Title

stata.com

lassogof — Goodness of fit after lasso for prediction

Description | Quick start | Menu | Syntax
Options | Remarks and examples | Stored results | Methods and formulas
References | Also see

Description

`lassogof` calculates goodness of fit of predictions after `lasso`, `sqrtlasso`, and `elasticnet`. It also calculates goodness of fit after `regress`, `logit`, `probit`, `poisson`, and `stcox` estimations for comparison purposes. For linear models, mean squared error of the prediction and R^2 are displayed. For logit, probit, Poisson, and Cox models, deviance and deviance ratio are shown.

Quick start

See goodness of fit for current lasso result using penalized coefficient estimates

```
lassogof
```

See goodness of fit for current lasso result using postselection coefficient estimates

```
lassogof, postselection
```

See goodness of fit for four stored estimation results

```
lassogof mylasso mysqrtlasso myelasticnet myregress
```

See goodness of fit for all stored estimation results

```
lassogof *
```

Randomly split sample into two, fit a lasso on the first sample, and calculate goodness of fit separately for both samples

```
splitsample, generate(sample) nsplit(2)
lasso linear y x* if sample == 1
lassogof, over(sample)
```

Menu

Statistics > Postestimation

Syntax

lassogof [*namelist*] [*if*] [*in*] [, *options*]

namelist is a name of a stored estimation result, a list of names, _all, or *. _all and * mean the same thing. See [R] **estimates store**.

options	Description
Main	
penalized	use penalized (shrunken) coefficient estimates; the default
postselection	use postselection coefficient estimates
over(*varname*)	display goodness of fit for samples defined by *varname*
noweights	do not use weights when calculating goodness of fit

collect is allowed; see [U] **11.1.10 Prefix commands**.

Options

Main

penalized specifies that the penalized coefficient estimates be used to calculate goodness of fit. Penalized coefficients are those estimated by lasso with shrinkage. This is the default.

postselection specifies that the postselection coefficient estimates be used to calculate goodness of fit. Postselection coefficients are estimated by taking the covariates selected by lasso and reestimating the coefficients using an unpenalized estimator—namely, an ordinary linear regression, logistic regression, probit model, Poisson regression, or Cox regression as appropriate.

over(*varname*) specifies that goodness of fit be calculated separately for groups of observations defined by the distinct values of *varname*. Typically, this option would be used when the lasso is fit on one sample and one wishes to compare the fit in that sample with the fit in another sample.

noweights specifies that any weights used to estimate the lasso be ignored in the calculation of goodness of fit.

Remarks and examples

lassogof is intended for use on out-of-sample data. That is, on data different from the data used to fit the lasso.

There are two ways to do this. One is to randomly split your data into two subsamples before fitting a lasso model. The examples in this entry show how to do this using splitsample.

The other way is to load a different dataset in memory and run lassogof with the lasso results on it. The steps for doing this are as follows.

1. Load the data on which you are going to fit your model.

```
. use datafile1
```

2. Run lasso (or sqrtlasso or elasticnet).

```
. lasso ...
```

3. Save the results in a file.

    . estimates save *filename*

4. Load the data for testing the prediction.

    . use *datafile2*, clear

5. Load the saved results, making them the current (active) estimation results.

    . estimates use *filename*

6. Run `lassogof`.

    . lassogof

▷ Example 1: Comparing fit in linear models

We will show how to use `lassogof` after `lasso linear`.

Here is an example using `lasso` from [LASSO] **lasso examples**. We load the data and make the `vl` variable lists active.

```
. use https://www.stata-press.com/data/r18/fakesurvey_vl
(Fictitious survey data with vl)
. vl rebuild
Rebuilding vl macros ...
  (output omitted)
```

We now use `splitsample` to generate a variable indicating the two subsamples.

```
. set seed 1234
. splitsample, generate(sample) nsplit(2)
. label define svalues 1 "Training" 2 "Testing"
. label values sample svalues
```

We run lasso on the first subsample and set the random-number seed using the `rseed()` option so we can reproduce our results.

```
. lasso linear q104 ($idemographics) $ifactors $vlcontinuous
> if sample == 1, rseed(1234)
  (output omitted)
Lasso linear model                          No. of obs         =        458
                                            No. of covariates  =        277
Selection: Cross-validation                 No. of CV folds    =         10
```

ID	Description	lambda	No. of nonzero coef.	Out-of-sample R-squared	CV mean prediction error
1	first lambda	.8978025	4	0.0147	16.93341
18	lambda before	.1846342	42	0.2953	12.10991
* 19	selected lambda	.1682318	49	0.2968	12.08516
20	lambda after	.1532866	55	0.2964	12.09189
23	last lambda	.1159557	74	0.2913	12.17933

```
* lambda selected by cross-validation.
. estimates store linearcv
```

After the command finished, we used `estimates store` to store the results in memory so we can later compare these results with those from other lassos.

We are now going to run an adaptive lasso, which we do by specifying the option `selection(adaptive)`.

```
. lasso linear q104 ($idemographics) $ifactors $vlcontinuous
> if sample == 1, rseed(4321) selection(adaptive)
  (output omitted)
Lasso linear model                          No. of obs        =        458
                                            No. of covariates =        277
Selection: Adaptive                         No. of lasso steps =         2
Final adaptive step results
```

ID	Description	lambda	No. of nonzero coef.	Out-of-sample R-squared	CV mean prediction error
25	first lambda	48.55244	4	0.0101	17.01083
77	lambda before	.3847698	46	0.3985	10.33691
* 78	selected lambda	.3505879	46	0.3987	10.33306
79	lambda after	.3194427	47	0.3985	10.33653
124	last lambda	.0048552	59	0.3677	10.86697

```
* lambda selected by cross-validation in final adaptive step.
. estimates store linearadaptive
```

We want to see which performs better for out-of-sample prediction. We specify the `over()` option with the name of our sample indicator variable, `sample`. We specify the `postselection` option because for linear models, postselection coefficients are theoretically slightly better for prediction than the penalized coefficients (which `lassogof` uses by default). See the discussion in *predict* in [LASSO] **lasso postestimation**.

```
. lassogof linearcv linearadaptive, over(sample) postselection
Postselection coefficients
```

Name	sample	MSE	R-squared	Obs
linearcv				
	Training	8.652771	0.5065	503
	Testing	14.58354	0.2658	493
linearadaptive				
	Training	8.637575	0.5057	504
	Testing	14.70756	0.2595	494

The ordinary lasso did a little better in this case than adaptive lasso.

◁

▷ Example 2: Comparing fit in logit and probit models

We fit a logit model on the same data we used in the previous example.

```
. lasso logit q106 $idemographics $ifactors $vlcontinuous
> if sample == 1, rseed(1234)
```

(output omitted)

```
Lasso logit model                         No. of obs        =        458
                                          No. of covariates =        277
Selection: Cross-validation               No. of CV folds   =         10
```

ID	Description	lambda	No. of nonzero coef.	Out-of-sample dev. ratio	CV mean deviance
1	first lambda	.1155342	0	-0.0004	1.384878
22	lambda before	.0163767	65	0.1857	1.127315
* 23	selected lambda	.0149218	69	0.1871	1.125331
24	lambda after	.0135962	73	0.1864	1.126333
27	last lambda	.010285	88	0.1712	1.147343

```
* lambda selected by cross-validation.
. estimates store logit
```

Let's now fit a probit model.

```
. lasso probit q106 $idemographics $ifactors $vlcontinuous
> if sample == 1, rseed(1234)
```

(output omitted)

```
Lasso probit model                        No. of obs        =        458
                                          No. of covariates =        277
Selection: Cross-validation               No. of CV folds   =         10
```

ID	Description	lambda	No. of nonzero coef.	Out-of-sample dev. ratio	CV mean deviance
1	first lambda	.1844415	0	-0.0004	1.384877
21	lambda before	.0286931	61	0.1820	1.132461
* 22	selected lambda	.0261441	64	0.1846	1.128895
23	lambda after	.0238215	70	0.1841	1.129499
26	last lambda	.0180201	87	0.1677	1.152188

```
* lambda selected by cross-validation.
. estimates store probit
```

We look at how they did for out-of-sample prediction.

```
. lassogof logit probit, over(sample)
Penalized coefficients
```

Name	sample	Deviance	Deviance ratio	Obs
logit				
	Training	.8768969	0.3674	499
	Testing	1.268346	0.0844	502
probit				
	Training	.8833892	0.3627	500
	Testing	1.27267	0.0812	503

They both did not do very well. The out-of-sample deviance ratios were notably worse than the in-sample values. The deviance ratio for nonlinear models is analogous to R^2 for linear models. See *Methods and formulas* for the formal definition.

We did not specify the `postselection` option in this case because there are no theoretical grounds for using postselection coefficients for prediction with nonlinear models.

◁

Stored results

`lassogof` stores the following in `r()`:

Macros

`r(names)`	names of estimation results displayed
`r(over_var)`	name of the `over()` variable
`r(over_levels)`	levels of the `over()` variable

Matrices

`r(table)`	matrix containing the values displayed

Methods and formulas

`lassogof` reports the mean squared error (MSE) and the R^2 measures of fit for linear models. It reports the deviance and the deviance ratio for `logit`, `probit`, `poisson`, and `cox` models. The deviance ratio is also known as D^2 in the literature.

See Wooldridge (2020, 720) for more about MSE and Wooldridge (2020, 76–77) for more about R^2. The deviance measures are described in Hastie, Tibshirani, and Wainwright (2015, 29–33) and McCullagh and Nelder (1989, 33–34). For the `cox` model deviance, see Simon, Friedman, Hastie, and Tibshirani (2011).

In the formulas below, we use $\texttt{xb}_i$ to denote the linear prediction for the ith observation. By default, the lasso penalized coefficients $\widehat{\beta}$ are used to compute $\texttt{xb}_i$. Specifying the option `postselection` causes the postselection estimates $\widehat{\widehat{\beta}}$ to be used to compute $\texttt{xb}_i$. See *predict* in [LASSO] **lasso postestimation** for a discussion of penalized estimates and postselection estimates.

We also use the following notation. y_i denotes the ith observation of the outcome. w_i is the weight applied to the ith observation; $w_i = 1$ if no weights were specified in the estimation command or if option `noweights` was specified in `lassogof`. N is the number of observations in the sample over which the goodness-of-fit statistics are computed. If frequency weights were specified at estimation $N_s = \sum_{i=1}^{N} w_i$; otherwise, $N_s = N$.

The formulas for the measures reported after linear models are

$$R^2 = 1 - \text{RSS}/\text{TSS}$$

$$\text{MSE} = 1/N_s\text{RSS}$$

where

$$\text{RSS} = \sum_{i=1}^{N} w_i (y_i - \texttt{xb}_i)^2$$

$$\text{TSS} = \sum_{i=1}^{N} w_i (y_i - \overline{y})^2$$

$$\overline{y} = \frac{1}{N_s} \sum_{i=1}^{N} w_i y_i$$

The deviance ratio D^2 is given by

$$D^2 = \frac{D_{\text{null}} - D}{D_{\text{null}}}$$

where D_{null} is the deviance calculated when only a constant term is included in the model and D is the deviance of the full model.

The formulas for the deviance and for D_{null} vary by model.

For `logit`, the deviance and the D_{null} are

$$D = -\frac{2}{N_s} \sum_{i=1}^{N} w_i \left[\widetilde{y}_i \texttt{xb}_i + \ln\{1 + \exp(\texttt{xb}_i)\} \right]$$

$$D_{\text{null}} = -\frac{2}{N_s} \sum_{i=1}^{N} w_i \{ \widetilde{y}_i \ln \overline{y} + (1 - \widetilde{y}_i) \ln(1 - \overline{y}) \}$$

$$\widetilde{y}_i = \begin{cases} 1 & y_i > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$\overline{y} = \frac{1}{N_s} \sum_{i=1}^{N} w_i \widetilde{y}_i$$

For `probit`, the deviance and the D_{null} are

$$D = -\frac{2}{N_s}\sum_{i=1}^{N} w_i \left[\widetilde{y}_i \ln\{\Phi(\texttt{xb}_i)\} + (1-\widetilde{y}_i)\ln\{1-\Phi(\texttt{xb}_i)\}\right]$$

$$D_{\text{null}} = -\frac{2}{N_s}\sum_{i=1}^{N} w_i \{\widetilde{y}_i \ln \overline{y} + (1-\widetilde{y}_i)\ln(1-\overline{y})\}$$

$$\widetilde{y}_i = \begin{cases} 1 & y_i > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$\overline{y} = \frac{1}{N_s}\sum_{i=1}^{N} s_i w_i \widetilde{y}_i$$

For `poisson`, the deviance and the D_{null} are

$$D = -\frac{2}{N_s}\sum_{i=1}^{N} w_i \{y_i \texttt{xb}_i - \exp(\texttt{xb}_i) - v_i\}$$

$$v_i = \begin{cases} 0 & \text{if } y_i = 0 \\ y_i \ln y_i - y_i & \text{otherwise} \end{cases}$$

$$D_{\text{null}} = -\frac{2}{N_s}\sum_{i=1}^{N} w_i (y_i \ln \overline{y} - \overline{y} - v_i)$$

$$\overline{y} = \frac{1}{N_s}\sum_{i=1}^{N} w_i y_i$$

For cox, the deviance and the D_{null} are

$$D = 2\left(l_{\text{saturated}} - l\right)$$

$$D_{\text{null}} = 2\left(l_{\text{saturated}} - l_{\text{null}}\right)$$

$$l_{\text{saturated}} = -\frac{1}{N_s}\sum_{j=1}^{N_f} d_j \log\left(d_j\right)$$

$$l = -\frac{1}{N_s}\sum_{j=1}^{N_f}\sum_{i\in D_j}\left[w_i(\mathtt{xb}_i) - w_i \log\left\{\sum_{\ell\in R_j} w_\ell \exp(\mathtt{xb}_\ell)\right\}\right]$$

$$l_{\text{null}} = -\frac{1}{N_s}\sum_{j=1}^{N_f} d_j \log\left(\sum_{i\in R_j} w_i\right)$$

$$d_j = \sum_{i\in D_j} w_i$$

where j indexes the ordered failure times $t_{(j)}$, $j = 1, \dots, N_f$; D_j is the set of observations that fail at $t_{(j)}$; R_j is the set of observations k that are at risk at time $t_{(j)}$ (that is, all k such that $t_{0k} < t_{(j)} \le t_k$, and t_{0k} is the entry time for the kth observation).

References

Hastie, T. J., R. J. Tibshirani, and M. Wainwright. 2015. *Statistical Learning with Sparsity: The Lasso and Generalizations*. Boca Raton, FL: CRC Press.

McCullagh, P., and J. A. Nelder. 1989. *Generalized Linear Models*. 2nd ed. London: Chapman and Hall/CRC.

Simon, N., J. H. Friedman, T. J. Hastie, and R. J. Tibshirani. 2011. Regularization paths for Cox's proportional hazards model via coordinate descent. *Journal of Statistical Software* 39: 1–13. https://doi.org/10.18637/jss.v039.i05.

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

Also see

[LASSO] **lasso** — Lasso for prediction and model selection

[LASSO] **lassoknots** — Display knot table after lasso estimation

[LASSO] **lasso postestimation** — Postestimation tools for lasso for prediction