

Description

h2omlestat gridsummary displays the grid summary for configurations of hyperparameters after h2oml gbm and h2oml rf perform tuning using a grid search.

When tuning is performed, the h2oml gbm and h2oml rf commands report performance metrics for the best model based on the tuning metric. h2omlestat gridsummary reports the tuning metric or another specified metric for additional models that were evaluated as part of the grid search. It also assigns an ID number to each model. You can then specify these ID numbers in h2omlexplore to compare a variety of performance metrics for the chosen models. You can also use h2omlselect to select a model based on the ID number so that subsequent postestimation commands will be based on this model instead of the one selected by tuning h2oml gbm or h2oml rf.

Quick start

Display the grid summary of log-loss metrics after h2oml gbbinclass

```
h2oml gbbinclass y x2-x5, ntrees(50(5)80) tune(grid(cartesian))
h2omlestat gridsummary
```

Same as above, but report the grid summary for the area under the curve (AUC) metric

```
h2omlestat gridsummary, metric(auc)
```

Menu

Statistics > H2O machine learning

Syntax

```
h2omlestat gridsummary [ , options ]
```

| options | Description |
|---------------------------------|--|
| <u>metric</u> (<i>metric</i>) | specify the metric to be reported |
| <u>top</u> (#) | report the top # models; top(_all) reports all models; default is top(10) |
| <u>title</u> (<i>string</i>) | specify title to be displayed above the table |

Options

`metric(metric)` specifies the metric for which the grid summary will be reported. Allowed metrics are provided in [\[H2OML\] *metric_option*](#). If the `metric()` suboption is specified in the `tune()` option of the `h2oml gbm` or `h2oml rf` command, then `h2omlestat gridsummary` will use the same metric. Otherwise, the default metric is deviance for regression and log loss for classification.

`top(#)` specifies that the top # models be included in the summary table. `top(_all)` specifies that all models be reported. The default is `top(10)`.

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

Remarks and examples

To build a machine learning model that generalizes well to new data involves choosing an appropriate method and selecting a model by tuning hyperparameters; see [Hyperparameter tuning](#) in [\[H2OML\] Intro](#) for more information on tuning. For example, suppose we want to perform gradient boosting binary classification and use an exhaustive grid search to select the optimal number of trees. We could type

```
h2oml gbbinclass y x1-x100, ntrees(10(5)100)
```

We can use `h2omlestat gridsummary` to report the models ranked based on the default log-loss tuning metric.

```
h2omlestat gridsummary
```

Alternatively, we can request a grid summary for another metric, such as the AUC.

```
h2omlestat gridsummary, metric(auc)
```

After reporting the grid-search summary, we can compare models with different hyperparameters based on other performance metrics by using the `h2omlexplore` command; we select the desired model by using the `h2omlselect` command. See [\[H2OML\] *h2omlexplore*](#) and [\[H2OML\] *h2omlselect*](#) for examples demonstrating how to use `h2omlestat gridsummary` in combination with these commands.

▷ Example 1: Sequential hyperparameter tuning

When the dataset is large and there are many hyperparameters, tuning these hyperparameters simultaneously can be computationally intensive. We can reduce the computational burden by tuning hyperparameters sequentially. That is, in the first iteration of tuning, a small set of hyperparameters are tuned to narrow the search space. Then in the second iteration, the best results from the previous iteration can be used with additional hyperparameters. However, note that this procedure might lead us to select suboptimal values for the hyperparameters, and it is only recommended for large datasets. As an alternative, which also may result in a suboptimal solution, one could use a random grid search and restrict the search space by specifying the `maxmodels()` or `maxtime()` suboption in the `tune()` option of the `h2oml gbm` or `h2oml rf` command.

In this example, we use [gradient boosting](#) to illustrate the sequential procedure.

We begin by opening the `auto.dta` dataset in Stata and then putting it 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. For details, see [Prepare your data for H2O machine learning in Stata](#) in [\[H2OML\] *h2oml*](#) and see [\[H2OML\] H2O setup](#).

```
. use https://www.stata-press.com/data/r19/auto
(1978 automobile data)

. h2o init
(output omitted)

. _h2oframe put, into(auto)
Progress (%): 0 100

. _h2oframe change auto
```

In the first step of our tuning procedure, we tune the maximum depth of the trees hyperparameter using 3-fold cross-validation and an exhaustive grid search. We set the learning rate to 0.05, a little higher than the recommended 0.01, because the learning rate decay is 0.9. For details on gradient boosting machine hyperparameters, see [H2OML] *h2oml gbm*.

```
. h2oml gbbinclass foreign price mpg weight length, cv(3, modulo) h2orseed(19)
> lratedecay(0.9) lrate(0.05) maxdepth(1(1)10) tune(grid(cartesian))
Progress (%): 0 100
Gradient boosting binary classification using H2O
Response: foreign
Loss:      Bernoulli
Frame:
  Training: auto
Number of observations:
  Training =      74
  Cross-validation = 74
Cross-validation: Modulo
Number of folds      =      3
Tuning information for hyperparameters
Method: Cartesian
Metric: Log loss
```

| Hyperparameters | Grid values | | |
|-----------------|-------------|---------|----------|
| | Minimum | Maximum | Selected |
| Max. tree depth | 1 | 10 | 10 |

Model parameters

```
Number of trees      = 50
actual              = 50
Learning rate        = .05
Learning rate decay = .9
Tree depth:
  Pred. sampling rate = 1
  Input max          = 10
  min                = 2
  avg                = 3.0
  max                = 4
  Sampling rate       = 1
  No. of bins cat.    = 1,024
  No. of bins root    = 1,024
  No. of bins cont.   = 20
  Min. obs. leaf split = 10
  Min. split thresh. = .00001
```

Metric summary

| Metric | Cross- | |
|------------------|----------|------------|
| | Training | validation |
| Log loss | .3679234 | .4914566 |
| Mean class error | .0576923 | .1958042 |
| AUC | .9820804 | .8535839 |
| AUCPR | .9584095 | .6989351 |
| Gini coefficient | .9641608 | .7071678 |
| MSE | .1063068 | .159142 |
| RMSE | .3260472 | .398926 |

Next we use `h2omlestat gridsummary` to report the configurations that achieve the best performance based on the log-loss metric.

```
. h2omlestat gridsummary
Grid summary using H2O
```

| ID | Max. tree depth | Log loss |
|----|--------------------|----------|
| 1 | 10 | .4914566 |
| 2 | 3 | .4914566 |
| 3 | 4 | .4914566 |
| 4 | 5 | .4914566 |
| 5 | 6 | .4914566 |
| 6 | 7 | .4914566 |
| 7 | 8 | .4914566 |
| 8 | 9 | .4914566 |
| 9 | 2 | .4919681 |
| 10 | 1 | .5266221 |

We see that the performance of the model in terms of the log-loss metric does not change for maximum tree depths between 3 and 10. Therefore, to have a parsimonious model, we select a maximum tree depth of 3. In the second step of our tuning procedure, we specify the `maxdepth(3)` option and tune the learning rate and sampling rate hyperparameters.

```
. h2oml gbbinclass foreign price mpg weight length, cv(3, modulo) h2orseed(19)
> lratedecay(0.9) maxdepth(3) samprate(0.4(0.1)1) lrate(0.2(0.02)0.3)
> tune(grid(cartesian))
```

Progress (%): 0 100

Gradient boosting binary classification using H2O

Response: foreign

Loss: Bernoulli

Frame:

Number of observations:

Training: auto

Training = 74

Cross-validation = 74

Cross-validation: Modulo

Number of folds = 3

Tuning information for hyperparameters

Method: Cartesian

Metric: Log loss

| Hyperparameters | Grid values | | |
|-----------------|-------------|---------|----------|
| | Minimum | Maximum | Selected |
| Learning rate | .2 | .3 | .28 |
| Sampling rate | .4 | 1 | 1 |

Model parameters

```

Number of trees      = 50      Learning rate        = .28
                    actual = 50  Learning rate decay = .9
Tree depth:         Pred. sampling rate = 1
                    Input max = 3  Sampling rate        = 1
                        min = 2    No. of bins cat.       = 1,024
                        avg = 3.0  No. of bins root    = 1,024
                        max = 3    No. of bins cont.     = 20
Min. obs. leaf split = 10      Min. split thresh.  = .00001

```

Metric summary

| Metric | Cross- | |
|------------------|----------|------------|
| | Training | validation |
| Log loss | .1357221 | .2983633 |
| Mean class error | .0227273 | .090035 |
| AUC | .9982517 | .9370629 |
| AUCPR | .9961309 | .8555774 |
| Gini coefficient | .9965035 | .8741259 |
| MSE | .0326208 | .097178 |
| RMSE | .1806123 | .3117338 |

Once again, we use `h2omlestat gridsummary` to report the configurations that achieve the best performance based on the log-loss metric.

```
. h2omlestat gridsummary
```

```
Grid summary using H2O
```

| ID | Learning rate | Sampling rate | Log loss |
|----|---------------|---------------|----------|
| 1 | .28 | 1 | .2983633 |
| 2 | .3 | 1 | .2998373 |
| 3 | .24 | 1 | .3038322 |
| 4 | .26 | 1 | .3042715 |
| 5 | .28 | .9 | .3087905 |
| 6 | .3 | .9 | .3102182 |
| 7 | .22 | 1 | .3137784 |
| 8 | .26 | .9 | .3159972 |
| 9 | .24 | .9 | .3176375 |
| 10 | .28 | .7 | .3319306 |

We see that the top model achieved a log-loss of 0.298, and the corresponding hyperparameters are a learning rate of 0.28 and a sampling rate of 1.



Stored results

`h2omlestat gridsummary` stores the following in `r()`:

Matrices

`r(gridsummary)`

grid-search summary of hyperparameters and metrics

Also see

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

[H2OML] **h2omlexplore** — Explore models after grid search

[H2OML] **h2omlselect** — Select model after grid search

Stata, Stata Press, and Mata are registered trademarks of StataCorp LLC. Stata and Stata Press are registered trademarks with the World Intellectual Property Organization of the United Nations. StataNow and NetCourseNow are trademarks of StataCorp LLC. Other brand and product names are registered trademarks or trademarks of their respective companies. Copyright © 1985–2025 StataCorp LLC, College Station, TX, USA. All rights reserved.

For suggested citations, see the FAQ on [citing Stata documentation](#).

