Title

stata.com

Lasso inference intro — Introduction to inferential lasso models

Description Remarks and examples References Also see

Description

Lasso selects covariates and estimates coefficients but does not provide the standard errors required for performing statistical inference. Stata provides three additional lasso-based methods for estimating the coefficients and standard errors for a subset of the covariates, and the results have the added advantage of being estimates of values from the true model that generated the data being analyzed.

The methods are double selection, partialing out, and cross-fit partialing out, which is also known as double machine learning. They can be applied to linear, logistic, and Poisson regression models. Partialing out and cross-fit partialing out can also be used with endogenous covariates and instrumental variables in linear models.

Remarks and examples

stata.com

Remarks are presented under the following headings:

The problem

You want to know the true effects of `z1` and `z2` on `y`, by which we mean the effect in the true underlying model that generated the data being analyzed. We specify two variables, but you could specify one or a handful.

You do not know whether `z1` and `z2` belong in the model, although you have your suspicions. Your problem is to estimate the effects (coefficients) of `z1` and `z2` and obtain their standard errors.

At the same time, you do not know the other variables that appear in the model, but you do know that they are among `x1-x500`. You also know that only a small number of them appear. We will be more precise about the meaning of "small" later.

Possible solutions

If you had a sufficient number of observations in your data, you could fit a linear regression of `y` on `z1`, `z2`, and `x1-x500`:

```
. regress y z1 z2 x1-x500
```

The above is a solution because including extra explanatory variables does not cause bias, at least as long as the number of covariates is not too large. Including extra variables merely causes a loss of efficiency. Except there may be no merely about it. You may not have a sufficient number of observations to fit this regression and answer questions about `z1` and `z2` with any degree of certainty.

In that case, you could use your scientific judgment to select the covariates that need to appear in the model:

```
. regress y z1 z2 x3 x9 x203 x333 x478
```

The problem here is that you must be correct about the variables you included and excluded. And the insight that led you to choose them cannot come from the data. And the choice you made is not testable. And theory seldom provides sufficient guidance about variables or their functional form. In the age of big data, modern research is increasingly looking for data-dependent guidance and rules.

Here is yet another solution. Select and fit the model using `lasso`, and force the inclusion of `z1` and `z2` with parentheses:

```
. lasso linear y (z1 z2) x1-x500
```

Now `lasso` will select a model and obtain its coefficients. Problem is, the lasso procedure does not provide standard errors. You cannot perform statistical tests of significance of `z1` and `z2` or obtain confidence intervals for them. And there is a reason for that. `lasso` does not account for mistakes in selecting from the potential covariates `x1-x500`. Any mistakes it makes in selecting covariates that are also correlated with `z1` or `z2` would lead to bias in estimating the coefficients and standard errors of `z1` and `z2`.

Here is a solution. Refit the model that `lasso` selected using `regress`. We would not recommend this. Everyone agrees that it would be better to split your data into two samples, use the first to select the model, and use the second to refit it. You will now have the standard errors you need. But even this will not provide good estimates and standard errors if your interest is in the true model that generated the data. The problem is twofold. First, this process still does not account sufficiently for the sampling variability of the selection process for `x1-x500`. Second, it does not account for the possibility of small coefficients in the true model. This second problem is more common and more detrimental than you might guess. See, for instance, Leeb and Pötscher (2005, 2006, 2008) and Belloni, Chernozhukov, and Hansen (2014a).

Solutions that focus on the true model

If your interest is inference about `z1` and `z2` in the true model that generated the data, the solution is to type

```
. dsregress y z1 z2, controls(x1-x500)
```

or

```
. poregress y z1 z2, controls(x1-x500)
```

or

```
. xporegress y z1 z2, controls(x1-x500)
```

These commands produce the double selection, partialing-out, and cross-fit partialing-out solutions, respectively, for the linear model, but commands also exist for logistic, Poisson, and instrumental-variables regression. These solutions all use multiple lassos and moment conditions that are robust to the model-selection mistakes that lasso makes; namely, that it does not select the covariates of the true model with probability 1. Of the three, the cross-fit partialing-out solution is best, but it can take a long time to run. The other two solutions are most certainly respectable. The cross-fit solution allows the true model to have more coefficients, and it allows the number of potential covariates, `x1-x500` in our examples, to be much larger. Technically, cross-fit has a less restrictive sparsity requirement.

All three of the methods have a sparsity requirement, and we have advice for you.

1. Let the commands fit the lassos using the default method, which is `plugin`.
2. If meeting the sparsity requirement concerns you, use cross-fit partialing out.

You may think of sparsity requirements as being unique to lasso, but they are not. Think about fitting an ordinary logistic regression model or any other estimator with only asymptotic properties. How many variables can be reliably included in the model if you have 100 observations? 500 observations? 1,000? 10,000? There is no answer to those questions except to say more with 1,000 than 500, more with 10,000 than 1,000, and so on.

The story is the same with the three inference methods. We can tell you more observations are better, and we can tell you more. Their requirements are stricter than those for logistic regression. The sparsity requirement for double selection and partialing out is that

$$\frac{s}{\sqrt{N}/\ln p} \qquad \text{is small}$$

where s is the number of covariates in the true model, N is the number of observations in the data, and p is the number of potential covariates. The sparsity requirement for cross-fit partialing out is the same, except that $\sqrt{N}$ is replaced by N. It is that

$$\frac{s}{N/\ln p} \qquad \text{is small}$$

N is much larger than $\sqrt{N}$. That is why we said that, if meeting the sparsity requirement concerns you, use cross-fit partialing out. It allows more covariates for all values of N.

We recommended that the lassos be fit using plugins because plugins were developed with these three methods in mind. Plugins tend to produce models with fewer "extra" covariates. Cross-validation, meanwhile, tends to include lots of extra covariates.

All three methods report the estimated coefficients for `z1` and `z2`, their standard errors, test statistics, and confidence intervals. Understanding how they work will be easier if we reduce the number of variables from two to one. Let's consider obtaining estimates for α in the model

$$y = d\alpha + \mathbf{x}\boldsymbol{\beta} + \epsilon$$

where d is the covariate of interest.

The double-selection solution

Double selection is the easiest of the three to explain. Its algorithm is the following:

1. Run a lasso of d on $\mathbf{x}$.
2. Run a lasso of y on $\mathbf{x}$.
3. Let $\widetilde{\mathbf{x}}$ be the union of the selected covariates from steps 1 and 2.
4. Regress y on d and $\widetilde{\mathbf{x}}$.

The estimate of α and its test statistics are then the coefficient on d and its test statistics.

Step 1 is the extra selection step from which double selection gets its name. It is this step that causes the method to be robust to the mistakes in model selection that lasso makes.

Stata provides three double-selection commands—`dsregress`, `dslogit`, and `dspoisson`.

The partialing-out solution

The algorithm is the following:

1. Run a lasso of d on $\mathbf{x}$. Let $\widetilde{\mathbf{x}}_d$ be the covariates selected.
2. Regress d on $\widetilde{\mathbf{x}}_d$. Let $\widetilde{d}$ be the residuals from this regression.
3. Run a lasso of y on $\mathbf{x}$. Let $\widetilde{\mathbf{x}}_y$ be the covariates selected.
4. Regress y on $\widetilde{\mathbf{x}}_y$. Let $\widetilde{y}$ be the residuals from this regression.
5. Regress $\widetilde{y}$ on $\widetilde{d}$.

The estimate of α and its test statistics are then the coefficient on $\widetilde{d}$ and its test statistics.

This algorithm is a high-dimensional version of the classic partialing-out estimator, which you can learn about in Wooldridge (2020, chap. 3-2). In the classic estimator, the moment conditions used to estimate the coefficient on d are orthogonal to the variables in $\mathbf{x}$. In the high-dimensional variant, the moment conditions in step 5 are orthogonal to the relevant variables in $\mathbf{x}$; thus, small changes in the variables included do not have a significant effect on the estimator for α.

Stata provides four partialing-out commands—`poregress`, `pologit`, `popoisson`, and `poivregress`.

`poivregress` provides a variation on the algorithm shown above that handles endogenous variables with instrumental variables in linear models.

The cross-fit partialing-out (double machine-learning) solution

Cross-fit partialing out is a split-sample version of partialing out. Cross-fit partialing out is also known as double machine learning (DML).

1. Divide the data in roughly equal-sized subsamples 1 and 2.
2. In sample 1:
    a. Run a lasso of d on $\mathbf{x}$. Let $\widetilde{\mathbf{x}}_{d1}$ be the covariates.
    b. Regress d on $\widetilde{\mathbf{x}}_{d1}$. Let $\widehat{\boldsymbol{\beta}}_1$ be the estimated coefficients.
    c. Run a lasso of y on $\mathbf{x}$. Let $\widetilde{\mathbf{x}}_{y1}$ be the covariates selected.
    d. Regress y on $\widetilde{\mathbf{x}}_{y1}$. Let $\widehat{\boldsymbol{\gamma}}_1$ be the estimated coefficients.
3. In sample 2:
    a. Fill in $\widetilde{d} = d - \widetilde{\mathbf{x}}_{d1}\widehat{\boldsymbol{\beta}}_1$.
    b. Fill in $\widetilde{y} = y - \widetilde{\mathbf{x}}_{y1}\widehat{\boldsymbol{\gamma}}_1$.
4. Still in sample 2:
    a. Run a lasso of d on $\mathbf{x}$. Let $\widetilde{\mathbf{x}}_{d2}$ be the covariates.
    b. Regress d on $\widetilde{\mathbf{x}}_{d2}$. Let $\widehat{\boldsymbol{\beta}}_2$ be the estimated coefficients.
    c. Run a lasso of y on $\mathbf{x}$. Let $\widetilde{\mathbf{x}}_{y2}$ be the covariates selected.
    d. Regress y on $\widetilde{\mathbf{x}}_{y2}$. Let $\widehat{\boldsymbol{\gamma}}_2$ be the estimated coefficients.

5. In sample 1:
   a. Fill in $\widetilde{d} = d - \widetilde{\mathbf{x}}_{d2}\widehat{\boldsymbol{\beta}}_2$.
   b. Fill in $\widetilde{y} = y - \widetilde{\mathbf{x}}_{y2}\widehat{\boldsymbol{\gamma}}_2$.
6. In the full sample: Regress $\widetilde{y}$ on $\widetilde{d}$.

The estimate of α and its test statistics are then the coefficient on $\widetilde{d}$ and its test statistics.

Cross-fit partialing out has a more relaxed sparsity requirement than partialing out and double selection, as we mentioned earlier. This is because the sample is split and coefficients are obtained from one sample and used in another, which is independent, and that adds robustness.

There are two variants of cross-fit partialing out (recall it is also known as DML): DML1 and DML2. Shown above is the algorithm for DML2, which is Stata's default method. DML1, available as an option, predates DML2 and solves the moment conditions within each fold (group) that is cross-fit and then averages. DML2, by comparison, solves the moment conditions jointly. See *Methods and formulas*. DML2 produced better results in simulations in Chernozhukov et al. (2018).

In the algorithm shown, the sample is split in two. The software splits it into K parts, where $K = 10$ by default. You could specify $K = 2$, but you would not want to do that. Larger K works better and $K = 10$ is viewed as sufficient. This is known as the 10-fold method.

Stata provides four cross-fit partialing-out commands—`xporegress`, `xpologit`, `xpopoisson`, and `xpoivregress`. `xpoivregress` provides a variation on the algorithms that handles endogenous variables with instrumental variables in linear models.

Where to learn more

See

[LASSO] **dsregress**	Double-selection lasso linear regression
[LASSO] **dslogit**	Double-selection lasso logistic regression
[LASSO] **dspoisson**	Double-selection lasso Poisson regression
[LASSO] **poregress**	Partialing-out lasso linear regression
[LASSO] **pologit**	Partialing-out lasso logistic regression
[LASSO] **popoisson**	Partialing-out lasso Poisson regression
[LASSO] **poivregress**	Partialing-out lasso instrumental-variables regression
[LASSO] **xporegress**	Cross-fit partialing-out lasso linear regression
[LASSO] **xpologit**	Cross-fit partialing-out lasso logistic regression
[LASSO] **xpopoisson**	Cross-fit partialing-out lasso Poisson regression
[LASSO] **xpoivregress**	Cross-fit partialing-out lasso instrumental-variables regression

And then there is the literature.

For a strikingly readable introduction, see Belloni, Chernozhukov, and Hansen (2014a). For a more technical discussion, see Belloni and Chernozhukov (2011).

Double selection was developed by Belloni, Chernozhukov, and Hansen (2014b). Their article also provides first-rate intuition on why the process works.

Partialing out was developed by Belloni et al. (2012). The 2012 date makes it appear that partialing out predates double selection, but the ordering was due to different publication lags. Their article also develops the plugin estimator for lasso and then develops the partialing-out instrumental-variables estimator. Partialing out was extended from linear to nonlinear models by Belloni, Chernozhukov,

and Wei (2016). For a three-page introduction to the partialing-out instrumental-variables estimator, see Chernozhukov, Hansen, and Spindler (2015).

Cross-fit partialing out was developed by Chernozhukov et al. (2018). The researchers of this article had worked on these issues for years. They later came together to assemble this important article, which is an odd but appealing mix of intuition and technical derivations, especially concerning sample splitting.

Bickel, Ritov, and Tsybakov (2009) predates all the above and provided the theoretical foundations for what would become the plugin estimator. It also provided rates of convergence for the lasso which was used by subsequent authors. The article is both seminal and technical.

References

Belloni, A., D. Chen, V. Chernozhukov, and C. B. Hansen. 2012. Sparse models and methods for optimal instruments with an application to eminent domain. *Econometrica* 80: 2369–2429. https://doi.org/10.3982/ECTA9626.

Belloni, A., and V. Chernozhukov. 2011. High dimensional sparse econometric models: An Introduction. In *Inverse Problems of High-Dimensional Estimation*, ed. P. Alguier, E. Gautier, and G. Stoltz, 121–156. Berlin: Springer.

Belloni, A., V. Chernozhukov, and C. B. Hansen. 2014a. High-dimensional methods and inference on structural and treatment effects. *Journal of Economic Perspectives* 28: 29–50. https://doi.org/10.1257/jep.28.2.29.

——. 2014b. Inference on treatment effects after selection among high-dimensional controls. *Review of Economic Studies* 81: 608–650. https://doi.org/10.1093/restud/rdt044.

Belloni, A., V. Chernozhukov, and Y. Wei. 2016. Post-selection inference for generalized linear models with many controls. *Journal of Business & Economic Statistics* 34: 606–619. https://doi.org/10.1080/07350015.2016.1166116.

Bickel, P. J., Y. Ritov, and A. B. Tsybakov. 2009. Simultaneous analysis of Lasso and Dantzig selector. *Annals of Statistics* 37: 1705–1732. https://doi.org/10.1214/08-AOS620.

Chernozhukov, V., D. Chetverikov, M. Demirer, E. Duflo, C. B. Hansen, W. K. Newey, and J. M. Robins. 2018. Double/debiased machine learning for treatment and structural parameters. *Econometrics Journal* 21: C1–C68. https://doi.org/10.1111/ectj.12097.

Chernozhukov, V., C. B. Hansen, and M. Spindler. 2015. Post-selection and post-regularization inference in linear models with many controls and instruments. *American Economic Review* 105: 486–490. https://doi.org/10.1257/aer.p20151022.

Drukker, D. M., and D. Liu. 2019. Using the lasso for inference in high-dimensional models. *The Stata Blog: Not Elsewhere Classified*. https://blog.stata.com/2019/09/09/using-the-lasso-for-inference-in-high-dimensional-models/.

Leeb, H., and B. M. Pötscher. 2005. Model selection and inference: Facts and fiction. *Econometric Theory* 21: 21–59. https://doi.org/10.1017/S0266466605050036.

——. 2006. Can one estimate the conditional distribution of post-model-selection estimators? *Annals of Statistics* 34: 2554–2591. https://doi.org/10.1214/009053606000000821.

——. 2008. Sparse estimators and the oracle property, or the return of Hodges' estimator. *Journal of Econometrics* 142: 201–211. https://doi.org/10.1016/j.jeconom.2007.05.017.

Wooldridge, J. M. 2020. *Introductory Econometrics: A Modern Approach*. 7th ed. Boston: Cengage.

Also see

[LASSO] **Lasso intro** — Introduction to lasso