h2omlselect — Select model after grid search+

+This command includes features that are part of StataNow.

Description Quick start Menu Syntax
Remarks and examples Stored results Also see

Description

h2omlselect retrieves the fitted model with the hyperparameter configuration you select after h2oml *gbm* and h2oml *rf* perform tuning using a grid search. These estimation commands select the top-performing model, the one with the most optimal tuning performance metric, as the working model. After estimation, you can use h2omlestat gridsummary to see performance metrics for models with different hyperparameter configurations and to obtain an ID for each of these models. You can then select a different model to be the working model by using h2omlselect. h2omlselect selects and retrieves the fitted model; afterward, you can treat this model just as you would treat estimation results from the h2oml *gbm* and h2oml *rf* estimation commands. Subsequent postestimation commands are based on the selected model.

Quick start

After performing multiclass classification and obtaining the grid-search summary, select the model that has id = 2

```
h2oml rfmulticlass y x1-x20, ntrees(10(5)100) maxdepth(3(1)10)
h2omlestat gridsummary
h2omlselect id = 2
```

Menu

Statistics > H2O machine learning

Syntax

```
h2omlselect id = #
```

where # is a grid ID from h2omlestat gridsummary corresponding to the desired model configuration.
stata.com

Remarks and examples

Building a machine learning model that generalizes well to new data involves choosing an appropriate method and selecting a model by tuning hyperparameters. We can perform a grid search using gradient boosting and random forest methods and then use h2omlestat gridsummary to report the hyperparameter configurations that achieve the top performance based on the specified metric. For example, you might use the log-loss metric to choose between models with 10, 20, and 30 trees. Typically, you would select the model that performs the best based on the chosen metric. However, you may want to explore different hyperparameter configurations that do not correspond to the best model, in which case you can use h2omlselect and h2omlexplore.

After you review the grid-search summary from h2omlestat gridsummary, you can select the model you are interested in by specifying the ID number with h2omlselect. Once you have selected a model with h2omlselect, you can treat the model in the same way you would treat results from the h2oml *gbm* and h2oml *rf* estimation commands. Postestimation commands will be based on the model selected by h2omlselect; for example, you could estimate variable importance for the selected model with h2omlgraph varimp. h2omlselect overwrites the previously stored estimation results, which can be recovered by refitting the original model or by storing the estimation results before running h2omlselect and then restoring them; see [H2OML] **h2omlest**.

▷ Example 1: Selecting the second-best model

In this example, we illustrate the use of h2omlselect by performing random forest binary classification with the social pressure dataset discussed in example 1 of [H2OML] ***h2oml rf***.

We start by opening the social pressure dataset 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 in an H2O frame, and _h2oframe change makes the specified frame the current H2O frame. We use the _h2oframe split command to randomly split the social frame into a training frame (80% of observations) and a validation frame (20% of observations), which we name train and valid, 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 see [H2OML] **H2O setup**.

```
. use https://www.stata-press.com/data/r18/socialpressure
(Social pressure data)

. h2o init
  (output omitted)

. _h2oframe _put, into(social)
Progress (%): 0 100

. _h2oframe _split social, into(train valid) split(0.8 0.2) rseed(19)

. _h2oframe _change train
```

We define a global macro, `predictors`, to store the names of our predictors. We perform random forest binary classification, and we specify the `maxdepth()` and `predsampvalue()` options to tune the maximum tree depth and predictor sampling rate hyperparameters. For illustration, we use the area under the precision–recall curve (AUCPR) metric for tuning.

```
. global predictors gender g2000 g2002 p2000 p2002 p2004 treatment age

. h2oml rfbinclass voted $predictors, validframe(valid) h2orseed(19)
> ntrees(200) maxdepth(3(3)12) predsampvalue(-1, 1(2)8) tune(metric(aucpr))

Progress (%): 0 100

Random forest binary classification using H2O

Response: voted
Frame:                                 Number of observations:
  Training:   train                                Training = 183,607
  Validation: valid                              Validation =  45,854

Tuning information for hyperparameters

Method: Cartesian
Metric: AUCPR

---------------------------------------------------------------------
                      |                    Grid values
      Hyperparameters |        Minimum         Maximum        Selected
----------------------+----------------------------------------------
      Max. tree depth |              3              12               6
 Pred. sampling value |             -1               7               7
---------------------------------------------------------------------

Model parameters

Number of trees      = 200
              actual = 200
Tree depth:                            Pred. sampling value =       7
           Input max =   6             Sampling rate        =    .632
                 min =   6             No. of bins cat.     =   1,024
                 avg = 6.0             No. of bins root     =   1,024
                 max =   6             No. of bins cont.    =      20
Min. obs. leaf split =   1             Min. split thresh.   =  .00001

Metric summary

------------------------------------------
           Metric |   Training  Validation
------------------+-----------------------
         Log loss |   .5724664    .5705699
 Mean class error |   .3935492    .3943867
              AUC |   .6705554    .6734867
            AUCPR |   .4658395    .4725543
 Gini coefficient |   .3411109    .3469735
              MSE |   .1946923    .1935647
             RMSE |   .4412395    .4399599
------------------------------------------
```

Next we obtain the grid-search summary by using the h2omlestat gridsummary command. This command lists the configuration of the hyperparameters we are tuning ranked by AUCPR.

```
. h2omlestat gridsummary

Grid summary using H2O

────────────────────────────────────

                    Pred.
     Max. tree   sampling
 ID      depth      value      AUCPR

────────────────────────────────────

  1          6          7   .4725543
  2          6          5   .4723736
  3          6          3   .4714554
  4          9          3   .4712076
  5          6         -1   .4708614
  6         12         -1   .4706606
  7          9         -1   .4705794
  8          9          5   .4689799
  9          9          7   .4682457
 10          9          1   .4674565
────────────────────────────────────
```

The top two models have very similar values of AUCPR, and they correspond to models with 7 and 5 randomly sampled predictors and a maximum tree depth of 6. As discussed in [H2OML] ***h2oml rf***, using a random sample of predictors improves the ability of the model to generalize to new data, compared with using the full set of predictors, because it introduces an additional randomness to the method. Therefore, we may prefer to continue our analysis with the second-best model.

To select the second-best model, we specify id = 2 in h2omlselect.

```
. h2omlselect id = 2

Random forest binary classification using H2O

Response: voted
Frame:                                 Number of observations:
  Training:   train                                Training = 183,607
  Validation: valid                              Validation =  45,854

Model parameters

Number of trees      = 200
              actual = 200
Tree depth:                            Pred. sampling value =       5
           Input max =   6             Sampling rate        =    .632
                 min =   6             No. of bins cat.     =   1,024
                 avg = 6.0             No. of bins root     =   1,024
                 max =   6             No. of bins cont.    =      20
Min. obs. leaf split =   1             Min. split thresh.   =  .00001

Metric summary
──────────────────┬───────────────────────
           Metric │   Training  Validation
──────────────────┼───────────────────────
         Log loss │     .57237    .5704978
 Mean class error │   .3979593    .3945857
              AUC │    .671146    .6737527
            AUCPR │   .4670326    .4723736
 Gini coefficient │    .342292    .3475054
              MSE │   .1946602    .1935627
             RMSE │   .4412031    .4399576
──────────────────┴───────────────────────
```

Now we can continue our analysis using the second-best model.

◁

Stored results

`h2omlselect` retrieves the selected fitted model and thus stores the same results as the estimation command used.

See *Stored results* in [H2OML] ***h2oml gbm*** or [H2OML] ***h2oml rf***.

Also see

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

[H2OML] **h2omlestat gridsummary** — Display grid-search summary+

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

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