.436269
------------------------------------------------------------------------------


. capture drop _eff*
. svmat e(eff),names(_eff)
. list _eff* in 1/5

     +-------------------------------------------------------------------+
     |     _eff1      _eff2       _eff3      _eff4       _eff5     _eff6 |
     |-------------------------------------------------------------------|
  1. |  .4773416   .1971785     2.42086   .0154838   -.0121484   .7607914 |
  2. |  .5283275   .1473942    3.584452   .0003378    .2435604   .8213457 |
  3. | -1.742738   .2996233   -5.816432   6.01e-09   -2.078237   -.903714 |
  4. |  .9235638   .2487895     3.71223   .0002054    .3756183   1.350873 |
  5. | -.3504456   .2901254   -1.207911   .2270816   -.8098537   .3274379 |
```

局部线性模型

第一个方程为结果方程，第二个方程为处理分配方程：

$$
\begin{aligned}
Y &= \theta(X) \cdot T + g(X, W) + U \\
T &= m(X, W) + V
\end{aligned}
$$

目标是估计平均处理效应$\theta(X)$。干扰（nuisance）参数模型：

$$
\begin{aligned}
E(Y|X, W) &= m(X, W)\theta(X) + g(X, W) = l(X, W) \\
E(T|X, W) &= m(X, W) \\
Y - E(Y|X, W) &= \theta(X) \cdot (T - m(X, W)) + U
\end{aligned}
$$

令$l(X,W)=E[Y|X,W]$，$m(X,W)=E[T|X,W]$，定义两个残差$\tilde{Y}$和$\tilde{T}$，

$$\tilde{Y}=\theta(X)\cdot\tilde{T}+U$$

定义纽曼得分函数：

$$\psi(.)=[Y-l(X)-\theta(X)(T-m(X))](T-m(X))$$

纽曼正交：$E[\psi(.)]=0$

如果θ为常数，由纽曼正交可得处理效应的partial-out估计量，

$$\theta=\frac{E[(Y-l(X))(T-m(X))]}{E[(T-m(X))(T-m(X))]}$$

$$\theta=\frac{E[(Y-l(X))(T-m(X))]}{E[(T-m(X))^2]}$$

$$\hat{\theta}=\frac{\sum_{i=1}^{N}(Y_i-l(X_i))(D_i-m(X_i))}{\sum_{i=1}^{N}(D_i-m(X_i))(D_i-m(X_i))}$$

双重机器学习通过两步来估计θ。

- 第一步，利用机器学习交互拟合的方法估计$l(x)$和$m(x)$。
- 第二步，用Y的残差$Y-\hat{l}(X)$对T的残差$T-\hat{m}(X)$进行回归。

假定1（非混淆性）：$E(U \mid X, W)=0$，$E(V \mid X, W)=0$。

假定2（纽曼正交）：$E(U \cdot V \mid X, W)=0$。

"Thus, the ML approach of evaluating performance in a test set does not address this concern at all. Instead, ML is likely to help make estimation methods more credible, while maintaining the identifying assumptions: in practice, coming up with estimation methods that give unbiased estimates of treatment effects requires flexibly modeling a variety of empirical relationships, such as the relationship between the treatment assignment and covariates. Since ML excels at data-driven model selection, it can be useful in systematizing the search for the best functional forms when implementing an estimation technique."

纽曼正交和交互拟合构成了DML的两个核心要素。交互拟合保证了第一步的估计误差和第二步的回归残差的独立性。

事实上，第一步中可以考虑用非参数方法，但非参方法存在维度诅咒问题，即随着X的维度增加，观测值变得越来越稀疏，非参估计会越来越不可行。传统的机器学习方法采用正则化避免过度拟合问题，但也导致了估计量的偏差[@Chernozhukov2018]。而双重机器学习则通过对因变量Y和处理变量T分别训练机器学习模型，通过交叉验证得到无偏估计量。

局部线性模型的DML估计步骤如下。

第一步：

- （a）将样本分为K组。令I_k表述第k组的观测值，I_k^c表示其他组的观测值。
- （b）将I_k^c作为训练集，分别对Y和T训练两个机器学习模型，

$$
\begin{aligned}
Y &= l_0(X, W) + V_Y \\
T &= m_0(X, W) + V_T
\end{aligned}
$$

- （c）计算检验集I_k的残差$\hat{V}_{k,T} = T_k - \hat{T}_k$，$\hat{V}_{k,Y} = Y_k - \hat{Y}_k$。其中，$Y_k$和$T_k$表示第$k$组的$Y$和$T$的观测值。
- （d）用$\hat{V}_{k,Y}$对$\hat{V}_{k,T}$回归。

注1：分组个数K并没有统一的标准，模拟研究发现K越大，那么DML表现越好。一般地采用$K = 5$作为基准。

注2：利用机器学习（线性回归、Lasso、随机森林等）的交互拟合得到$E(Y|X,W)$和$E(T|X,W)$的样本外预测，分别表示为$\hat{l}(X,W)$和$\hat{m}(X,W)$，可以选择任意的机器学习方法，不同的机器学习模型存在几个差别。

- 函数形式的弹性
- 选择变量的能力
- 对样本量的要求。

其一，随机森林、提升树、神经网络能够体现非线性关系，而LASSO在本质上是线性的（虽然也可以手动加入变量的平方项或交互项）。GAMs可以拟合多个自变量的平滑函数，但随着维度的增加这一方法面临计算的挑战；而且GAMs仅限于变量（包括交互项）的加式关系。其二，LASSO、随机森林、提升树等方法可以选择重要的变量，因此更适合于高维情形，但GAMs、神经网络则不具有变量选择的能力。其三，随机森林、提升

注3：多数机器学习拟合具有一定的随机性。比如，随机森林通过自助采样（Bootstrap）生成多棵决策树（`ntrees`设置树的数量），并对结果进行平均。其随机性体现在：

- 数据随机性：每棵树使用不同的自助样本（`samprate`设置比例）。
- 特征随机性：每次分裂时随机选择部分特征（`splitmeanvars`设置变量平均个数）。设每次抽取的变量的个数服从泊松分布，均值为μ（`splitmeanvars`）。

注4：DML采用交互拟合（cross fit），对于算力具有一定的要求。

- 在随机森林中，可以采用袋外预测（out-of-bag）以节省计算时间。Out-of-Bag (OOB) 预测是随机森林和基于自助采样（Bootstrap）的集成学习模型中特有的一种验证技术。它利用模型训练时未被选中的数据样本（即 "袋外数据"）进行预测评估，无需额外的验证集或交叉验证，从而节省计算资源并提供无偏的性能估计。
- 诚实随机森林是诚实树的集成。诚实树将所有样本随机分为两组，一组用于分割树（称为分割样本），另外一组用于预测（称为标签样本）。Stata函数`cate`通过`honestrate`选项设置分割样本的比例。

第二步：

- （a）（DML1）依次对$k=(1,2,\cdots,K)$的每一组估计第一步，得到K个估计量，其均值即为$\theta(X)$的估计量$\hat{\theta}(X)$。
- （b）（DML2）依次对$k=(1,2,\cdots,K)$的每一组估计第一步，得到所有观测值的残差$\hat{V}_Y$和$\hat{V}_T$，用$\hat{V}_Y$对$\hat{V}_T$回归，即为$\hat{\theta}(X)$。

第三步：为了降低样本随机划分对估计结果的影响，对第一、二步重复S次，得到S个估计量$(\hat{\theta}_1,\hat{\theta}_2,\cdots,\hat{\theta}_S)$，方差为$(\hat{\sigma}_1^2,\hat{\sigma}_2^2,\cdots,\hat{\sigma}_S^2)$。

$$\tilde{\theta}=\text{median}(\hat{\theta}_1,\hat{\theta}_2,\cdots,\hat{\theta}_S),$$
$$\tilde{\sigma}^2=\text{median}(\hat{\sigma}_s^2+(\hat{\theta}_s-\tilde{\theta})^2)$$

一般的$\theta(X)$:

$$\hat{\theta} = \arg\min_{\theta \in \Theta} E_n \left[(\tilde{Y} - \theta(X) \cdot \tilde{T})^2 \right]$$

`LinearDML`： $\theta(X)$为常数或低维线性函数[@Chernozhukov2016]，称之为线性DML（LinearDML）。比如，

模型：$Y - E(Y|X,W) = \theta(X) \cdot (T - m(X,W)) + U$

$$\theta(X) = \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k$$

$$\text{argmin}_{\beta_1,\ldots,\beta_k} \sum_{i=1}^{N} (\tilde{Y}_i - \beta_1 x_1 \cdot \tilde{T} - \beta_2 x_2 \cdot \tilde{T}, \ldots, -\beta_K x_K \cdot \tilde{T})^2$$

$$\theta(X) = \alpha_0 + \alpha_1 X + \alpha_2 X^2 + \ldots$$
$$\theta(X) = \alpha_0 + \alpha_1 1(id = 1) + \alpha_2 1(id = 2) + \ldots$$
$$\theta(X) = f(X)$$

`SparseLinearDML`：$\theta(X)$为稀疏线性模型 [@Chernozhukov2017; @Chernozhukov2018]，称之为稀疏线性DML（SparseLinearDML）。

`NonParamDML`:由$\tilde{Y}=\theta(X)\tilde{T}+U$，$\frac{\tilde{Y}}{\tilde{T}}=\theta(X)+\frac{U}{\tilde{T}}$。根据矩方程：$E\left(\tilde{T}U\right)=0$，$E\left(\tilde{T}^2\frac{U}{\tilde{T}}\right)=0$，

$$\sum_{i=1}^{N}\tilde{T}_i^2\left(\frac{\tilde{Y}_i}{\tilde{T}_i}-\theta(X)\right)=0$$

即用$\tilde{Y}_i/\tilde{T}_i$对$\theta(X_i)$回归（随机森林、梯度提升等），观测值权重为$\tilde{T}_i^2$。

`CausalForestDML`：因果森林又叫做森林双重机器学习。回顾随机森林的预测$E(y|X_0)$的步骤：

- 种T棵树（为每棵树确定分裂规则）
- 对$X = X_0$，寻找与X_0落在同一片与叶子上的观测值。
- 这些观测值的加权和即为$\hat{y}(X_0)$：

$$\hat{y}(X_0) = \sum_{i=1}^{N} \alpha(X_i, X_0) \cdot y_i$$

其中，观测值i的权重为

$$\alpha(X_i, X_0) = \frac{1}{T} \sum_{t=1}^{T} \frac{\eta_{i,t}(X_0)}{\sum_{i=1}^{N} \eta_{i,t}(X_0)}$$

$0 \leq \alpha(X_i, X_0) \leq 1$，$\sum_{i=1}^{N} \alpha(X_i, X_0) = 1$。

其中，$\sum_{i=1}^{N}\eta_{i,t}(X_0)$为树$t$上与$X_0$在同一个树叶的观测值个数。$\sum_{i=1}^{N}\frac{\eta_{i,t}(X_0)}{\sum_{i=1}^{N}\eta_{i,t}(X_0)}\cdot y_i$即为$X_0$所属叶子上观测值的$y$的均值。$\hat{y}(X_0)$表示将所有树的均值进一步求均值。

$$\hat{y}(X_0)=\sum_{i=1}^{N}\cdot\frac{1}{T}\sum_{t=1}^{T}\frac{\eta_{i,t}(X_0)}{\sum_{i=1}^{N}\eta_{i,t}(X_0)}\cdot y_i$$

实际上，$\hat{y}(X_0)$为下面目标函数的解：

$$Min_\theta\sum_{i=1}^{N}\alpha(X_i,X_0)\cdot(Y_i-\theta)^2$$

一阶条件为：

$$2\cdot\sum_{i=1}^{N}\alpha_i(X_i,X_0)(\tilde{Y}_i-\theta)=0$$

Athey, Tibshirani, and Wager (2019)将随机森林扩展到广义随机森林以估计各种统计量（CATE、条件分位数效应等）：

$$\sum_{i=1}^{n} \alpha_i(X)\psi(\cdot) = 0$$

$\psi()$称为得分函数，不同模型的得分函数不同。在因果森林中，确定权重α_i时也需要把Y_i做相应的替换：

- 在随机森林中，$\psi(\cdot) = Y_i - \theta;\ Y_i^* = Y_i - \hat{\theta}$
- 在因果森林中，$\psi(\cdot) = (\tilde{Y}_i - \theta\tilde{T}_i)\tilde{T}_i;;\ Y_i^* = (\tilde{Y}_i - \hat{\theta}\tilde{T}_i)\tilde{T}_i$
- 在分位数森林中，$\psi(\cdot) = qI(Y_i > \theta) - (1-q)I(Y_i \leq \theta);\ Y_i^* = I(Y_i > \hat{\theta})$

因果森林求解如下方程：

$$\hat{\theta}(X_0) = \text{argmin}_\theta \sum_{i=1}^{N} \alpha_i(X_i, X_0)(\tilde{Y}_i - \theta \cdot \tilde{T}_i)^2$$

其中，$\alpha_i(X_i, X_0)$为观测值i的权重。一阶条件：

$$2 \cdot \sum_{i=1}^{N} \alpha_i(X_i, X_0)(\tilde{Y}_i - \theta \cdot \tilde{T}_i)\tilde{T}_i = 0$$

$$\hat{\theta}(X_0) = \frac{\sum_{i=1}^{N} \alpha_i(X_i, X_0) \cdot \tilde{Y}_i \cdot \tilde{T}_i}{\sum_{i=1}^{N} \alpha_i(X_i, X_0) \cdot \tilde{T}_i^2}$$

$$\alpha_i(X_i, X_0) = \frac{1}{T} \sum_{t=1}^{T} \frac{\eta_{i,t}(X_i, X_0)}{\sum_{i=1}^{N} \eta_{i,t}(X_i, X_0)}$$

其中，如果在树t上，i与X_0同属于一片叶子，$\eta_{i,t}(X_i, X_0) = 1$；否则，取值为零。

`OrthoForest`：正交随机森林（Orthogonal Random Forest，简写为ORF）[@Oprescu2019]:

模型：$Y - E(Y|X,W) = \theta(X) \cdot (T - m(X,W)) + U$，识别$\theta(X)$的矩条件为：

$$E\left[((Y - l(x,W)) - \theta(x)(T - m(x,W))) \cdot (T - m(x,W))|X = x\right] = 0$$

这是 DML 损失的局部版本，等价于在x处局部地最小化$\tilde{Y}$对$\tilde{T}$的平方损失:

$$\sum_{i=1}^{n} \alpha_x(X_i) \cdot (Y_i - \hat{l}_x(X_i,W_i) - \hat{\theta}(x) \cdot (T_i - \hat{m}_x(X_i,W_i)) \cdot (T_i - \hat{m}_x(X_i,W_i)) = 0$$

或者等价地

$$\hat{\theta}(x) = \text{argmin}_\theta \sum_{i=1}^{n} \alpha_x(X_i) \cdot \left(Y_i - \hat{l}_x(X_i,W_i) - \theta \cdot (T_i - \hat{m}_x(X_i,W_i))\right)^2$$

@Friedberg2018 对正交随机森林做了两点改进：

- 设θ为局部线性函数，而不是局部常数，即$\hat{\theta}(x) = \hat{\beta} \cdot x + \hat{\alpha}$
- 正则化目标函数

$$\hat{\alpha}, \hat{\beta} = \text{argmin}_{\alpha,\beta} \sum_{i=1}^{n} \alpha_x(X_i) \Big(Y_i - \hat{q}_x(X_i, W_i) - (\beta \cdot X_i + \alpha)(T_i - \hat{f}_x(X_i, W_i)) \Big)^2 + \lambda \|\beta\|_2^2$$

对每个目标值x，需要估计局部干扰参数$q(X,W) = E(Y|X,W)$和$m(X,W) = E(T|X,W)$。最小化一个加权（惩罚）损失函数（例如平方损失或多项逻辑损失）：

$$\hat{q}_x = \text{argmin}_{q_x \in Q} \sum_{i=1}^{n} \alpha_x(X_i) \cdot loss(Y_i, q_x(X_i, W_i)) + R(q_x)$$

$$\hat{m}_x = \text{argmin}_{m_x \in F} \sum_{i=1}^{n} \alpha_x(X_i) \cdot loss(T_i, f_x(X_i, W_i)) + R(f_x)$$

对离散型的处理变量，基于不同的方程进行估计。设$T \in \{0, 1, \ldots, k\}$，矩方程为

$$E\left[Y_{i,t}^{DR} - Y_{i,0}^{DR} - \theta_t(x)|X = x\right] = 0$$

$$Y_{i,t}^{DR} = E(Y \mid T = t, X_i, W_i) + 1(T_i = t)\frac{Y_i - E(Y \mid T = t, X_i, W_i)}{E[1(T = t) \mid X_i, W_i]}$$

第一阶段，计算$E(Y \mid T = t, X, W)$的局部估计量$g_x(T, X, W)$，
$E[1(T = t) \mid X_i, W_i]$的局部估计量$p_{x,t}(X, W)$。

可以等价地将目标函数可以写为

$$\theta_t(x) = \text{argmin}_{\theta_t} E\left[(Y_{i,t}^{DR} - Y_{i,0}^{DR} - \theta_t)^2|X = x\right]$$

如果$\hat{\theta}_t(x) = \hat{\beta}_t \cdot x + \hat{\alpha}_t$，

$$\hat{\alpha}_t, \hat{\beta}_t = \text{argmin}_{\alpha_t, \beta_t} \sum_{i=1}^{n} K(x, X_i) \cdot \left(Y_{i,t}^{DR} - Y_{i,0}^{DR} - \beta_t \cdot X_i + \alpha_t\right)^2 + \lambda\|\beta_t\|_2^2$$

双重机器学习（DML）的主要优势在于：

DML具有双重稳健特征，即只要因变量Y和处理变量T有一个模型的设定是正确的，那么估计量即是一致的。DML的稳健性源于以下事实：对应于最终最小二乘估计的矩方程（即平方损失的梯度）满足关于干扰参数$l(X,W)$ 和$m(X,W)$Neyman 正交条件。

双重机器学习不对 $l(X,W)$（结果变量的回归模型） 和 $m(X,W)$（倾向得分模型）做进一步的结构性假设，而是使用任意非参数的机器学习方法对它们进行非参数估计。

如果对干扰参数模型（结果回归模型或倾向得分模型）做出参数化假设，即使第一阶段对$E(Y|X,W)$和$E(T|X,W)$的估计仅在均方根误差（RMSE）意义下达到 $n^{-1/4}$阶的一致性，该方法仍能实现快速估计速率，并且对于许多第二阶段估计量，其最终估计 $\theta(X)$ 也满足渐近正态性。为使该定理成立，干扰参数估计需要以交叉拟合的方式进行。

$$Y_{i,t} = \beta D_{i,(t-1)} + g(Z_{i,t}) + \epsilon_{i,t}.$$

$$Y_{j,t} = \beta D_{j,(t-1)} + g\left(X_{j,t}, \bar{X}_j, \bar{X}_t, X_{j,0}, Y_{j,0}, t\right) + \epsilon_{j,t}$$

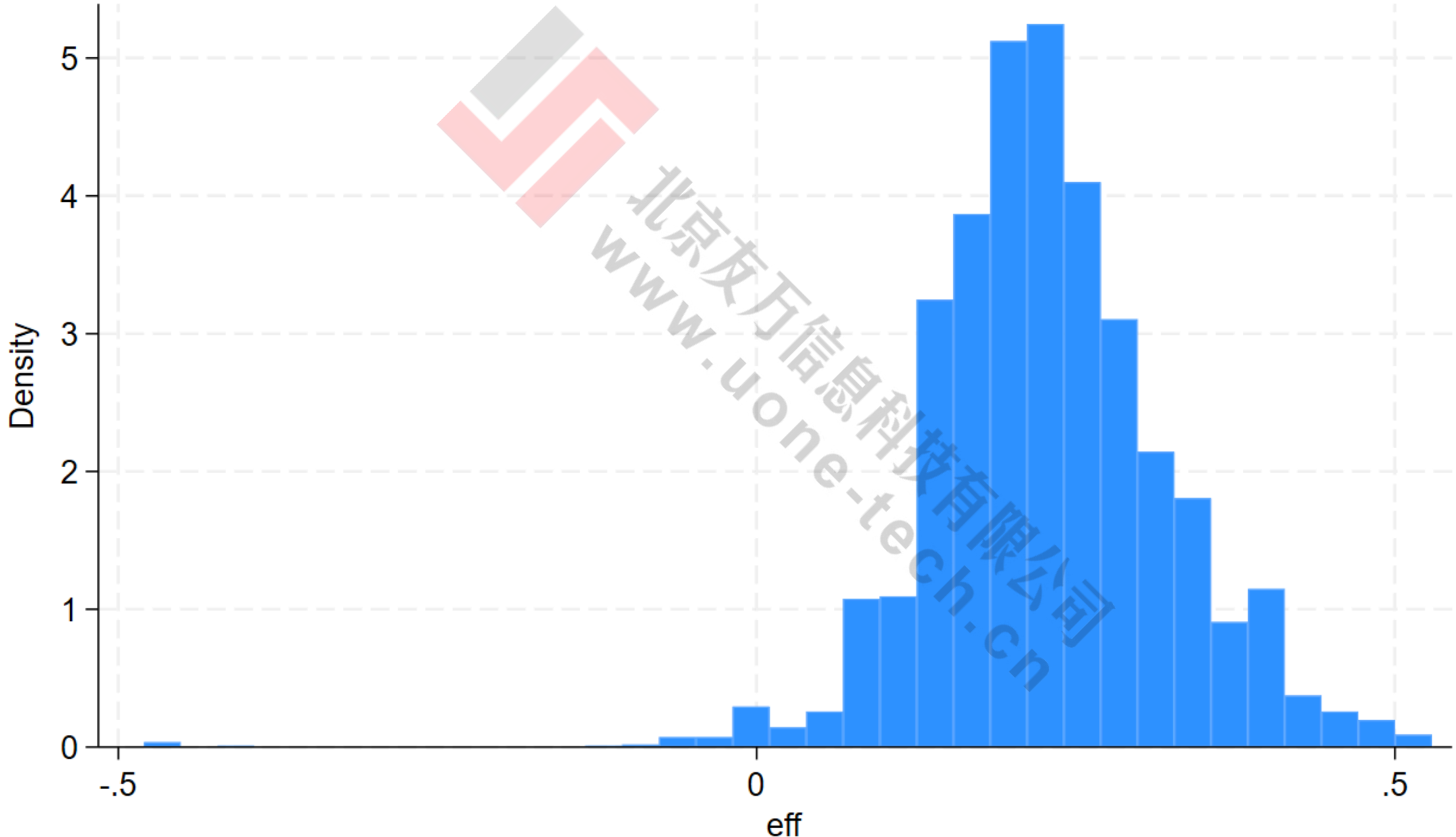
$Y_{j,t}$ is the log homicide rate in county j at time t, $D_{j,t-1}$ is the log fraction of suicides committed with a firearm in county j at time $t-1$, which we use as a proxy for gun ownership, and $Z_{j,t}$ is a set of demographic and economic characteristics of county j at time t. Assuming the firearm suicide rate is a good proxy for gun ownership, the parameter β is the effect of gun ownership on homicide rates, controlling for county-level demographic and economic characteristics.

The sample covers 195 large United States counties between the years 1980 through 1999, 3900 observations.

```
. use "gunclean.dta", clear
. order logghomr logfssl newblack logrobr
. econmlcate logghomr newblack logrobr, treat(logfssl) wlist(timeind-VST020DT) ///
modely("lassocv") modelt("lassocv") learner("lineardml")

Learner    =             LinearDML        No. of obs.   =         3900
E(Y|X,W)   =               lassocv        No. of folds  =            5
E(T|X,W)   =               lassocv        No. of rep    =          100
Score      =                0.1732
------------------------------------------------------------------------------
             |                   OLS
    logghomr | Coefficient  std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
     logfssl |   .2289695   .0578829     3.96   0.000     .115521     .342418
------------------------------------------------------------------------------


------------------------------------------------------------------------------
    theta(X) |      coef  se(clas~c)          t          p      lower      upper
-------------+----------------------------------------------------------------
    newblack | -.1120391   .0365624  -3.064327   .0021816  -.1837014  -.0403768
     logrobr |  .0265965   .0145435   1.828749   .0674372  -.0019088   .0551018
------------------------------------------------------------------------------
```



```
. use "gunclean.dta", clear
. skcrossfit logghomr newblack logrobr  timeind-VST020DT, model("lassocv") kwargs(random_state=123)
. skcrossfit logfssl newblack logrobr  timeind-VST020DT, model("lassocv") kwargs(random_state=123)
. gen resy = logghomr-_logghomrpred
. gen rest= logfssl - _logfsslpred
. reg resy c.rest c.rest#c.( logburg newblack logrobr newfhh newmove newdens newmal)
```

`econmlcate`(EconML）和`dmlcate`（DoubleML）处理二值变量、多水平干预变量和连续型干预变量。

`modely`和`modelt`：

- `linear lasso lassocv lassolars lassolarscv ridge ridgecv bayesianridge elasticnet elasticnetcv lars larscv logit logitcv poisson dtc dtr svc svr knc knr mlpc mlpr rfc rfr bagc bagr gbc gbr`
- https://scikit-learn.org/stable/api/index.html

`infer`：

- `infer('bootstrap')`，`infer(BootstrapInference(n_bootstrap_samples=100)`
- `infer('statsmodels')`，`infer(StatsModelsInferenceDiscrete(cov_type='HC1')`

Learner	Inference	options
All	`"bootstrap"`，`BootstrapInference(...)`	`n_bootstrap_samples=100`
`LinearDRLearner`	`"statsmodels"`，`StatsModelsInference(...)`，`StatsModelsInferenceDiscrete(...)`	`cov_type='HC1'`
`SparseLinearDML`	`'auto'`，`'debiasedlasso'`	
`SparseDRLearner`	`'auto'`，`'debiasedlasso'`	
`CausalForestDML`	`'auto'`，`'blb'`	

```
. estat shap
```

主要内容：

- 机器学习在经济学中的应用概述
- 局部线性模型
- **交互模型**
- 机器学习因果推断（DML-IV、DML-RD、DML-DID、DML-SS）
- 政策学习

交互模型的双重稳健估计

设处理水平$t \in \{1, 2, \cdots, n_t\}$。定义$Y^{(t)}$为处理水平$t$对应的潜在结果。

双重稳健方法（Doubly Robust，简写为DRL）估计两个学习模型：

- 对结果变量建立模型：$y = f(T, X, W) + U$
- 对干预变量建立模型：$T = m(X, W) + V$

控制变量为(X, W)，CATE为X的函数。

$$Y^{(t)} = g_t(X, W) + U_t \quad E(U|X, W) = 0$$
$$P[T = t|X, W] = p_t(X, W)$$

假定$Y^{(t)} \perp T|X, W$。CATE为

$$\theta_t(X) = E[Y^{(t)} - Y^{(0)}|X] = E[g_t(X, W) - g_0(X, W)|X]$$

当$t=(0,1)$时,

$$
\begin{aligned}
y(1) &= g_1(X,W)+u_1 \\
y(0) &= g_0(X,W)+u_0 \\
P(T=1) &= m(X,W)
\end{aligned}
$$

IATE函数为

$$\theta(X)=E[y(1)-y(0)|X]=E[g_1(X,W)-g_0(X,W)|X]$$

一种方法是，先学习模型$g_T(X,W)=E[Y|T,X,W]$，然后用

$$Y_{i,t}=g_t(X_i,W_i)-g_0(X_i,W_i)$$

对X回归，即得到$\theta(X)$的估计。

这种方法的主要问题在于它严重依赖于拟合的模型。从本质上讲，当我们对特征为(X,W)、接受其他处理t' 的样本评估 $g_t(X,W)$ 时，我们实际上是从其他具有相似(X,W)的接受处理 t 的样本中进行外推。然而，“相似性”的定义高度依赖模型，在某些情况下，我们甚至可能从非常远的点进行外推。

逆概率加权法（IPW）：令

$$Y_{i,t}^{IPS} = \frac{Y_i 1(T_i = t)}{P(T_i = t|X_i, W_i)} = \frac{Y_i 1(T_i = t)}{p_t(X_i, W_i)}.$$

$$E[Y_{i,t}^{IPS}|X, W] = E\left[\frac{Y_i 1(T_i = t)}{p_t(X_i, W_i)}|X_i, W_i\right] = E\left[\frac{Y_i^{(t)} 1(T_i = t)}{p_t(X_i, W_i)}|X_i, W_i\right]$$

$$= E\left[\frac{Y_i^{(t)} E[1(T_i = t)|X_i, W_i]}{p_t(X_i, W_i)}|X_i, W_i\right] = E\left[Y_i^{(t)}|X_i, W_i\right]$$

用$Y_{i,t}^{IPS} - Y_{i,0}^{IPS}$对$X$回归即得到$\theta(X)$的估计。

IPW方法有两个缺点。

- 首先，由于用相对较小的数值去除观测值（尤其是在某些区间，干预发生的可能性极低时），因此这种方法存在高方差问题。
- 其次，在观测数据中，我们通常并不知道干预的概率，因此需要估计干预概率的模型。这相当于一项多类别分类任务，当数据(X, W)是高维的，或者当使用随机森林等非线性模型时，估计速度可能会很慢。
- 而且，当使用机器学习来拟合这些倾向得分模型，那么很难刻画估计值的极限分布，从而难以提供有效的置信区间。

双重稳健（DR）方法拟合了一个直接回归模型，然后通过对该模型的残差应用逆倾向得分方法来消除该模型的偏差，即它构建了以下潜在结果的估计值：

$$Y_{i,t}^{DR} = E[Y|X_i, W_i, T_i = t] + \frac{Y_i - E[Y|X_i, W_i, T_i = t]}{P[T_i = t|X_i, W_i]} \cdot 1(T_i = t)$$

IATE函数为

$$\theta_t(X_i) = E\left[Y_{i,t}^{DR} - Y_{i,0}^{DR}|X_i\right]$$

第一阶段：估计$g(X, W, T) = E[Y|X, W, T]$和$p_t(X, W) = P[T = t|X, W]$。

- 该阶段的任务属于纯预测，因此任何适用于预测的机器学习方法都是可行的。
- 该阶段中机器学习中的超参数在程序包中大多可以通过交叉验证的方法自动确定。
- 与DML不同，DR的结果方程的解释变量包括干预变量和控制变量，而DML只包括控制变量。

第二阶段：对于每一个t，$Y_{i,t}^{DR} - Y_{i,0}^{DR}$对$X_i$回归。

对于GATE的AIPW估计，用$Y_{i,t}^{DR}$对分组虚拟变量做OLS回归即得到每一组的平均处理效应。如果没有指定的分组变量，那么按照IATE的估计值的分位数进行分组，再对分组虚拟变量做OLS回归即可。

对于$t = \{0, 1\}$：首先，估计如下模型：

$$p(x) = E[D \mid X = x] = \Pr(D = 1 \mid X = x), \quad (\mathrm{E(T|X=x)})$$
$$g_0(x) = E[Y \mid D = 0, X = x], \quad (\mathrm{E(y=0)})$$
$$g_1(x) = E[Y \mid D = 1, X = x], \quad (\mathrm{E(y=1)})$$

利用交互拟合得到各个方程的交互拟合值和残差，$\hat{p}(X_i)$，$\hat{g}_0(X_i)$，$\hat{g}_1(X_i)$，定义

$$\hat{g}_{D_i}(X_i) = \hat{g}_0(X_i)(1 - D_i) + \hat{g}_1(X_i)D_i$$

然后，计算Y_i^{DR}:

$$Y_i^{DR}(\hat{g},\hat{p}) = \hat{g}_1(X_i) - \hat{g}_0(X_i) + (Y_i - \hat{g}_{D_i}(X_i))\frac{D_i - \hat{p}(X_i)}{\hat{p}(X_i)(1-\hat{p}(X_i))}$$

$$ATE = E\left[Y^{DR}(\hat{g},\hat{p})\right]$$

为了保证数值稳定性，上式求分母的协方差时一般要进行截断处理：

$$Y_i^{DR}(\hat{g},\hat{p}) := \hat{g}_1(X_i) - \hat{g}_0(X_i) + (Y_i - \hat{g}_{D_i}(X_i))\frac{D_i - \hat{p}(X_i)}{\min\{c, \hat{p}(X_i)(1-\hat{p}(X_i))\}}$$

`LinearDRLearner`：用$Y_{i,t}^{DR} - Y_{i,0}^{DR}$对$X_i$做线性回归。

```
. econmlcate $y $xlist , treat(p401) wlist($wlist) tdisc(True) learner("lineardrl")

Learner    =       LinearDRLearner         No. of obs.   =          9717
E(Y|X,W)   =                 lasso         No. of folds  =             5
E(T|X,W)   =                 logit         No. of rep    =           100
Score      =       4633494282.7408
------------------------------------------------------------------------------
     net_tfa | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
        p401 |   11293.52    690.704    16.35   0.000     9939.764    12647.27
------------------------------------------------------------------------------


------------------------------------------------------------------------------
    theta(X) |      coef         se          t        p      lower      upper
-------------+----------------------------------------------------------------
         age |   321.235   81.07862   3.962018   .0000743   162.3209   480.1491
         inc |  .2722905   .0618516   4.402318   .0000107   .1510613   .3935197
        educ | -422.4962    289.378  -1.460015    .144286   -989.677   144.6847
       hown |  1649.144   1462.545   1.127585   .2594952  -1217.445   4515.733
------------------------------------------------------------------------------
```

`SparseLinearDRLearner`：用$Y_{i,t}^{DR}-Y_{i,0}^{DR}$对$X_i$做Lasso线性回归。

`ForestDRLearner`：用$Y_{i,t}^{DR}-Y_{i,0}^{DR}$做森林双重稳健学习（Forest Doubly Robust Learner） [@Wager2018, @Athey2019, @Oprescu2019]，是广义随机森林和正交随机森林的变形。设局部回归模型为

$$\theta_t(x)=\text{argmin}_{\theta_t}\sum_{i=1}^{n}K(x,X_i)\cdot\left(Y_{i,t}^{DR}-Y_{i,0}^{DR}-\theta_t\right)^2$$

其中，

$$Y_{i,t}^{DR}=\hat{g}(t,X_i,W_i)+1(T_i=t)\frac{Y_i-\hat{g}(t,X_i,W_i)}{\hat{p}_t(X_i,W_i)]}$$

其中，$\hat{g}(t,X,W)$为$E(Y|T=t,X,W)$的估计量，$\hat{p}_t(X,W)$为$P(T=t|X,W)$估计量。

双重稳健方法的主要优势在于，最终估计量 $\theta(X)$ 的均方误差仅受回归估计 $\hat{g}(X,W)$ 和倾向得分估计 $\hat{p}(X,W)$ 的均方误差乘积的影响。因此，只要其中一个估计准确，最终模型就是正确的。例如，只要两者的收敛速度均不慢于 $n^{-1/4}$，则最终模型可达到参数化的 $n^{-1/2}$ 收敛速度。

在最终阶段的估计量渐近正态，从而可以构建有效的置信区间。干扰参数估计必须以交叉拟合的方式进行。

DR方法相对于DML方法的一个优势在于，即使最终回归所优化的函数空间不包含真实的条件平均处理效应（CATE）函数，最终回归仍然具有意义。在这种情况下，该方法会估计 CATE 函数在最终回归优化的模型空间上的投影。例如，这允许对 CATE 函数的最佳线性投影进行推断，或对可能导致异质性的特征子集上的最佳 CATE 函数进行推断。例如，可以将 DR 方法与诚实森林等非参数最终模型结合使用，在不对处理效应异质性的具体形式做进一步假设的情况下，推断单个特征的边际处理效应异质性。DR 方法相对于 DML 的缺点是它通常具有更高的方差，特别是当(X,W)的某些区间的处理分配概率很低的情况下（文献中通常称为 “小重叠”）。在这种设置下，DML 方法可能具有更好的外

```
. use dmlirm, clear
. econmlcate y X1-X2, treat(d) wlist(X3-X10) tdisc(True) modelt("logit") learner("lineardrl")

Learner    =       LinearDRLearner        No. of obs.   =          500
E(Y|X,W)   =                 lasso        No. of folds  =            5
E(T|X,W)   =                 logit        No. of rep    =          100
Score      =               17.8083
------------------------------------------------------------------------------
           y | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
           d |   .6075278   .1892891     3.21   0.001      .236528    .9785276
------------------------------------------------------------------------------

------------------------------------------------------------------------------
    theta(X) |      coef         se          t         p      lower      upper
-------------+----------------------------------------------------------------
          X1 |  .9632839   .3048822   3.159528   .0015802   .3657148   1.560853
          X2 |  .0214181   .1831132   .1169665   .9068866  -.3374838    .38032
------------------------------------------------------------------------------
```

```
. econmlcate y X1-X2, treat(d) wlist(X3-X10) tdisc(True) learner('cfdml')

Learner    =                 CausalForestDML      No. of obs.   =          500
E(Y|X,W)   =                        "forest"      No. of folds  =            5
E(T|X,W)   =                        "forest"      No. of rep    =          100
Score      =                          1.1422
------------------------------------------------------------------------------
           y | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
           d |       .532       .374     1.42   0.155    -.2010265    1.265027
------------------------------------------------------------------------------
```

CATE模型的检验

CATE 模型的验证测试包括： Chernozhukov etc.(2022)提出的最佳线性预测器（BLP）测试、Dwivedi etc. (2020)的校准测试，以及 Radcliffe(2007)的 QINI 系数。

BLP检验：对 CATE 模型的双重稳健 (DR) 结果预测值（及一个常数项）进行双重稳健结果的普通最小二乘法 (OLS) 回归。如果 CATE 模型捕捉到了真实的异质性，那么针对 CATE 预测值的 OLS 估计系数应显著为正且与 0 存在显著差异。

第1步：使用 CATE 模型对样本外数据（或未参与模型训练的数据）生成因果效应预测值 $\hat{\theta}(X)$，其中 X 代表个体特征。

$$Y_t^{DR} = Y\frac{T}{\hat{P}(X)} + \hat{Y}_0(X)\left(1 - \frac{T}{\hat{P}(X)}\right).$$

其中，Y为因变量观测值，T为干预变量，$\hat{P}(X)$为倾向得分。$\hat{Y}_t(X)$为t组的结果预测值。

第2步：回归模型$Y^{DR} = \gamma_0 + \gamma_1 \hat{\theta}(X) + \epsilon$。如果$\gamma_1 > 0$，则表明CATE 模型有效捕捉了真实因果异质性。

Chernozhukov et al.（2022）将BLP 检验称为 CATE 模型评价的 “黄金标准” 之一，尤其适用于处理效应存在线性异质性的场景。不过BLP方法仅验证线性相关性，无法捕捉复杂的非线性异质性模式。

校准检验：首先，基于 CATE 预测值$\theta(X)$的样本外分位数对单元进行分箱。在每个分箱（k）中，计算 CATE 预测均值与双重稳健（DR）结果之间的绝对差，以及 CATE 预测均值与总体平均处理效应（ATE）之间的绝对差。然后，将这些指标在所有分箱上求和，并以单元出现在每个分箱中的概率为权重。

$$\mathrm{Cal}_G = \sum_k \pi(k)|E[\theta(X)|k] - E[Y^{DR}|k]|$$

$$\mathrm{Cal}_O = \sum_k \pi(k)|E[\theta(X)|k] - E[Y^{DR}]|$$

校准可决系数定义为

$$\mathcal{R}^2{}_C = 1 - \frac{\mathrm{Cal}_G}{\mathrm{Cal}_O}$$

首先，根据预测的 CATE 值对单元进行排序，并在遍历排序时持续记录每个队列的平均处理效应。由此产生的 TOC 曲线可用于绘制图表，计算其积分并将其作为 CATE 模型捕捉到的真实异质性的度量；该积分被称为 AUTOC（TOC 曲线下面积）。QINI 曲线是此曲线的一个变体，它还纳入了处理概率；其积分被称为 QINI 系数。

$$\tau_{TOC}(q) = \mathrm{Cov}\left(Y^{DR}(g,p), \frac{\mathbb{1}\left\{\hat{\theta}(X) \geq \hat{\mu}(q)\right\}}{\Pr(\hat{\theta}(X) \geq \hat{\mu}(q))}\right)$$

$$\tau_{QINI}(q) = \mathrm{Cov}\left(Y^{DR}(g,p), \mathbb{1}\left\{\hat{\theta}(X) \geq \hat{\mu}(q)\right\}\right)$$

其中，$\hat{\mu}(q)$为q分位数。$Y^{DR}(g,p)$为DR差异。

AUTOC和QINI系数定义为：

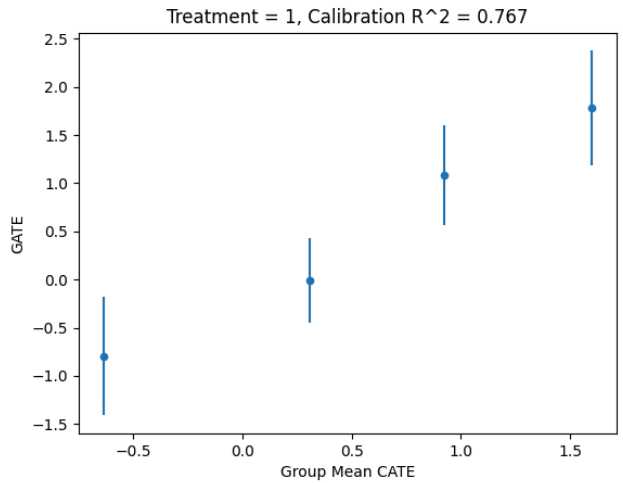
$$AUTOC = \int_0^1 \tau_{TOC}(q)dq$$

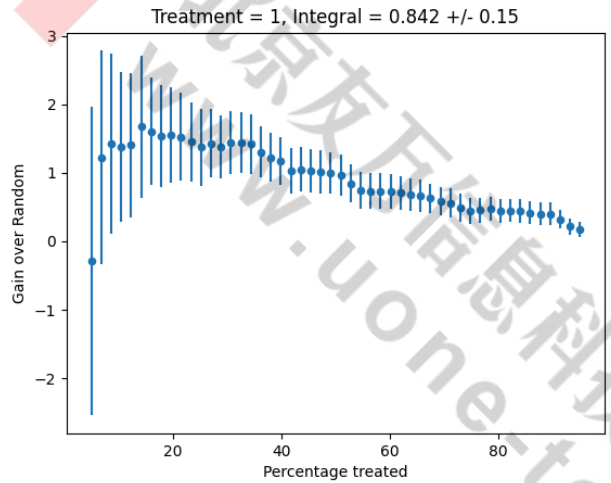
$$QINI = \int_0^1 \tau_{QINI}(q)dq$$

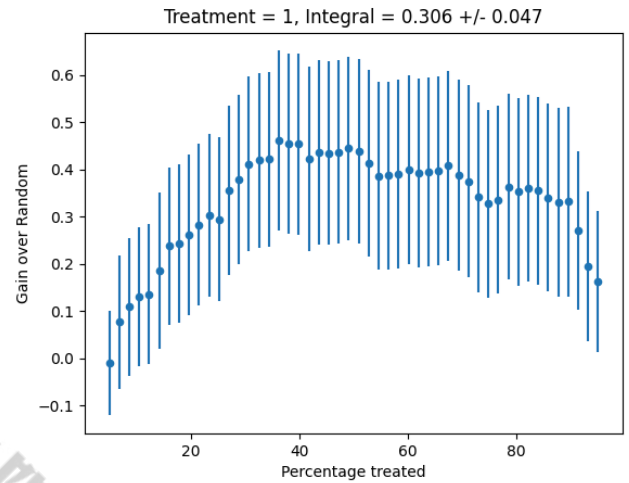
```
. estat valid

CATE model test:

------------------------------------------------------
             |       Stat          Se          Prob
-------------+----------------------------------------
         BLP |     1.6040      0.1470        0.0000
        Qini |  2396.3980    265.2170        0.0000
       Autoc |  7247.8770    896.1090        0.0000
------------------------------------------------------
```







DML-IV

正交工具变量回归[@Syrgkanis2019]：`OrthoIV` 矩方程为：

$$E[(Y - E[Y|X,W] - \theta(X) * (E(T|X,W,Z) - E[T|X,W]))(Z - E[Z|X,W])] = 0$$

`DMLIV` 最小化如下目标函数：

$$\sum_i (Y_i - E[Y|X_i,W_i] - \theta(X) * (E[T|X_i,W_i,Z_i] - E[T|X_i,W_i]))^2$$

- `modely()`：$E[Y|X_i,W_i]$
- `modelt()`：$E[T|X_i,W_i]$
- `modelt2()`：$E[T|X_i,Z_i,W_i]$
- `modelf()`：$\theta(X)$

`econmliv y xlist, treat() wlist() zlist() learner() ...`

`learner()`：`OrthoIV, DMLIV, NonParamDMLIV`

```
. use dmliv, clear
. econmliv y X1 X2, treat(d) zlist(z) wlist(X6-X10) ydisc(False) tdisc(True) learner("orthoiv")

Learner       =                 orthoiv        No. of obs.     =         1000
E(Y|X,W)      =                  "auto"        No. of folds    =            5
E(T|X,W)      =                  "auto"        No. of rep      =            1
E(T|X,W,Z)    =                  "auto"        Agg.            =         mean
E(Z|X,W)      =                  "auto"        Score           =       0.0000
------------------------------------------------------------------------------
           y | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
           y |       .449         .2     2.25   0.025     .0570072    .8409928
------------------------------------------------------------------------------
```

```
. svmat e(effect), names(_eff)
. list _eff1-_eff6 in 1/5

     +-------------------------------------------------------------------+
     |    _eff1      _eff2       _eff3      _eff4       _eff5      _eff6 |
     |-------------------------------------------------------------------|
  1. | .2444565   .4411006    .5541967   .5794442   -.6201006   1.109014 |
  2. | .7066314   .2472946    2.857447   .0042706    .2219339   1.191329 |
  3. | .4931455   .2222287     2.21909   .0264806    .0575773   .9287137 |
  4. | .0601316   .2886956    .2082873   .8350047   -.5057117    .625975 |
  5. | -.047791   .4341471   -.1100801   .9123459   -.8987193   .8031374 |
     +-------------------------------------------------------------------+
```

双重稳健IV估计：内生干预变量T和工具变量Z都为0-1变量。

首先将样本分为三组：$\{Y^i, X^i, W^i, Z^i\}, i = \{1, 2, 3\}$。

第1步：把样本分为三组$\{Y^i, X^i, W^i, Z^i\}, i = \{1, 2, 3\}$。用$\{X^1, W^1, Z^1\}$分别估计控制组和处理组的倾向得分$\hat{p}_0(x)$、$\hat{p}_1(x)$，用$\{Y^2, X^2, W^2, Z^2\}$分别拟合控制组和处理组的结果变量$\hat{g}_0(x)$和$\hat{g}_1(x)$。估计$Z$的分配概率$p_Z$。

第2步：用$\{Y^3, X^3, W^3\}$最小化如下损失函数

$$L(\hat{\theta}(X)) = \hat{E}\Bigg[\Bigg(\hat{g}_1(X) - \hat{g}_0(X) + \frac{Z(Y - \hat{g}_1(X))}{p_Z} - \frac{(1-Z)(Y - \hat{g}_0(X))}{1 - p_Z} - \Big(\hat{p}_1(X) - \hat{p}_0(X) + \frac{Z(T - \hat{p}_1(X))}{p_Z} - \frac{(1-Z)(T - \hat{p}_0(X))}{1 - p_Z}\Big)\hat{\theta}(X)\Bigg)^2\Bigg]$$

第3步：与双重稳健估计相类似，重复第1、2步，最终的CATE估计量为三个CATE估计量的均值。

`learner()`：`LinearDRIV, SparseLinearDRIV, ForestDRIV, IntentToTreatDRIV, LinearIntentToTreatDRIV`

```
. use dmliv, clear
. econmliv y X1 X2, treat(d) zlist(z) wlist(X6-X10) ydisc(False) tdisc(True) learner("lineardriv")
[est.joblib]

Learner         =        lineardriv      No. of obs.     =1000
E(Y|X,W)        =    "auto"      No. of folds    =    .
E(T|X,W)        =    "auto"      No. of rep      =    .
E(T|X,W,Z)      =    "auto"      Agg.    =
E(Z|X,W)        =    "auto"      Score   =      38.8065
------------------------------------------------------------------------------
  y | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
  y |       .429       .197     2.18   0.029     .0428871    .8151129
------------------------------------------------------------------------------

. matlist e(coef)

    |      coefse t p      lower      upper
-------------+------------------------------------------------------------
 X1 | -.1267866  -.0359845   3.523372   .0004261  -.5751634  -.7162224
 X2 |  -.300404  -.0525958   5.711554   1.12e-08   .3215901   .1154144
```

深度工具变量（deep instrumental variables） [@Hartford2017]

$$Y = g(T, X, W) + U$$

$E[U|X, W, Z] = h(X, W)$，T的分布$F(T|X, W, Z)$为Z的函数。异质处理效应为

$$\theta(t_0, t_1, X) = E[g(t_1, X, W) - g(t_0, X, W)]$$

深度IV通过神经网络估计g，SIEVE方法假定g为基函数的加权和。

两阶段基础扩展方法（two-stage basis expansion approach）[Newey2003]。

主要内容：

- 机器学习在经济学中的应用概述
- 局部线性模型
- 交互模型
- **机器学习因果推断**
- 政策学习

机器学习因果推断

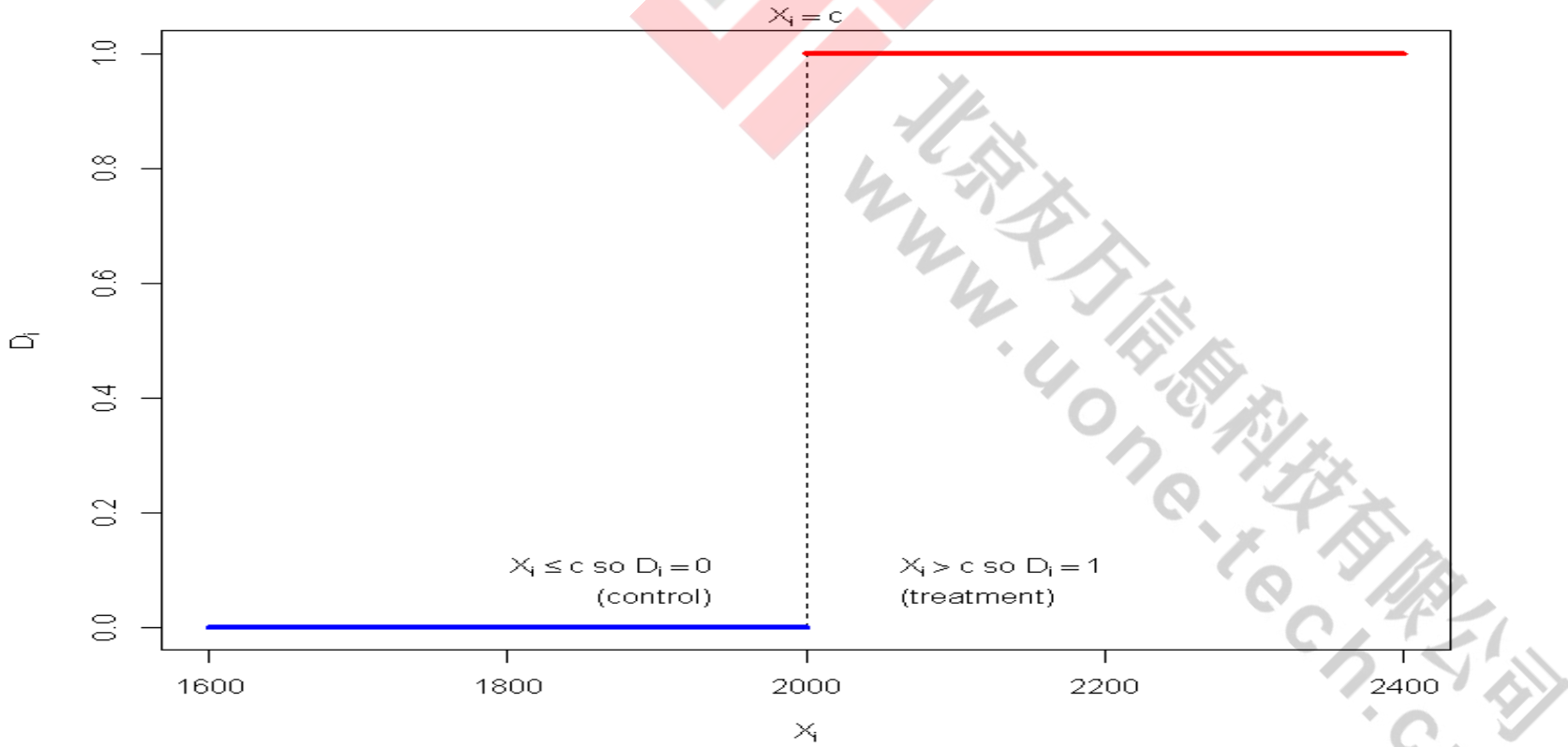
DML-IV, DML-DID, DML-RDD, DML-SS

断点设计包括两类：精确断点设计和模糊断点设计。

- 如果干预变量是某个驱动变量的示性函数，即精确断点设计。
- 如果干预变量是某个驱动变量的概率函数，称为模糊断点设计。

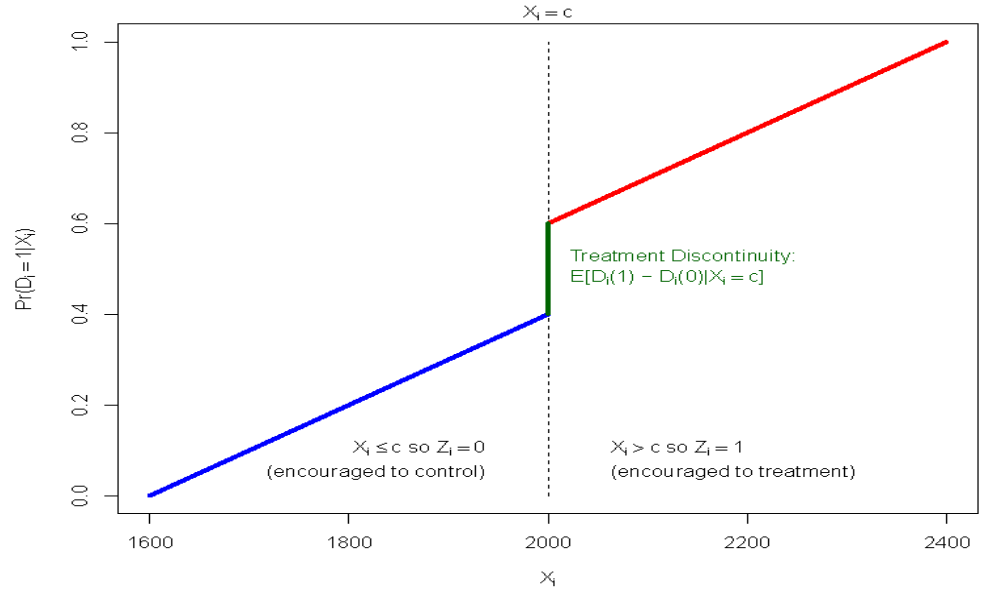
早在1960年由Thistlethwate and Campbell提出。近些年在因果推断的框架下得到了被重新认识和发展。

精确断点设计（sharp RD）：$\mathrm{P}(D=1|x\geq c)=1,\mathrm{P}(D=1|x<c)=0.$



模糊断点设计（fuzzy RD）：$0 < \mathrm{P}(D = 1|x \geq c) < 1, 0 < \mathrm{P}(D = 1|x < c) < 1.$

在模糊断点设计中，处理分配仍然是确定性的。然而，在断点附近，存在不依从的概率。因此，某些个体实际接受的处理可能与分配的处理不同。这些情况导致干预概率在断点处的跳跃幅度小于 1 但大于 0。换句话说，在断点附近，两侧都可能存在处理随机化。



干预变量为$D_i = \mathbb{I}[S_i \geq c]$。平均处理效应为

$$\tau = \mathbb{E}[Y_i(1) - Y_i(0) \mid S_i = c]$$

$$\mathrm{E}(y|x) = \mathrm{E}(y_0|x) + (\mathrm{E}(y_1 - y_0|x))D = (1-D)\mathrm{E}(y_0|x) + D\mathrm{E}(y_1|x)$$

设

$$\mathrm{E}(y_0|x) = f_0(x) = \alpha_0 + \alpha_1(x-c) + \cdots + \alpha_p(x-c)^p$$
$$\mathrm{E}(y_1|x) = f_1(x) = \beta_0 + \beta_1(x-c) + \cdots + \beta_p(x-c)^p$$

$$\tau = \mathrm{E}(y_1|c) - \mathrm{E}(y_0|c) = \beta_0 - \alpha_0.$$

带入即可得到一般的估计式。经常约束$D=0$和$D=1$的部分参数相同。

估计时，分组回归也可以采用交互项的形式。比如，$y = \beta_0 + \delta D + \beta_1 (x - c) + u$

左右两侧存在不同的斜率：

$$
\begin{aligned}
y &= \alpha_0 + \delta D + \alpha_1 (1 - D)(x - c) + \beta_1 D (x - c) + u \\
&= \begin{cases} \alpha_0 + \alpha_1 (x - c) + u, & D = 0 \\ \alpha_0 + \delta + \beta_1 (x - c) + u, & D = 1 \end{cases}
\end{aligned}
$$

左右两侧不同的多项式模型：

$$
\begin{aligned}
y &= \alpha_0 + \delta D + (1 - D) \times f_0 (x - c) + D \times f_1 (x - c) + u \\
&= \begin{cases} \alpha_0 + f_0 (x - c) + u, & D = 0 \\ \alpha_0 + \delta + f_1 (x - c) + u, & D = 1 \end{cases}
\end{aligned}
$$

无条件估计量（非参估计）：

$$\hat{\tau}_{\text{base}}(h) = \sum_{i=1}^{n} w_i(h) Y_i,$$

其中，$w_i(h)$为由驱动变量决定的局部线性回归权重，h为窗宽。

$$\hat{\tau}_{\text{base}}(h) \overset{a}{\sim} N\left(\tau + h^2 B_{\text{base}}, (nh)^{-1} V_{\text{base}}\right).$$

线性调整估计量：

$$\hat{\tau}_{lin}(h) = \sum_{i=1}^{n} w_i(h)(Y_i - X_i^T \hat{\gamma}_h)$$

其中，$\hat{\gamma}_h$为最小化如下目标函数的估计量

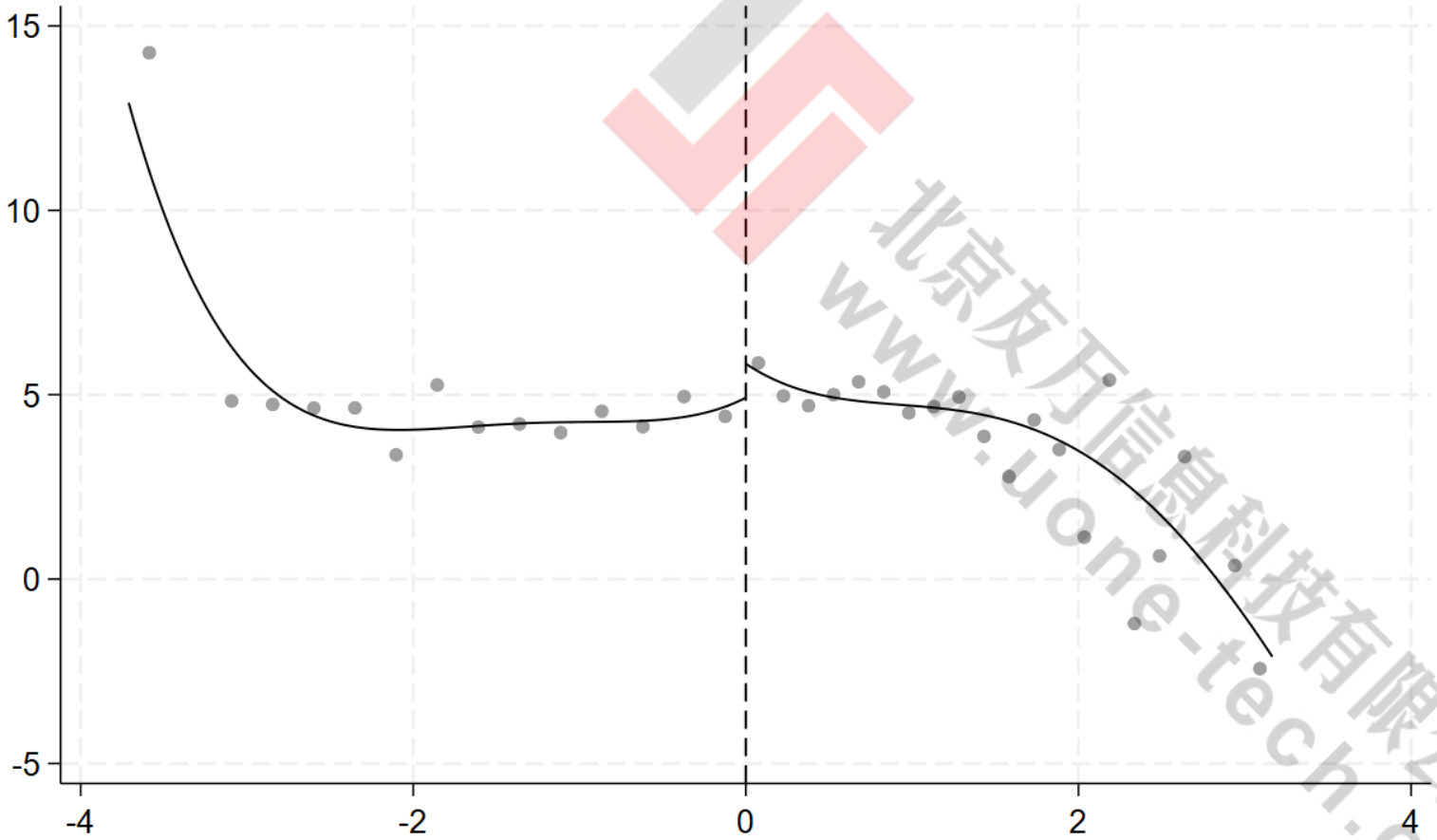
$$\arg\min_{\beta,\gamma} \sum K_h(S_i)(Y_i - Q_i^\top \beta - X_i^\top \gamma)^2.$$

其中，$Q_i = (D_i, S_i, D_i S_i, 1)^T$。

非线性调整估计量（DML）：

$$\hat{\tau}_{\mathrm{RDFlex}}(h;\eta) = \sum_{i=1}^{n} w_i(h) M_i(\eta), \quad M_i(\eta) = Y_i - \eta(X_i),$$

其中，$\eta(\cdot)$为非线性调整函数。



```
doublemlrd varlist, treat(varname) score(varname) cutoff(0)  ///
         method(robust|dml) fuzzy(False|True) modely(regression) modelt(classifier)  ///
         kernel(string) nfolds(5) nrep(1) level(95) *
```

`method`：`robust` or `dml`. default= `robust`

`fuzzy`: defualt= `False`, only for DML-RD. For `method(robust)`, the program will judge automatically.

`kernel`：`uniform`, `triangular`, `epanechnikov`

`modely`: default = `LassoCV()`, only for DML-RD

`modelt`: default = `LogisticRegressionCV()`, only for DML-RD

```
. use dmlrd, clear
. summ

    Variable |        Obs        Mean    Std. dev.       Min        Max
-------------+---------------------------------------------------------
           y |      1,000    4.579999    4.165335  -2.868718   24.26434
           d |      1,000        .513    .5000811          0          1
       score |      1,000   -.0116202    1.026664  -3.710679   3.166557
          x0 |      1,000   -.0197126    .5765338  -.9984912    .999351
          x1 |      1,000   -.0406238     .571024  -.9990833   .9981243
-------------+---------------------------------------------------------
          x2 |      1,000   -.0069902    .5747975  -.9999586   .9989593
```

```
 . doublemlrd y x0 x1 x2, treat(d) score(score) method(robust)

Method          =          robust                No. of obs.    =         1000
Obs(left)       =             490                Effobs(left)   =          244
Obs(right)      =             510                Effobs(right)  =          263
Bandwidth(left) =          0.6530                Bias(left)     =       1.0232
Bandwidth(right)=          0.6530                Bias(right)    =       1.0232
BW selection    =           mserd                Kernel         =   triangular
Order           =               1                VCE            =           NN
Order(Bias)     =               2

------------------------------------------------------------------------------
        LATE |      coef         se      tstat      pvale      cilow      ciupp
-------------+----------------------------------------------------------------
Conventional |  2.797709   3.979983   .7029448   .4820901  -.8011955   1.547919
Biascorrec~d |  3.347862   3.979983    .841175   .4002499  -.7291005   1.620014
      Robust |  3.347862   4.670639    .716789   .4735043   -.941819   1.832733
------------------------------------------------------------------------------
```

DML-SRD

$$\mathrm{P}(D=1|x)=F(x)$$

由$y=y_0+D(y_1-y_0)$，

$$\begin{aligned}\mathrm{E}(y|x)&=\mathrm{E}(y_0|x)+\mathrm{E}(D|x)\mathrm{E}(y_1-y_0|x)\\&=\mu_0(x)+\mathrm{E}(D|x)\tau(x)\end{aligned}$$

当x从左侧和右侧趋于c时，

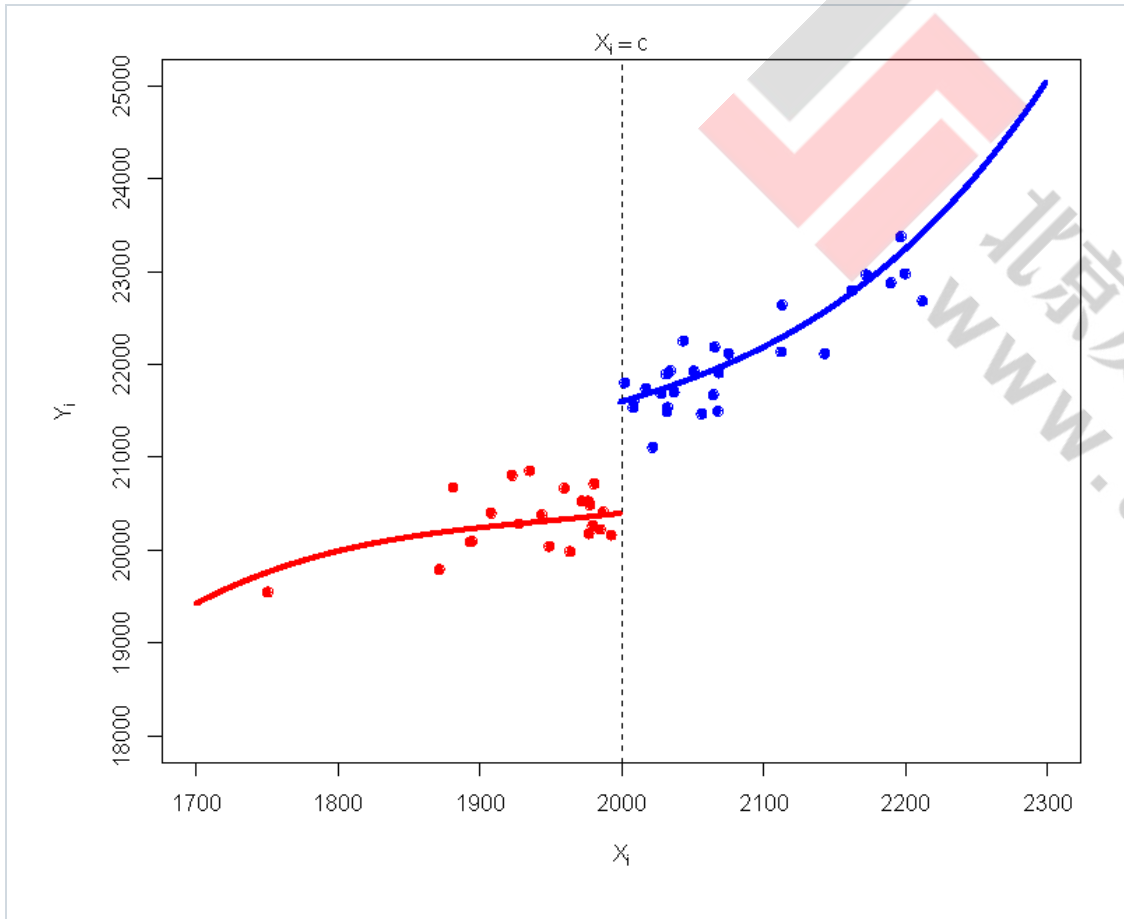
$$\begin{aligned}m^+(c)&=\mu_0(c)+P^+(c)\tau_{FRD}\\m^-(c)&=\mu_0(c)+P^-(c)\tau_{FRD}\end{aligned}$$

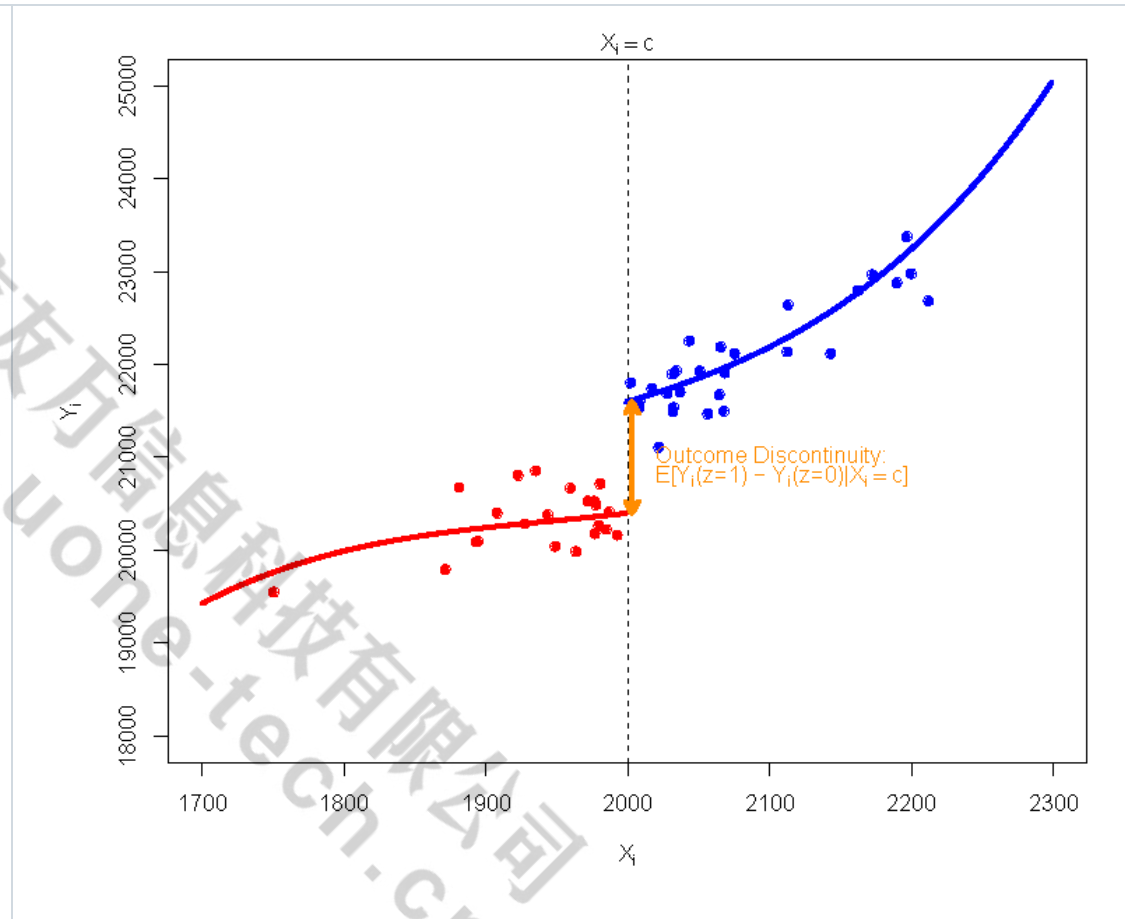
其中，

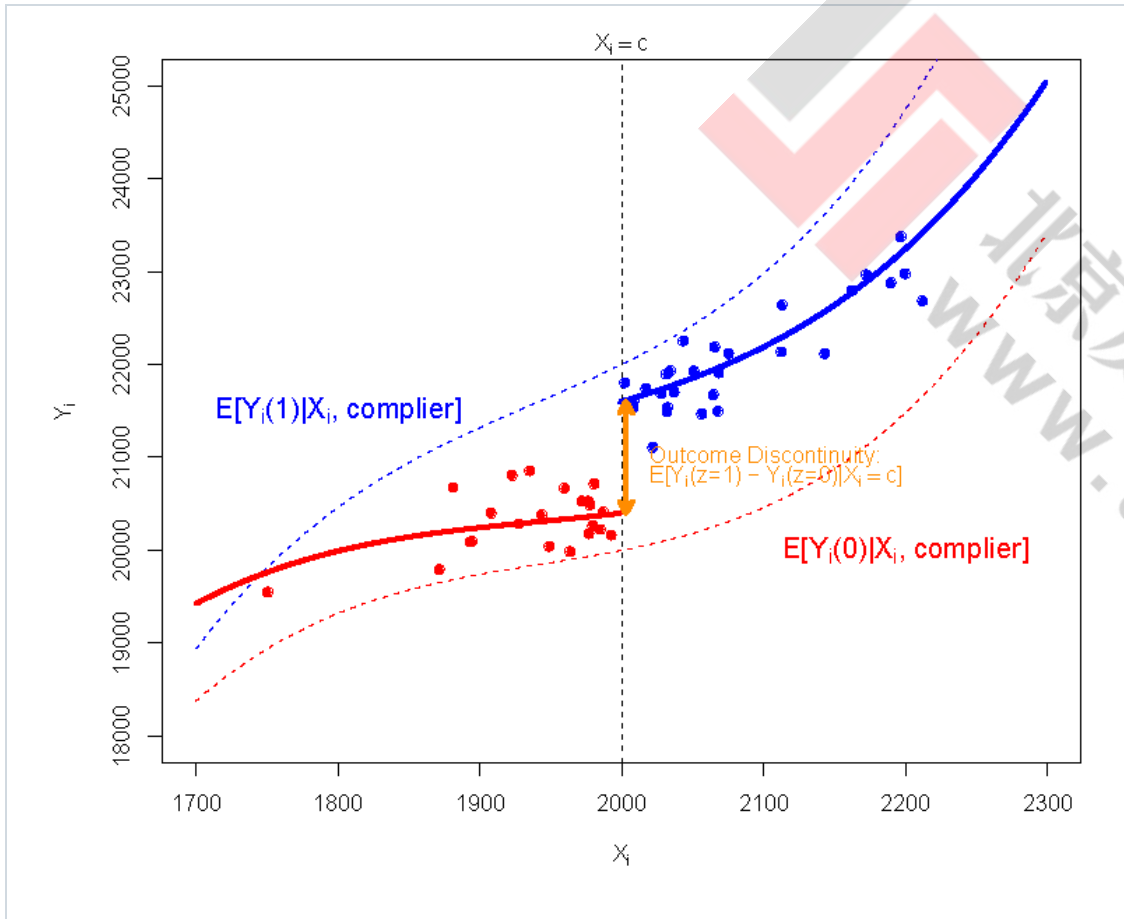
$$\begin{aligned}m^+(c)&=\mathrm{E}_{x\downarrow c}(y|x),\quad m^-(c)=\mathrm{E}_{x\uparrow c}(y|x),\\P^+(c)&=\mathrm{E}_{x\downarrow c}(D|x),\quad P^-(c)=\mathrm{E}_{x\uparrow c}(D|x).\end{aligned}$$

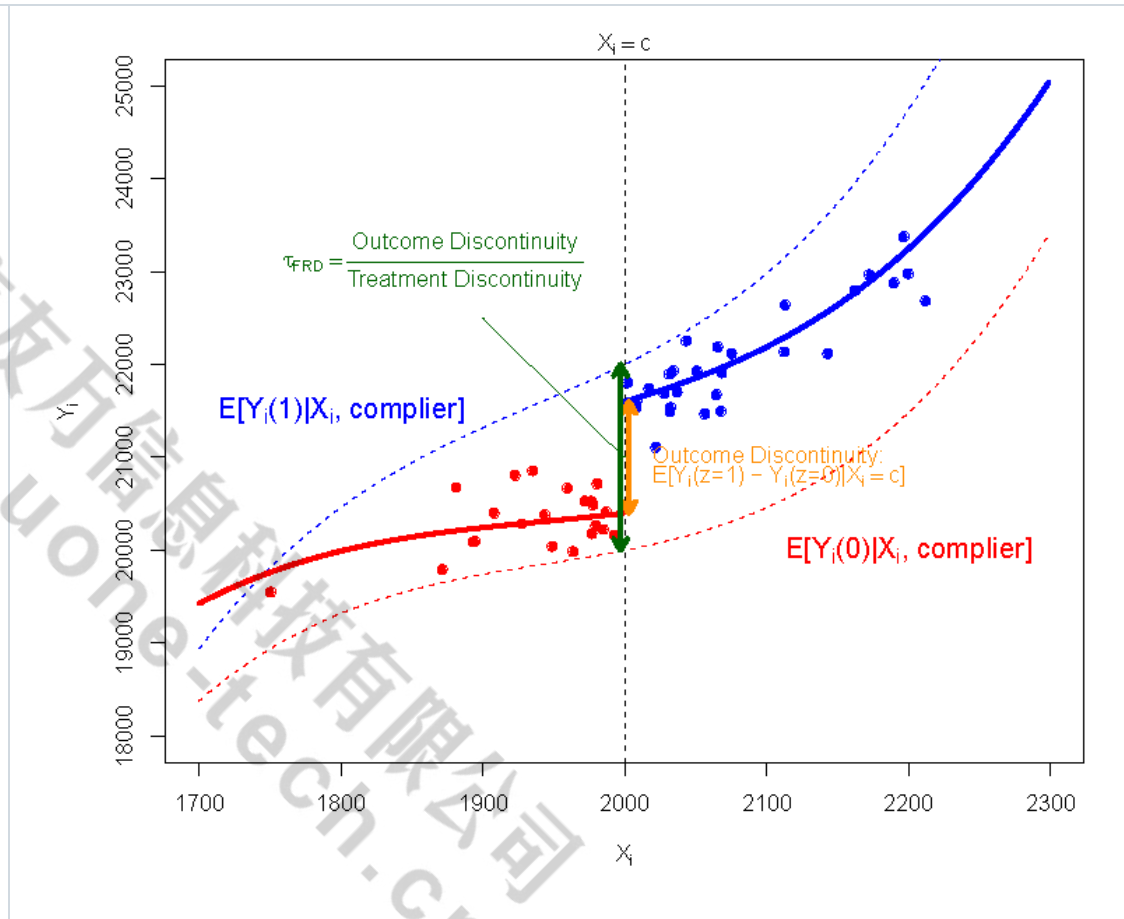
假定： $P^+(c) \neq P^-(c)$，可得

$$\tau_{FRD} = \frac{m^+(c) - m^-(c)}{P^+(c) - P^-(c)}$$









Imbens and Lemieux (2008)建议利用局部多项式回归。

假定：

$$(y_{0i}, y_{1i}) \perp w_i | x_i \in (c-h, c+h)$$

- 在$(c-h \leq x < c)$区间估计$(\hat{m}^+(c), \hat{P}^+(c))$，
- 在$(c < x \leq c+h)$区间估计$(\hat{m}^-(c), \hat{P}^-(c))$。

采用Probit或Logit估计$(\hat{P}^+(c), \hat{P}^-(c))$。

HTV(2001)证明，模糊断点估计等价于局部IV估计（Wald估计量）。

无条件的处理效应：

$$\hat{\theta}_{base}(h) = \frac{\hat{\tau}_{\mathrm{Y},base}(h)}{\hat{\tau}_{\mathrm{D},base}(h)} = \frac{\sum_{i=1}^{n} w_i(h) Y_i}{\sum_{i=1}^{n} w_i(h) D_i}$$

条件处理效应：线性调整（非参方法）

$$\hat{\theta}_{lin}(h) = \frac{\hat{\tau}_{\mathrm{Y},lin}(h)}{\hat{\tau}_{\mathrm{D},lin}(h)} = \frac{\sum_{i=1}^{n} w_i(h)(Y_i - X_i^T \hat{\gamma}_{Y,h})}{\sum_{i=1}^{n} w_i(h)(D_i - X_i^T \hat{\gamma}_{D,h})}$$

条件处理效应：非线性调整（DML）

$$\hat{\theta}_{\mathrm{RDFlex}}(h;\eta) = \frac{\sum_{i=1}^{n} w_i(h)(Y_i - \hat{\eta}_Y(X_i))}{\sum_{i=1}^{n} w_i(h)(D_i - \hat{\eta}_D(X_i))},$$

```
. use dmlfrd, clear
. summ

    Variable |        Obs        Mean    Std. dev.       Min        Max
-------------+---------------------------------------------------------
           y |      1,000    4.820415    4.099335  -1.895659   20.66342
           d |      1,000        .853    .3542831          0          1
       score |      1,000    .0193321    .9792159  -3.241267   3.852731
          x0 |      1,000    .0080321     .570023  -.9997306   .9988275
          x1 |      1,000   -.0135667    .5793069  -.9999767   .9991154
-------------+---------------------------------------------------------
          x2 |      1,000   -.0117039    .5765549  -.9999386   .9954988
```

```
. use dmlfrd, clear
. doublemlrd y x0 x1 x2, treat(d) score(score) method(dml)

Method          =             dml               No. of obs.     =         1000
Kernel          =      triangular               No. of folds    =            5
Bandwidth       =          1.0046               No. of rep      =            1

------------------------------------------------------------------------------
        LATE |      coef         se      tstat      pvale      cilow      ciupp
-------------+----------------------------------------------------------------
Conventional |   .3733619    .599275    .6230226    .5332697  -.8011955   1.547919
Biascorrec~d |   .4454569    .599275    .7433264    .4572841  -.7291005   1.620014
      Robust |   .4454569   .7078069    .6293481    .5291212   -.941819   1.832733
------------------------------------------------------------------------------
ml_g for E(y|X):  LassoCV()
ml_m for E(T|X):  LogisticRegressionCV()
```

```
. doublemlrd y x0 x1 x2, treat(d) score(score) method(dml) fuzzy(True)

Method          =             dml                 No. of obs.     =         1000
Kernel          =      triangular                 No. of folds    =            5
Bandwidth       =          0.9862                 No. of rep      =            1

------------------------------------------------------------------------------
        LATE |      coef         se      tstat      pvale      cilow      ciupp
-------------+----------------------------------------------------------------
Conventional |   3.23816   4.976168   .6507337   .5152184   -6.51495   12.99127
Biascorrec~d |   4.06652   4.976168    .817199   .4138147   -5.68659   13.81963
      Robust |   4.06652    5.85546   .6944834   .4873791  -7.409971   15.54301
------------------------------------------------------------------------------
ml_g for E(y|X):  LassoCV()
ml_m for E(T|X):  LogisticRegressionCV()
```

```
. local mody LGBMRegressor(n_estimators=500, learning_rate=0.01, verbose=-1)
. local modt LGBMClassifier(n_estimators=500, learning_rate=0.01, verbose=-1)
. doublemlrd y x0 x1 x2, treat(d) score(score) method(dml) fuzzy(True) modely(`mody') modelt(`modt')

Method          =             dml                 No. of obs.     =         1000
Kernel          =      triangular                 No. of folds    =            5
Bandwidth       =          0.9862                 No. of rep      =            1

------------------------------------------------------------------------------
        LATE |      coef         se      tstat     pvale     cilow      ciupp
-------------+----------------------------------------------------------------
Conventional |  2.741325   1.880668   1.457634   .1449415  -.9447166   6.427367
Biascorrec~d |  3.139905   1.880668   1.669569   .0950047  -.5461372   6.825947
      Robust |  3.139905   2.225679   1.410763   .1583146  -1.222346   7.502155
------------------------------------------------------------------------------
ml_g for E(y|X):  LGBMRegressor(n_estimators=500, learning_rate=0.01, verbose=-1)
ml_m for E(T|X):  LGBMClassifier(n_estimators=500, learning_rate=0.01, verbose=-1)
```

默认情况下，`RDFlex` 使用核权重拟合机器学习方法，从而在断点周围形成 “局部” 拟合。如果调整还应包含来自数据中可用全支撑集的 “全局” 信息，可以使用 `GlobalLearner` 学习器。`GlobalLearner` 忽略权重，并在数据的全支撑集上拟合机器学习方法。

学习器还可以进行堆叠。所有学习器必须在其拟合方法中支持样本权重。通过堆叠和使用局部及全局学习器，可以进一步调整估计，甚至有可能进一步降低标准误差。

主要内容：

- 局部线性模型
- 交互模型
- 机器学习因果推断
- **政策学习**

政策树是一种结合了决策树算法与政策优化的机器学习模型，主要用于在异质群体中确定最优政策分配策略。其核心目标是通过数据驱动的方式，将人群划分为不同子群体，并为每个子群体推荐最有效的政策（如是否发放优惠券、发放金额等），以最大化整体政策效果（如经济收益、社会福利等）。

政策树和政策森林的目标不是估计因果效应，而是直接学习最优决策规则。这与传统因果模型（如 DML、X-Learner）的设计逻辑有本质区别。

当目标是直接制定决策而非解释因果效应时，PolicyTree或PolicyForest 是更理想选择。典型场景包括：

- 个性化推荐：根据用户特征决定推荐内容（如广告、产品）。
- 资源分配：根据患者特征决定治疗方案。
- 政策评估：在无法进行随机实验时，从观测数据中学习最优政策。

政策树的基本结构:

- 根节点：包含全部样本，初始状态下未进行任何划分。
- 内部节点：基于特征（如年龄、收入、消费习惯等）对样本进行分裂，每个节点对应一个分裂条件（如 “收入是否> 5000 元”）。
- 终端节点（叶节点）：每个叶节点对应一个特定子群体，最终输出针对该群体的最优政策。

构建过程（以优惠券发放为例）

- 特征选择：选取与政策效果相关的特征（如用户历史消费额、活跃度、地域等）。
- 分裂准则：通过最大化政策效果差异（如不同子群体使用优惠券后的消费增量差异）确定最优分裂条件。常用方法包括：
  - 因果分裂：比较不同政策在子群体中的反事实效果（如发放 vs. 不发放优惠券的消费差异）。
  - 损失函数优化：最小化政策实施后的预期损失（如成本 - 收益差）。
- 终端节点政策确定：对每个叶节点内的样本，选择效果最优的政策（如发放 20 元优惠券比 10 元更能促进消费）。

政策树的应用：电商平台或零售企业需通过优惠券提升销售额，同时控制成本。

响应变量：历史优惠券使用次数、平均折扣敏感度（优惠金额/订单金额）

政策变量：优惠券类型，包括：无券（对照组）、满 100 减 10（轻度刺激）、满 200 减 50（重度刺激）

特征选取：用户历史消费金额、浏览时长、会员等级、地域消费水平、商品偏好（如是否购买过电子产品）、年龄、性别、会员等级（1-5 级）

树结构设计：

- 分裂条件："过去 3 个月消费≥5 次"→"客单价≥200 元"→"是否为会员"。

终端政策：

- 高频高消费会员：发放满 500 减 100 元大额优惠券（刺激复购与高客单价）；
- 高频低消费非会员：发放满 100 减 30 元优惠券（提升转化为会员的概率）；
- 低频用户：发放无门槛 10 元券（激活沉睡用户）。

政策树的应用：某国家利用政策树制定碳排放管控政策

特征选取：行业类型（高耗能 / 低耗能）、企业规模、过去三年碳排放量变化、减排技术投入。

分裂逻辑："行业为钢铁 / 化工"→"年碳排放量 > 10 万吨"→"是否已投入碳捕获技术"。

终端政策：

- 高耗能 + 高排放 + 未投入技术：征收高额碳税 + 强制减排目标；
- 高耗能 + 高排放 + 已投入技术：提供减排补贴 + 税收优惠；
- 低耗能企业：自愿减排奖励机制。

提升树（uplift tree）以提升增益为标准来构建的政策树。它通过决策树的结构，识别出哪些群体最能从干预中受益，从而优化资源分配。提升值定义为

$$Uplift = E[Y(1) - Y(0) \mid X]$$

提升树的目标在于找到特征 X 的哪些组合能最大化干预效果，而不是整体平均效果。

提升树种树的分裂准则：节点分裂时，选择能最大化子节点间提升值差异的特征和阈值。

- Hansotia and Rukstales (2002)：$\Delta = |\theta(Left) - \theta(Right)|$
- Rzepakowski and Jaroszewicz (2012)基于信息准则的定义：

$$D_{gain} = D_{after}(P^T, P^C) - D_{before}(P^T, P^C)$$

其中，P^T和P^C分别表示处理组和控制组的因变量的分布。

差异D（divergence）的三种定义：

- Kullback-Leibler (KL) :衡量干预组与控制组的分布差异。

$$KL(P:Q) = \sum_{k=left,right} p_k log \frac{p_k}{q_k}$$

p和q分别表示处理组和控制组的样本均值，k表示叶子。

- Euclidean Distance:直接比较两组的平均结果。

$$ED(P:Q) = \sum_{k=left,right} (p_k - q_k)^2$$

- χ^2 divergence: 检验处理效应在子节点中的显著性。

$$\chi^2(P:Q) = \sum_{k=left,right} \frac{(p_k - q_k)^2}{q_k}$$

提升树的其他分裂标准：

Athey and Imbens(2015): $\Delta = \frac{1}{\#children} \sum_{k=1}^{\#children} \theta_k^2.$

DDP (delta-delta-p, $\Delta\Delta p$) approach by [@Hansotia2002]

IDDP approach by [@Roessler2022]

Interaction Tree (IT) proposed by [@Su2009]

Causal Inference Tree (CIT) by [@Stuart2010]

Contextual Treatment Selection (CTS) approach by [@Zhao2017]

评估提升模型（uplift model）

提升曲线：Radcliff (2007, 2008) Qini measure.

首先，分别预测干预组和控制组的提升值，对两组样本分别计算每个个体的预测提升值，并从高到低排序，等分为 10 个分位数组（decile）。然后，计算每个分位数组内的平均预测提升值。然后，对每个分位数组，计算干预组与控制组的平均提升值之差。良好的提升模型的提升值应该随分位数组递减。

或者对分位数组的提升值之差计算累积和，得到累积提升值。

累计增益（cumulative gain）：每组的平均提升值乘以该组的个体数量

$$\left(\frac{Y^T}{N^T} - \frac{Y^C}{N^C}\right)(N^T + N^C)$$

其中，Y^T和y^C分别表示处理组和控制组的个体的因变量之和。

提升曲线 (uplift curve)：

$$f(q) = \left(\frac{Y_q^T}{N_q^T} - \frac{Y_q^C}{N^C} \right) (N_q^T + N_q^C)$$

其中，q表示按照提升值降序排序后后的前q比例的观测值。

令CATE的q分位数为$\hat{\mu}(q)$。如果按照CATE的由大到小的顺序将CATE最高的q比例的个体分配到处理组，那么该组的ATE应该高于整体的ATE，设$\tau(q)$为二者之间的差异。

$$\begin{aligned}\tau(q) &= E[Y(1)-Y(0) \mid \hat{\theta}(X) \geq \hat{\mu}(q)] - E[Y(1)-Y(0)] \\ &= E\left[(Y(1)-Y(0))\frac{\hat{\theta}(X) \geq \hat{\mu}(q)}{Pr(\hat{\theta}(X) \geq \hat{\mu}(q))}\right] - E[Y(1)-Y(0)] \\ &= \text{Cov}\left(Y(1)-Y(0), \frac{\hat{\theta}(X) \geq \hat{\mu}(q)}{Pr(\hat{\theta}(X) \geq \hat{\mu}(q))}\right)\end{aligned}$$

提升曲线也可以通过$Y^{DR}(g,p)$来计算：

$$\begin{aligned}\tau(q) = &\ E\left[Y^{DR}(g,p) \mid \hat{\theta}(X) \geq \hat{\mu}(q)\right] - E\left[Y^{DR}(g,p)\right] \\ = &\ \text{Cov}\left(Y^{DR}(g,p), \frac{1\{\hat{\theta}(X) \geq \hat{\mu}(q)\}}{\Pr(\hat{\theta}(X) \geq \hat{\mu}(q))}\right)\end{aligned}$$

QINI 曲线源于Uplift 曲线和累积增益曲线，其横轴为按预测因果效应从高到低排序的样本比例（通常以百分比表示），纵轴为累积干预收益（Cumulative Treatment Effect）与随机分配策略的收益差异。

提升曲线是由 CATE 模型优先排序的人群组的平均效应减去总的平均效应，QINI曲线是 CATE 模型优先排序的人群组的平均效应减去随机分配策略的收益差异。

基准线（Random Line）对角线：代表随机分配干预的效果（即无模型预测时的平均收益）。

- 若曲线低于对角线：说明模型表现不如随机分配，预测能力差。
- QINI 曲线的形状曲线上升越快：模型对高因果效应群体的识别能力越强。
- 曲线最高点：表示在该比例的样本上干预收益最大。曲线最高点对应的横轴值表示应干预的最佳样本比例。
- 曲线下降：说明后续群体的因果效应为负或模型预测不准确。

$$
\begin{aligned}
\tau_{\text{QINI}}(q) :=&\ \tau(q)\ \Pr(\hat{\theta}(X) \geq \hat{\mu}(q)) \\
=&\ \Big(E[Y(1) - Y(0) \mid \hat{\theta}(X) \geq \hat{\mu}(q)] - E[Y(1) - Y(0)]\Big)\ \Pr(\hat{\theta}(X) \geq \hat{\mu}(q)) \\
=&\ E\Big[(Y(1) - Y(0)) 1\{\hat{\theta}(X) \geq \hat{\mu}(q)\}\Big] - E[Y(1) - Y(0)]\ \Pr(\hat{\theta}(X) \geq \hat{\mu}(q)) \\
=&\ E\Big[(Y(1) - Y(0)) 1\{\hat{\theta}(X) \geq \hat{\mu}(q)\}\Big] - E[Y(1) - Y(0)]\, E\Big[1\{\hat{\theta}(X) \geq \hat{\mu}(q)\}\Big] \\
=&\ \text{Cov}\Big(Y(1) - Y(0), 1\{\hat{\theta}(X) \geq \hat{\mu}(q)\}\Big)
\end{aligned}
$$

$\tau_{\text{QINI}}(q) \approx q\tau(q)$.

将$Y(1) - Y(0)$替换为$Y^{DR}(g, p)$，即可识别$\tau_{\text{QINI}}(q)$:

$$
\tau_{\text{QINI}}(q) := \text{Cov}\Big(Y^{DR}(g, p), 1\{\hat{\theta}(X) \geq \hat{\mu}(q)\}\Big)
$$

Qini曲线：

$$g(q) = \left(Y_q^T - \frac{Y_q^C N_q^T}{N_q^C} \right)$$

Qini曲线与提升曲线是平行的，二者存在关系：

$$f(q) = g(q) \frac{N_q^T + N_q^C}{N_q^T}$$

Qini系数定义为

$$\mathrm{QINI} = \int_0^1 \tau_{\mathrm{QINI}}(q)dq$$

QINI系数：模型曲线与随机线之间的面积除以完美模型与随机线之间的面积，取值范围[0,1]，越接近 1 表示模型效果越好。计算公式：

$$\mathrm{QINI} = \frac{\text{Area between QINI curve and random line}}{\text{Area under the perfect model line}}$$

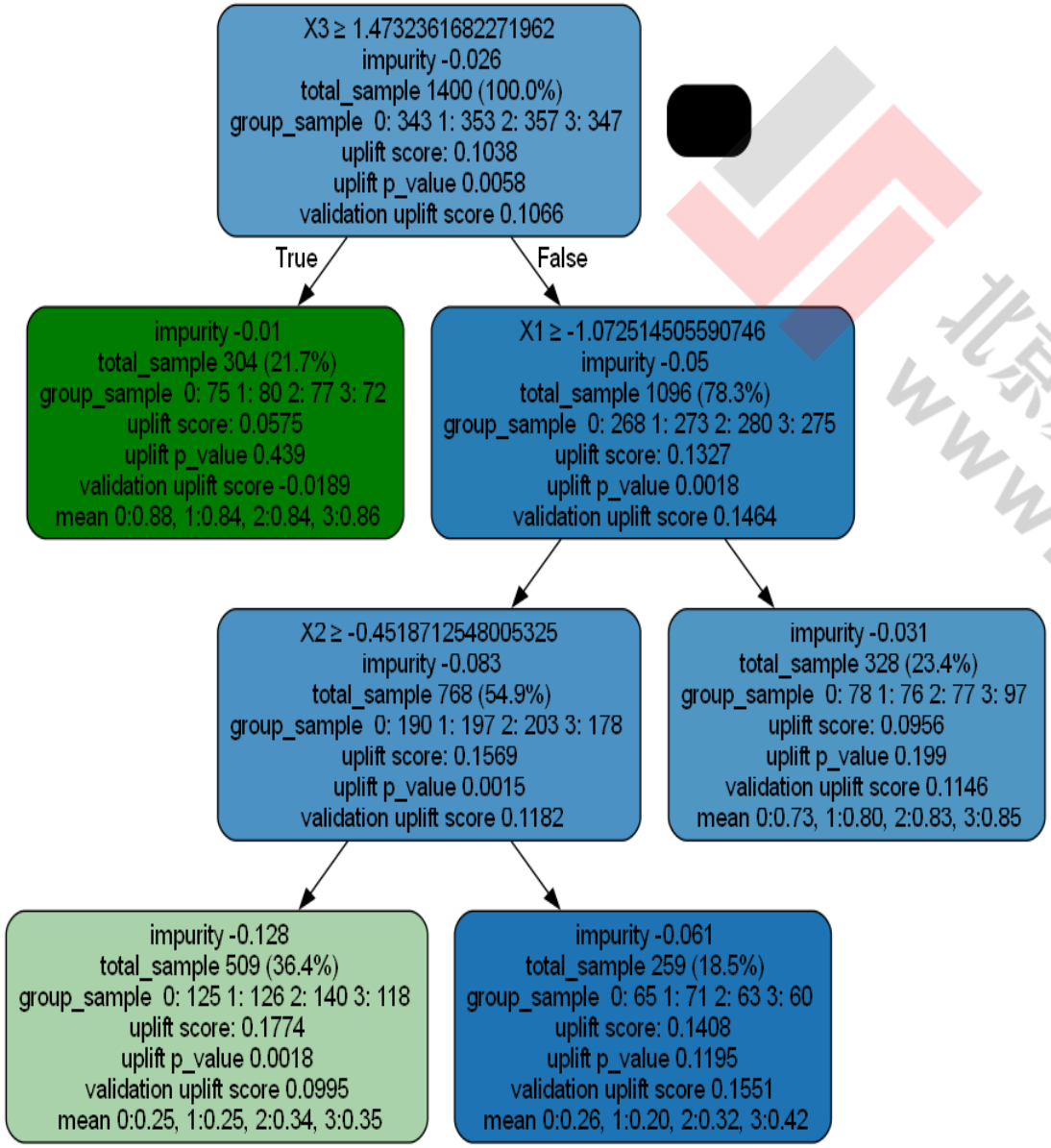
- QINI = 1：完美模型（能完全区分高 / 低因果效应群体）。
- QINI = 0：等同于随机分配。

```
. use upliftdata, clear
. causalmluplift y X1-X5, treat(d) criterion("KL")

Predicted effect of each treatment (first ten obs):

-------------------------------------------
             |  ATE(d=1)   ATE(d=2)   ATE(d=3)
-------------+-----------------------------
          r1 | -.0643554   .0559219   .1551282
          r2 |  .0059683   .0877143   .0994576
          r3 |  .0718623   .1003996   .1145916
          r4 |  .0059683   .0877143   .0994576
          r5 |    -.0425  -.0358442  -.0188889
-------------------------------------------
Policy tree saved as cmltree.png, cmltree.dot and upliftcurve.png under the working directory
[cmlest.joblib]
```

`criteion` : 'KL', 'ED', 'Chi', 'CTS', 'DDP', 'IT', 'CIT', 'IDDP'



节点分裂条件：决定样本被划分到左子树或右子树的规则（X3>1.47）。

impurity: 损失函数值，衡量节点样本异质性的指标（如均方误差 MSE 或不纯度），分裂时目标是降低该值。

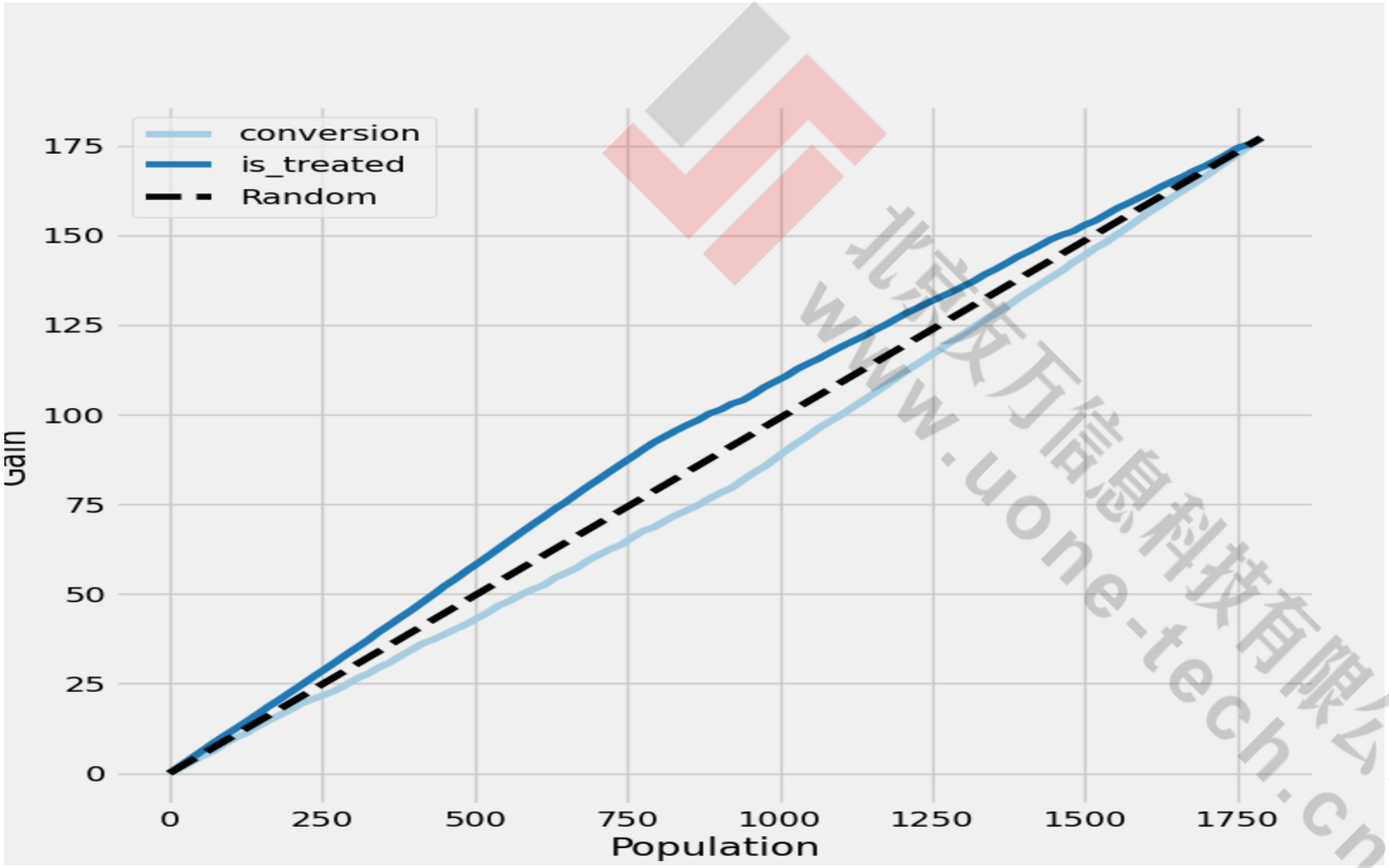
total_sample: 当前节点的总样本量

group_sample: 按处理组划分的样本量

uplift score: 处理组与控制组的因果效应，即$E[Y(1)-Y(0)]$，多处理组时取最大效应值。

uplift p_value: 因果效应的 p 值

validation uplift score: 使用验证数据计算



谢 谢！