xporegress — Cross-fit partialing-out lasso linear regression

Description

xporegress fits a lasso linear regression model and reports coefficients along with standard errors, test statistics, and confidence intervals for specified covariates of interest. The cross-fit partialing-out method is used to estimate effects for these variables and to select from potential control variables to be included in the model.

Quick start

Estimate a coefficient for \( d_1 \) in a linear regression of \( y \) on \( d_1 \), and include \( x_1-\text{x100} \) as potential control variables to be selected by lassos

\[
\text{xporegress } y \ d_1, \ \text{controls}(x_1\text{-x100})
\]

As above, and estimate coefficients for the levels of categorical \( d_2 \)

\[
\text{xporegress } y \ d_1 \ i.d_2, \ \text{controls}(x_1\text{-x100})
\]

As above, but use 20 folds instead of 10 for cross-fitting

\[
\text{xporegress } y \ d_1 \ i.d_2, \ \text{controls}(x_1\text{-x100}) \ \text{xfolds}(20)
\]

As above, but repeat the cross-fitting procedure 15 times, and average the results

\[
\text{xporegress } y \ d_1 \ i.d_2, \ \text{controls}(x_1\text{-x100}) \ \text{xfolds}(20) \ \text{resample}(15)
\]

Use cross-validation (CV) instead of a plugin iterative formula to select the optimal \( \lambda^* \) in each lasso

\[
\text{xporegress } y \ d_1 \ i.d_2, \ \text{controls}(x_1\text{-x100}) \ \text{selection(cv)}
\]

As above, and set a random-number seed for reproducibility

\[
\text{xporegress } y \ d_1 \ i.d_2, \ \text{controls}(x_1\text{-x100}) \ \text{selection(cv)} \ \text{rseed}(28)
\]

Specify CV for the lasso for \( y \) only, with the stopping rule criterion turned off

\[
\text{xporegress } y \ d_1 \ i.d_2, \ \text{controls}(x_1\text{-x100}) \ \text{lasso}(y, \ \text{selection(cv)}, \ \text{stop}(0))
\]

As above, but apply the option to the lassos for \( y, \ d_1, \) and \( i.d_2 \)

\[
\text{xporegress } y \ d_1 \ i.d_2, \ \text{controls}(x_1\text{-x100}) \ \text{lasso}(*, \ \text{selection(cv)}, \ \text{stop}(0))
\]

Menu

Statistics > Lasso > Lasso inferential models > Continuous outcomes > Cross-fit partialing-out model
### Syntax

```bash
xporegress depvar varsofinterest [if] [in],
    controls([(alwaysvars)] othervars) [options]
```

*varsofinterest* are variables for which coefficients and their standard errors are estimated.

**options**

<table>
<thead>
<tr>
<th>Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td></td>
</tr>
<tr>
<td><em>controls([(alwaysvars)] othervars) alwaysvars and othervars make up the set of control variables; alwaysvars are always included; lassos choose whether to include or exclude othervars</em></td>
<td></td>
</tr>
<tr>
<td>selection(plugin)</td>
<td>use a plugin iterative formula to select an optimal value of the lasso penalty parameter λ* for each lasso; the default</td>
</tr>
<tr>
<td>selection(cv)</td>
<td>use CV to select an optimal value of the lasso penalty parameter λ* for each lasso</td>
</tr>
<tr>
<td>selection(adaptive)</td>
<td>use adaptive lasso to select an optimal value of the lasso penalty parameter λ* for each lasso</td>
</tr>
<tr>
<td>sqrtlasso</td>
<td>use square-root lassos</td>
</tr>
<tr>
<td>xfolds(#)</td>
<td>use # folds for cross-fitting</td>
</tr>
<tr>
<td>resample[(#)]</td>
<td>repeat sample splitting # times and average results</td>
</tr>
<tr>
<td>technique(dml1</td>
<td>dml2)</td>
</tr>
<tr>
<td>semi</td>
<td>use semi partialing-out lasso regression estimator</td>
</tr>
<tr>
<td>missingok</td>
<td>after fitting lassos, ignore missing values in any othervars not selected, and include these observations in the final model</td>
</tr>
<tr>
<td>Reporting</td>
<td></td>
</tr>
<tr>
<td>level(#)</td>
<td>set confidence level; default is level(95)</td>
</tr>
<tr>
<td>display_options</td>
<td>control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling</td>
</tr>
<tr>
<td>Optimization</td>
<td></td>
</tr>
<tr>
<td>[no]log</td>
<td>display or suppress an iteration log</td>
</tr>
<tr>
<td>verbose</td>
<td>display a verbose iteration log</td>
</tr>
<tr>
<td>rseed(#)</td>
<td>set random-number seed</td>
</tr>
<tr>
<td>Advanced</td>
<td></td>
</tr>
<tr>
<td>lasso(varlist, lasso_options)</td>
<td>specify options for the lassos for variables in varlist; may be repeated</td>
</tr>
<tr>
<td>sqrtlasso(varlist, lasso_options)</td>
<td>specify options for square-root lassos for variables in varlist; may be repeated</td>
</tr>
<tr>
<td>vce(robust)</td>
<td>robust VCE is the only VCE available</td>
</tr>
<tr>
<td>reestimate</td>
<td>refit the model after using lassoselect to select a different λ*</td>
</tr>
<tr>
<td>noheader</td>
<td>do not display the header on the coefficient table</td>
</tr>
<tr>
<td>coeflegend</td>
<td>display legend instead of statistics</td>
</tr>
</tbody>
</table>
*controls() is required.

`varsofinterest`, `alwaysvars`, and `othervars` may contain factor variables. Base levels of factor variables cannot be set for `alwaysvars` and `othervars`. See [U] 11.4.3 Factor variables.

`vce(robust), reestimate, noheader, and coeflegend` do not appear in the dialog box.

See [U] 20 Estimation and postestimation commands for more capabilities of estimation commands.

### Options

The `xpregress` command offers several options to customize the estimation process.

**Model**

- `controls([alwaysvars] othervars)` specifies the set of control variables, which control for omitted variables. Control variables are also known as confounding variables. `xpregress` fits lassos for `depvar` and each of the `varsofinterest`. `alwaysvars` are variables that are always to be included in these lassos. `alwaysvars` are optional. `othervars` are variables that each lasso will choose to include or exclude. That is, each lasso will select a subset of `othervars` and other lassos will potentially select different subsets of `othervars`. `controls()` is required.

- `selection(plugin|cv|adaptive)` specifies the selection method for choosing an optimal value of the lasso penalty parameter $\lambda^*$ for each lasso or square-root lasso estimation. Separate lassos are estimated for `depvar` and each variable in `varsofinterest`. Specifying `selection()` changes the selection method for all of these lassos. You can specify different selection methods for different lassos using the option `lasso()` or `sqrtlasso()`. When `lasso()` or `sqrtlasso()` is used to specify a different selection method for the lassos of some variables, they override the global setting made using `selection()` for the specified variables.

- `selection(plugin)` is the default. It selects $\lambda^*$ based on a “plugin” iterative formula dependent on the data. See [LASSO lasso options].

- `selection(cv)` selects the $\lambda^*$ that gives the minimum of the CV function. See [LASSO lasso options].

- `selection(adaptive)` selects $\lambda^*$ using the adaptive lasso selection method. It cannot be specified when `sqrtlasso` is specified. See [LASSO lasso options].

- `sqrtlasso` specifies that square-root lassos be done rather than regular lassos. The option `lasso()` can be used with `sqrtlasso` to specify that regular lasso be done for some variables, overriding the global `sqrtlasso` setting for these variables. See [LASSO lasso options].

- `xfolds(#)` specifies the number of folds for cross-fitting. The default is 10.

- `resample[#]` specifies that sample splitting be repeated and results averaged. This reduces the effects of the randomness of sample splitting on the estimated coefficients. Not specifying `resample` or `resample(#)` is equivalent to specifying `resample(1)`. In other words, by default no resampling is done. Specifying `resample` alone is equivalent to specifying `resample(10)`. That is, sample splitting is repeated 10 times. For each sample split, lassos are computed. So when this option is not specified, lassos are repeated `xfolds(#)` times. But when `resample(#)` is specified, lassos are repeated `xfolds(#) × resample(#)` times. Thus, while we recommend using `resample` to get final results, note that it can be an extremely time-consuming procedure.

- `technique(dm11|dm12)` specifies which cross-fitting technique is used, either double machine learning 1 (dm11) or double machine learning 2 (dm12). For both techniques, the initial estimation steps are the same. The sample is split into $K = xfolds(#)$ folds. Then, coefficients on the controls are estimated using only the observations not in the $k$th fold, for $k = 1, 2, \ldots, K$. Moment conditions for the coefficients on the `varsofinterest` are formed using the observations in fold $k$. The default technique, `dm12`, solves the moment conditions jointly across all the observations. The optional technique, `dm11`, solves the moment conditions in each fold $k$ to produce $K$ different estimates, which are then averaged to form a single vector of estimates. See Methods and formulas.
semi specifies that the semi partialing-out lasso regression estimator be used instead of the fully partialing-out lasso estimator, which is the default. See Methods and formulas.

missingok specifies that, after fitting lassos, the estimation sample be redefined based on only the nonmissing observations of variables in the final model. In all cases, any observation with missing values for depvar, varsofinterest, alwaysvars, and othervars is omitted from the estimation sample for the lassos. By default, the same sample is used for calculation of the coefficients of the varsofinterest and their standard errors.

When missingok is specified, the initial estimation sample is the same as the default, but the sample used for the calculation of the coefficients of the varsofinterest can be larger. Now observations with missing values for any othervars not selected will be added to the estimation sample (provided there are no missing values for any of the variables in the final model).

missingok may produce more efficient estimates when data are missing completely at random. It does, however, have the consequence that estimation samples can change when selected variables differ in models fit using different selection methods. That is, when othervars contain missing values, the estimation sample for a model fit using the default selection(plugin) will likely differ from the estimation sample for a model fit using, for example, selection(cv).

level(#); see [R] Estimation options.

display_options: noci, nopvalues, noomitted, vsquish, noemptycells, baselevels, allbaselevels, nofvlabel, fvwrap(#), fvwrapon(style), cformat(%fmt), pformat(%fmt), sformat(%fmt), and nolstretch; see [R] Estimation options.

[no]log displays or suppresses a log showing the progress of the estimation. By default, one-line messages indicating when each lasso estimation begins are shown. Specify verbose to see a more detailed log.

verbose displays a verbose log showing the iterations of each lasso estimation. This option is useful when doing selection(cv) or selection(adaptive). It allows you to monitor the progress of the lasso estimations for these selection methods, which can be time consuming when there are many othervars specified in controls().

rseed(#) sets the random-number seed. This option can be used to reproduce results. rseed(#) is equivalent to typing set seed # prior to running xporegress. Random numbers are used to produce split samples for cross-fitting. So for all selection() options, if you want to reproduce your results, you must either use this option or use set seed. See [R] set seed.

lasso(varlist, lasso_options) lets you set different options for different lassos, or advanced options for all lassos. You specify a varlist followed by the options you want to apply to the lassos for these variables. varlist consists of one or more variables from depvar or varsofinterest. _all or * may be used to specify depvar and all varsofinterest. This option is repeatable as long as different variables are given in each specification. lasso_options are selection(...), grid(...), stop(#), tolerance(#), dtolerance(#), and cvtolerance(#). When lasso(varlist, selection(...)) is specified, it overrides any global selection() option for the variables in varlist. It also overrides the global sqrtlasso option for these variables. See [LASSO] lasso options.

sqrtlasso(varlist, lasso_options) works like the option lasso(), except square-root lassos for the variables in varlist are done rather than regular lassos. varlist consists of one or more variables
from depvar or varsofinterest. This option is repeatable as long as different variables are given in each specification. lasso_options are selection(...), grid(...), stop(#), tolerance(#), dtolerance(#), and cvtolerance(#). When sqrtlasso(varlist, selection(...)) is specified, it overrides any global selection() option for the variables in varlist. See [LASSO] lasso options.

The following options are available with xporegress but are not shown in the dialog box:

vce(robust) is the only variance–covariance estimator available for these models. The large-sample variance of the method of moments estimator for the partialed-out model has the sandwich structure, the same structure that would be termed a “robust” variance estimator. Only under special assumptions, like a correctly specified parametric model, can the robust form be simplified. No such assumptions are commonly used for partialed-out models, so only the robust VCE is available.

reestimate is an advanced option that refits the xporegress model based on changes made to the underlying lassos using lassoselect. After running xporegress, you can select a different $\lambda^*$ for one or more of the lassos estimated by xporegress. After selecting $\lambda^*$, you type xporegress, reestimate to refit the xporegress model based on the newly selected $\lambda$’s.

reestimate may be combined only with reporting options.

noheader prevents the coefficient table header from being displayed.

doflegend; see [R] Estimation options.

Remarks and examples

xporegress performs cross-fit partialing-out lasso linear regression. This command estimates coefficients, standard errors, and confidence intervals and performs tests for variables of interest while using lassos to select from among potential control variables.

The linear regression model is

$$E[y|d, x] = d\alpha' + x\beta'$$

where $d$ are the variables for which we wish to make inferences and $x$ are the potential control variables from which the lassos select. xporegress reports estimated coefficients for $\alpha$. However, cross-fit partialing-out does not provide estimates of the coefficients on the control variables ($\beta$) or their standard errors. No estimation results can be reported for $\beta$.

For an introduction to the cross-fit partialing-out lasso method for inference, as well as the double-selection and partialing-out methods, see [LASSO] Lasso inference intro.

Examples that demonstrate how to use xporegress and the other lasso inference commands are presented in [LASSO] Inference examples. In particular, we recommend reading 1 Overview for an introduction to the examples and to the vl command, which provides tools for working with the large lists of variables that are often included when using lasso methods. See 2 Fitting and interpreting inferential models for examples of fitting inferential lasso linear models and comparisons of the different methods available in Stata.

If you are interested in digging deeper into the lassos that are used to select controls, see 5 Exploring inferential model lassos in [LASSO] Inference examples.
Stored results

`xporegress` stores the following in `e()`:

Scalars
- `e(N)`: number of observations
- `e(k_varsofinterest)`: number of variables of interest
- `e(k_controls)`: number of potential control variables
- `e(k_controls_sel)`: number of selected control variables
- `e(df)`: degrees of freedom for test of variables of interest
- `e(chi2)`: $\chi^2$ test of variables of interest
- `e(p)`: $p$-value for test of variables of interest
- `e(nxfolds)`: number of folds for cross-fitting
- `e(nresample)`: number of resamples
- `e(rank)`: rank of `e(V)`

Macros
- `e(cmd)`: `xporegress`
- `e(cmdline)`: command as typed
- `e(depvar)`: name of dependent variable
- `e(lasso_depvars)`: names of dependent variables for all lassos
- `e(varsofinterest)`: variables of interest
- `e(controls)`: potential control variables
- `e(controls_sel)`: selected control variables
- `e(model)`: linear
- `e(title)`: title in estimation output
- `e(chi2type)`: Wald; type of $\chi^2$ test
- `e(vctype)`: vcetype specified in vce()
- `e(properties)`: b V
- `e(rngstate)`: random-number state used
- `e(predict)`: program used to implement predict
- `e(select_cmd)`: program used to implement lassoselect
- `e(marginsnotok)`: predictions disallowed by margins
- `e(asbalanced)`: factor variables fvset as asbalanced
- `e(asobserved)`: factor variables fvset as asobserved

Matrices
- `e(b)`: coefficient vector
- `e(V)`: variance–covariance matrix of the estimators

Functions
- `e(sample)`: marks estimation sample

Methods and formulas

`xporegress` implements cross-fit partialing-out lasso linear regression as described in Chernozhukov et al. (2018), where they derived two versions of cross-fitting that are known as double machine learning 1 (DML1) and double machine learning 2 (DML2). DML2 is the default method and corresponds with option `technique(dml2)`. Specify option `technique(dml1)` to get DML1 instead.

Methods DML1 and DML2 have a similar structure. Each does the following.

1. Partitions the sample into $K$ folds.
2. Uses the post-lasso estimates computed using the observations not in a specific fold to fill in the moment conditions for the observations in that fold.

DML1 solves the moment conditions using the observations in each fold to produce $K$ different estimates and then averages these $K$ estimates to produce the final estimate for the coefficients of interest. DML2 uses all the observations to solve the moment conditions to produce a single final estimate for the coefficients of interest.
xporegress implements two methods for the partialing-out lasso regression. We call the default method partialing-out lasso regression (POLR). We call the optional method, obtained by specifying option sem, a semi partialing-out lasso regression (SPOLR). See Methods and formulas in [LASSO] poregress for a brief literature review of POLR and SPOLR.

The regression model is

$$E[y|d, x] = d\alpha' + \beta_0 + x\beta'$$

where \(d\) contains the \(J\) covariates of interest and \(x\) contains the \(p\) controls. The number of covariates in \(d\) must be small and fixed. The number of controls in \(x\) can be large and, in theory, can grow with the sample size; however, the number of nonzero elements in \(\beta\) must not be too large, which is to say that the model must be sparse.

**Cross-fit POLR algorithm**

1. Randomly partition the sample into \(K\) subsamples called folds.
2. Define \(I_k\) to be the observations in fold \(k\), and define \(IC_k\) to be the sample observations not in fold \(k\).
3. For each \(k = 1, \ldots, K\), fill in the observations of \(i \in I_k\) for the \(J\) moment conditions that identify \(\alpha\). These moment conditions use out-of-sample estimates of the high-dimensional components estimated using the observations \(i \in IC_k\).
   a. Using the observations \(i \in IC_k\), perform a linear lasso of \(y\) on \(x\) to select covariates \(\tilde{x}_{k,y}\).
      This lasso can choose the lasso penalty parameter (\(\lambda^*\)) using one of the plugin estimators for a linear lasso, adaptive lasso, or CV. The heteroskedastic plugin estimator for the linear lasso is the default.
   b. Using the observations \(i \in IC_k\), fit a linear regression of \(y\) on \(\tilde{x}_{k,y}\), and let \(\hat{\delta}_k\) be the estimated coefficients on \(\tilde{x}_{k,y}\).
   c. For the observations \(i \in I_k\), fill in the partial outcome.
      $$\tilde{y}_i = y_i - \tilde{x}_{k,y,i} \hat{\delta}'_k$$
   d. Using the observations \(i \in IC_k\), for each \(j = 1, \ldots, J\), perform a linear lasso of \(d_j\) on \(x\) to select covariates \(\tilde{x}_{k,j}\).
      Each of these lassos can choose the lasso penalty parameter (\(\lambda^*_j\)) using one of the plugin estimators for a linear lasso, adaptive lasso, or CV. The heteroskedastic plugin estimator for the linear lasso is the default.
   e. Using the observations \(i \in IC_k\), for \(j = 1, \ldots, J\), fit a linear regression of \(d_j\) on \(\tilde{x}_{k,j}\), and denote the coefficient estimates by \(\hat{\gamma}_{k,j}\).
   f. For each observation \(i \in I_k\), and for \(j = 1, \ldots, J\), fill in the instrument
      $$z_{j,i} = d_{j,i} - \tilde{x}_{k,j,i} \hat{\gamma}'_{k,j}$$
   g. For each observation \(i \in I_k\), collect the instruments into a vector \(z_i = (z_{1,i}, z_{2,i}, \ldots, z_{J,i})\).
4. Compute the point estimates.

For DML2, compute $\hat{\alpha}$ by solving the following sample-moment equations.

$$\frac{1}{n} \sum_{i=1}^{n} z_i' (\tilde{y}_i - z_i \alpha') = 0$$

For DML1, $\hat{\alpha}$ is given by

$$\hat{\alpha} = \frac{1}{K} \sum_{k=1}^{K} \hat{\alpha}_k$$

where $\hat{\alpha}_k$ is computed by solving the sample-moment equations

$$\frac{1}{n_k} \sum_{i \in I_k} z_i' (\tilde{y}_i - z_i \alpha'_k) = 0$$

and $n_k$ is the number of observations in $I_k$.

5. The VCE is estimated by

$$\widehat{\text{Var}}(\hat{\alpha}) = \frac{1}{n} J_0^{-1} \hat{\Psi} \left( J_0^{-1} \right)'$$

where

$$\hat{\Psi} = \frac{1}{K} \sum_{k=1}^{K} \hat{\Psi}_k$$

$$\hat{\Psi}_k = \frac{1}{n_k} \sum_{i \in I_k} \hat{\psi}_i \hat{\psi}_i'$$

$$\hat{\psi}_i = z_i' (\tilde{y}_i - z_i \hat{\alpha}')$$

$$\hat{J}_0 = \frac{1}{K} \sum_{k=1}^{K} \left( \frac{1}{n_k} \sum_{i \in I_k} \hat{\psi}_i^a \right)$$

and

$$\hat{\psi}_i^a = z_i' z_i$$
Cross-fit SPOLR algorithm

1. Randomly partition the sample into $K$ subsamples called folds.

2. Define $I_k$ to be the observations in fold $k$, and define $IC_k$ to be the sample observations not in fold $k$.

3. For each $k = 1, \ldots, K$, fill in the observations of $i \in I_k$ for the $J$ moment conditions that identify $\alpha$. These moment conditions use out-of-sample estimates of the high-dimensional components estimated using the observations $i \in IC_k$.
   
   a. Using the observations $i \in IC_k$, perform a linear lasso of $y$ on $d$ and $x$ to select covariates $\widetilde{x}_{k,y}$.

      This lasso can choose the lasso penalty parameter ($\lambda^\star$) using one of the plugin estimators for a linear lasso, adaptive lasso, or CV. The heteroskedastic plugin estimator for the linear lasso is the default.

   b. Using the observations $i \in IC_k$, fit a linear regression of $y$ on $d$ and $\widetilde{x}_{k,y}$, and let $\widehat{\delta}_k$ be the estimated coefficients on $\widetilde{x}_{k,y}$.

   c. For the observations $i \in I_k$, fill in the partial outcome.

   \[ \tilde{y}_i = y_i - \widetilde{x}_{k,y,i} \widehat{\delta}_k \]

   d. Using the observations $i \in IC_k$, for each $j = 1, \ldots, J$, perform a linear lasso of $d_j$ on $x$ to select covariates $\widetilde{x}_{k,j}$.

      Each of these lassos can choose the lasso penalty parameter ($\lambda^\star_j$) using one of the plugin estimators for a linear lasso, adaptive lasso, or CV. The heteroskedastic plugin estimator for the linear lasso is the default.

   e. Using the observations $i \in IC_k$, for $j = 1, \ldots, J$, fit a linear regression of $d_j$ on $\widetilde{x}_{k,j}$, and denote the coefficient estimates by $\widehat{\gamma}_{k,j}$.

   f. For each observation $i \in I_k$, and for $j = 1, \ldots, J$, fill in the instrument

   \[ z_{j,i} = d_{j,i} - \widetilde{x}_{k,j,i} \widehat{\gamma}_{k,j} \]

   g. For each observation $i \in I_k$, collect the instruments into a vector $z_i = (z_{1,i}, z_{2,i}, \ldots, z_{J,i})$.

4. Compute the point estimates.

   For DML2, compute $\widehat{\alpha}$ by solving the following sample-moment equations.

   \[ \frac{1}{n} \sum_{i=1}^{n} (\tilde{y}_i - d_i \alpha') z_i' = 0 \]

   For DML1, $\widehat{\alpha}$ is given by

   \[ \widehat{\alpha} = \frac{1}{K} \sum_{k=1}^{K} \widehat{\alpha}_k \]
where $\hat{\alpha}_k$ is computed by solving the sample-moment equations

$$\frac{1}{n_k} \sum_{i \in I_k} (\bar{y}_i - d_i \hat{\alpha}_k')z_i' = 0$$

and $n_k$ is the number of observations in $I_k$.

5. The VCE is estimated by

$$\hat{\text{Var}}(\hat{\alpha}) = \frac{1}{n} \hat{J}_0^{-1} \hat{\Psi} \left( \hat{J}_0^{-1} \right)'$$

where

$$\hat{\Psi} = \frac{1}{K} \sum_{k=1}^{K} \hat{\Psi}_k$$

$$\hat{\Psi}_k = \frac{1}{n_k} \sum_{i \in I_k} \hat{\psi}_i \hat{\psi}_i'$$

$$\hat{\psi}_i = (\bar{y}_i - d_i \hat{\alpha}')z_i'$$

$$\hat{J}_0 = \frac{1}{K} \sum_{k=1}^{K} \left( \frac{1}{n_k} \sum_{i \in I_k} \hat{\psi}_i^a \right)$$

and

$$\psi_i^a = d_i z_i'$$

See Methods and formulas in [LASSO] lasso for details on how the lassos in steps 3a and 3d of both algorithms choose their penalty parameters ($\lambda^*$).
Resampling the partitions

The $K$ folds are chosen once by default. Specify option `resample(#)` to have the $K$ folds randomly selected # times. This resampling removes the dependence of the estimator on any specifically selected folds, at the cost of more computer time.

Let $S$ be the specified number of resamples.

1. For each random partition $s = 1, \ldots, S$, use a cross-fit estimator to obtain the DM1 or the DM2 point estimates $\hat{\alpha}_s$ and the estimated VCE $\hat{\text{Var}}(\hat{\alpha}_s)$.

2. The mean resampling-corrected point estimates are

$$\tilde{\alpha} = \frac{1}{S} \sum_{s=1}^{S} \hat{\alpha}_s$$

3. The mean resampling-corrected estimate of the VCE is

$$\overline{\text{Var}}(\tilde{\alpha}) = \frac{1}{S} \sum_{s=1}^{S} \left\{ \text{Var}(\hat{\alpha}_s) + (\hat{\alpha}_s - \tilde{\alpha})(\hat{\alpha}_s - \tilde{\alpha})' \right\}$$

Reference


Also see

[LASSO] lasso inference postestimation — Postestimation tools for lasso inferential models

[LASSO] dsregress — Double-selection lasso linear regression

[LASSO] poregress — Partialing-out lasso linear regression

[R] regress — Linear regression

[U] 20 Estimation and postestimation commands