h2omlgof — Compare goodness of fit for machine learning models+

+This command includes features that are part of StataNow.

Description

h2omlgof reports goodness of fit after the h2oml *rf* and h2oml *gbm* commands. This command creates a table with side-by-side performance metrics from selected machine learning methods or models for easy comparison.

Quick start

Goodness of fit for comparing stored estimation results myrf and mygbm

```
h2omlgof myrf mygbm
```

Goodness-of-fit for comparing all stored estimation results using H2O frame mynewframe

```
h2omlgof *, frame(mynewframe)
```

Menu

Statistics > H2O machine learning

Syntax

`h2omlgof` *namelist* [, *options*]

namelist is a name of a stored estimation result, a list of names, `_all`, or `*`. `_all` or `*` requests all stored results. See [H2OML] **h2omlest**.

options	Description
Main	
`title(`*string*`)`	specify the title to be displayed above the table
`train`	specify that performance metrics be reported using training results
`valid`	specify that performance metrics be reported using validation results
`cv`	specify that performance metrics be reported using cross-validation results
`test`	specify that performance metrics be computed using the testing frame
`test(`*framename*`)`	specify that performance metrics be computed using data in testing frame *framename*
`frame(`*framename*`)`	specify that performance metrics be computed using data in H2O frame *framename*
`framelabel(`*string*`)`	label frame as *string* in the output

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

`train`, `valid`, `cv`, `test`, `test()`, `frame()`, and `framelabel()` do not appear in the dialog box.

Options

Main

`title(`*string*`)` specifies the title to be displayed above the table.

The following options are available with `h2omlgof` but are not shown in the dialog box:

`train`, `valid`, `cv`, `test`, `test()`, and `frame()` specify the H2O frame for which performance metrics are reported. Only one of `train`, `valid`, `cv`, `test`, `test()`, or `frame()` is allowed.

`train` specifies that performance metrics be reported using training results. This is the default when neither validation nor cross-validation is performed during estimation and when a postestimation frame has not been set with `h2omlpostestframe`.

`valid` specifies that performance metrics be reported using validation results. This is the default when validation is performed during estimation and when a postestimation frame has not been set with `h2omlpostestframe`. `valid` may be specified only when the `validframe()` option is specified with `h2oml` *gbm* or `h2oml` *rf*.

`cv` specifies that performance metrics be reported using cross-validation results. This is the default when cross-validation is performed during estimation and when a postestimation frame has not been set with `h2omlpostestframe`. `cv` may be specified only when the `cv` or `cv()` option is specified with `h2oml` *gbm* or `h2oml` *rf*.

test specifies that performance metrics be computed on the testing frame specified with h2omlpostestframe. This is the default when a testing frame is specified with h2omlpostestframe. test may be specified only after a testing frame is set with h2omlpostestframe. test is necessary only when a subsequent h2omlpostestframe command is used to set a default postestimation frame other than the testing frame.

test(*framename*) specifies that performance metrics be computed using data in testing frame *framename* and is rarely used. This option is most useful when running a single postestimation command on the named frame. If multiple postestimation commands are to be run on the same test frame, h2omlpostestframe provides a more convenient and computationally efficient process for doing this.

frame(*framename*) specifies that performance metrics be computed using the data in H2O frame *framename*.

framelabel(*string*) specifies the label to be used for the frame in the output. This option is not allowed with the cv option.

stata.com

Remarks and examples

The h2omlgof command provides a concise table of performance metrics for comparing different machine learning methods or models.

After h2oml gbregress and h2oml rfregress, h2omlgof reports the deviance, mean squared error (MSE), root mean squared error (RMSE), root mean squared logarithmic error (RMSLE), mean absolute error (MAE), and R^2. After h2oml gbbinclass and h2oml rfbinclass, it reports log loss, mean of per-class error rates, area under the curve (AUC), area under the precision–recall curve (AUCPR), Gini coefficient, MSE, and RMSE. Finally, after h2oml gbmulticlass and h2oml rfmulticlass, it reports log loss, mean of per-class error rates, MSE, and RMSE. See [H2OML] ***metric_option*** for more information on the reported metrics.

▷ Example 1: Comparing performance in H2OML

In this example, we use h2omlgof to compare results of h2oml *rf* and h2oml *gbm*.

We start by opening the 1978 automobile data (auto.dta) in Stata and then putting the data into an H2O frame. Recall that h2o init initiates an H2O cluster, _h2oframe put loads the current Stata dataset into an H2O frame, and _h2oframe change makes the specified frame the current H2O frame. We then use the _h2oframe split command to randomly split the auto frame into a training frame (70% of observations), a validation frame (20% of observations), and a testing frame (10% of observations), which we name train, valid, and test, respectively. We also change the current frame to train. For details, see *Prepare your data for H2O machine learning in Stata* in [H2OML] **h2oml** and [H2OML] **H2O setup**.

```
. use https://www.stata-press.com/data/r18/auto
(1978 automobile data)
. h2o init
  (output omitted)
. _h2oframe _put, into(auto)
Progress (%): 0 100
. _h2oframe split auto, into(train valid test) split(0.7 0.2 0.1) rseed(19)
. _h2oframe change train
```

We perform random forest binary classification with default values, and we specify the validation frame in the `validframe()` option. We store the estimation results by using the `h2omlest store` command.

```
. h2oml rfbinclass foreign price length weight, validframe(valid)
> h2orseed(19)
Progress (%): 0 60.0 100
Random forest binary classification using H2O
Response: foreign
Frame:                                  Number of observations:
  Training:   train                                Training =     57
  Validation: valid                              Validation =     10
Model parameters
Number of trees      =  50
              actual =  50
Tree depth:                             Pred. sampling value =     -1
           Input max =  20              Sampling rate        =   .632
                 min =   3              No. of bins cat.     =  1,024
                 avg = 5.7              No. of bins root     =  1,024
                 max =   8              No. of bins cont.    =     20
Min. obs. leaf split =   1              Min. split thresh.   = .00001
Metric summary
```

Metric	Training	Validation
Log loss	.8466057	.3177202
Mean class error	.0625	.1666667
AUC	.9235294	.9047619
AUCPR	.6822189	.8512376
Gini coefficient	.8470588	.8095238
MSE	.0948292	.11421
RMSE	.3079434	.3379497

```
. h2omlest store RF
```

Next we perform gradient boosting binary classification and store the estimation results.

```
. h2oml gbbinclass foreign price length weight, validframe(valid)
> h2orseed(19)
Progress (%): 0 100
Gradient boosting binary classification using H2O
Response: foreign
Loss:     Bernoulli
Frame:                                  Number of observations:
  Training:   train                              Training =     57
  Validation: valid                            Validation =     10
Model parameters
Number of trees      =  50              Learning rate       =     .1
              actual =  50              Learning rate decay =      1
Tree depth:                             Pred. sampling rate =      1
           Input max =   5              Sampling rate       =      1
                 min =   2              No. of bins cat.    =  1,024
                 avg = 2.9              No. of bins root    =  1,024
                 max =   4              No. of bins cont.   =     20
Min. obs. leaf split =  10              Min. split thresh.  = .00001
```

```
Metric summary

-----------------------------------------
           Metric |  Training  Validation
------------------+----------------------
         Log loss |  .1072901    .2774807
Mean class error  |     .0125    .0714286
              AUC |  .9955882     .952381
           AUCPR  |  .9889171     .904106
 Gini coefficient |  .9911765    .9047619
              MSE |  .0261993    .1002502
             RMSE |   .161862    .3166232
-----------------------------------------

. h2omlest store GBM
```

To compare random forest (RF) and gradient boosting machine (GBM) models, we type

```
. h2omlgof RF GBM

Performance metrics for model comparison using H2O
Training frame:   train
Validation frame: valid

---------------------------------------------------
                       |          RF          GBM
-----------------------+---------------------------
Training               |
  No. of observations  |          57           57
             Log loss  |    .8466057     .1072901
     Mean class error  |       .0625        .0125
                  AUC  |    .9235294     .9955882
                AUCPR  |    .6822189     .9889171
     Gini coefficient  |    .8470588     .9911765
                  MSE  |    .0948292     .0261993
                 RMSE  |    .3079434      .161862
-----------------------+---------------------------
Validation             |
  No. of observations  |          10           10
             Log loss  |    .3177202     .2774807
     Mean class error  |    .1666667     .0714286
                  AUC  |    .9047619      .952381
                AUCPR  |    .8512376      .904106
     Gini coefficient  |    .8095238     .9047619
                  MSE  |      .11421     .1002502
                 RMSE  |    .3379497     .3166232
---------------------------------------------------
```

In the output, the first section reports training results, and the second section reports validation results. Looking at the validation results, we see that the GBM method outperforms the RF method. The log loss, mean of per-class error rates, MSE, and RMSE are all smaller for GBM, while AUC, AUCPR, and the Gini coefficient are larger for GBM, all of which indicate better performance.

◁

▷ Example 2: Comparing performance in H2OML on a new frame

In example 1, we compared the performance of two methods on the validation frame. If we instead wish to compare methods on a new data frame, we can take one of two approaches. In the first, we specify the frame in the `frame()` option or, if it is a testing frame, in the `test()` option.

```
. h2omlgof RF GBM, test(test)

Performance metrics for model comparison using H2O
Testing frame: test

─────────────────────────────────────────────
                      │        RF        GBM
──────────────────────┼──────────────────────
Testing               │
  No. of observations │         7          7
             Log loss │   .236301   .1155489
     Mean class error │         0          0
                  AUC │         1          1
                AUCPR │         1          1
     Gini coefficient │         1          1
                  MSE │  .0878302   .0364771
                 RMSE │  .2963615   .1909897
─────────────────────────────────────────────
```

In the second approach, which we recommend, we use the `h2omlpostestframe` command to specify the postestimation frame to be used by this and other postestimation commands. With this approach, the new frame must be set for each set of estimation results. Thus, we first need to restore each set of estimates by using the `h2omlest restore` command. For the GBM results, we type

```
. h2omlest restore GBM
(results GBM are active now)

. h2omlpostestframe test
(testing frame test is now active for h2oml postestimation)
```

Similarly, for the RF results, we type

```
. h2omlest restore RF
(results RF are active now)

. h2omlpostestframe test
(testing frame test is now active for h2oml postestimation)
```

Finally, we compare the testing results by using the `h2omlgof` command.

```
. h2omlgof RF GBM

Performance metrics for model comparison using H2O
Testing frame: test

─────────────────────────────────────────────
                      │        RF        GBM
──────────────────────┼──────────────────────
Testing               │
  No. of observations │         7          7
             Log loss │   .236301   .1155489
     Mean class error │         0          0
                  AUC │         1          1
                AUCPR │         1          1
     Gini coefficient │         1          1
                  MSE │  .0878302   .0364771
                 RMSE │  .2963615   .1909897
─────────────────────────────────────────────
```

Here GBM again outperforms RF for most of the performance metrics.

◁

Stored results

h2omlgof stores the following in r():

Macros	
r(names)	names of estimation results displayed
Matrices	
r(table)	matrix containing the values displayed

Also see

[H2OML] **h2oml** — Introduction to commands for Stata integration with H2O machine learning+

[H2OML] **h2omlestat metrics** — Display performance metrics+

For suggested citations, see the FAQ on citing Stata documentation.