h2omlgraph roc — Produce ROC curve plot

Description Quick start Menu Syntax
Options Remarks and examples Also see

Description

h2omlgraph roc plots the receiver operating characteristic (ROC) curve after binary classification performed by h2oml gbbinclass and h2oml rfbinclass. With binary classification, the predicted probability for each observation is compared with a threshold value to determine whether the observation is predicted to be in the positive class or the negative class. Thus, for different threshold values, different numbers of observations are classified as positive and negative. The ROC curve allows us to evaluate the tradeoff between the true-positive rate (TPR) and false-positive rate (FPR) by plotting these metrics for a variety of threshold values.

The curve produced by plotting TPR versus FPR is useful for evaluating model performance. A large area under the curve (AUC) indicates that the model has a high true-positive rate and low false-positive rate.

Quick start

Plot the ROC curve

```
h2omlgraph roc
```

Same as above, but report results based on the validation data

```
h2omlgraph roc, valid
```

Same as above, but remove the reference line

```
h2omlgraph roc, valid norefline
```

Menu

Statistics > H2O machine learning

Syntax

h2omlgraph roc [, *options*]

options	Description
Main	
models(*namelist*)	specify the name or a list of names of stored estimation results
savedata(*filename*[, replace])	save plot data to *filename*
Plot options	
rlopts(*line_options*)	affect rendition of reference line
norefline	suppress plotting reference line
lineopts(*line_options*)	affect rendition of all ROC curves
line#opts(*line_options*)	affect rendition of the ROC curve for model #
twoway_options	any options other than by() documented in [G-3] ***twoway_options***
train	specify that the TPR and FPR be reported using training results
valid	specify that the TPR and FPR be reported using validation results
cv	specify that the TPR and FPR be reported using cross-validation results
test	specify that the TPR and FPR be computed using the testing frame
test(*framename*)	specify that the TPR and FPR be computed using data in testing frame *framename*
frame(*framename*)	specify that the TPR and FPR be computed using data in H2O frame *framename*
framelabel(*string*)	label frame as *string* in the output

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

Options

Main

models(*namelist*) specifies the name or the list of the names of the stored estimation results for which the ROC curves are plotted. For each model, the displayed curve corresponds to the default frame of that model when a postestimation frame has not been set with h2omlpostestframe.

savedata(*filename*[, replace]) saves the plot data to a Stata data file (.dta file). replace specifies that filename be overwritten if it exists.

Plot options

rlopts(*line_options*) affects the rendition of the reference line. See [G-3] ***line_options***.

norefline suppresses plotting the reference line. The 45-degree reference line is the ROC curve that is expected if predictions are a random guess. The area between the ROC curve for the model and the reference line indicates how much better the model performs over a random guess.

lineopts(*line_options*) affects the rendition of all ROC curves. See [G-3] ***line_options***.

`line#opts(`*line_options*`)` affects the rendition of the ROC curve for model #. See [G-3] ***line_options***.

twoway_options are any of the options documented in [G-3] ***twoway_options***, excluding `by()`. These include options for titling the graph (see [G-3] ***title_options***) and options for saving the graph to disk (see [G-3] ***saving_option***).

The following options are available with `h2omlgraph roc` but are not shown in the dialog box:

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

`train` specifies that TPR and FPR 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 TPR and FPR 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 TPR and FPR 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 TPR and FPR 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 TPR and FPR 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 TPR and FPR 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.

Remarks and examples

ROC curves graphically illustrate how well a model performs in terms of the TPR and FPR.

After binary classification, the predicted probability for each observation is compared with a threshold value to determine whether the observation is predicted to be in the positive class or the negative class. Observations with probabilities greater than the threshold are classified as positive, and the remaining observations are classified as negative. Different threshold values lead to different predicted classes. Therefore, as the threshold changes, the numbers of true positives and false positives also change.

The ROC curve plots the TPR on the y axis and FPR on the x axis, where each metric is computed across a range of threshold values. This is useful for evaluating model performance. When the area under the ROC curve is large (close to 1), the model has a high TPR and low FPR.

▷ Example 1: Basic example

To best understand the ROC curve, we can find it helpful to first consider the TPR and FPR for individual threshold values. Below, we use the `h2omlestat threshmetric` command to obtain these metrics for three different threshold values.

```
. h2omlestat threshmetric, threshold(0)

Metrics for specific threshold using H2O
Training frame: auto

──────────────────────────────────────
Threshold                 │
                    Input │          0
                 Computed │          0
──────────────────────────┼───────────
Metric                    │
                       F1 │      .4583
                       F2 │       .679
                     F0.5 │      .3459
                 Accuracy │      .2973
                Precision │      .2973
                   Recall │          1
              Specificity │          0
      Min. class accuracy │          0
      Mean class accuracy │         .5
           True negatives │          0
          False negatives │          0
           True positives │         22
          False positives │         52
       True-negative rate │          0
      False-negative rate │          0
       True-positive rate │          1
      False-positive rate │          1
                      MCC │          0
──────────────────────────────────────
```

A threshold of 0 produces a TPR of 1 and an FPR of 1.

```
. h2omlestat threshmetric, threshold(0.1)
Metrics for specific threshold using H2O
Training frame: auto

Threshold
                    Input          .1
                 Computed        .125

Metric
                       F1          .7
                       F2       .8333
                     F0.5       .6034
                 Accuracy       .7568
                Precision       .5526
                   Recall       .9545
              Specificity       .6731
      Min. class accuracy       .6731
     Mean class accuracy       .8138
           True negatives          35
          False negatives           1
           True positives          21
          False positives          17
       True-negative rate       .6731
      False-negative rate       .0455
       True-positive rate       .9545
      False-positive rate       .3269
                      MCC       .5739
```

A threshold of 0.1 produces a TPR of 0.9545 and an FPR of 0.3269.

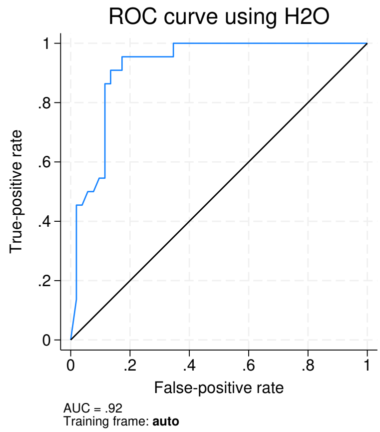
```
. h2omlestat threshmetric, threshold(1)
Metrics for specific threshold using H2O
Training frame: auto

Threshold
                    Input           1
                 Computed           1

Metric
                       F1       .2308
                       F2        .163
                     F0.5       .3947
                 Accuracy       .7297
                Precision         .75
                   Recall       .1364
              Specificity       .9808
      Min. class accuracy       .1364
     Mean class accuracy       .5586
           True negatives          51
          False negatives          19
           True positives           3
          False positives           1
       True-negative rate       .9808
      False-negative rate       .8636
       True-positive rate       .1364
      False-positive rate       .0192
                      MCC       .2368
```

A threshold of 1 produces a TPR of 0.1364 and an FPR of 0.0192.

If we repeat the same exercise with more threshold values and graph the corresponding TPRs and FPRs, the resulting curve is the ROC curve in the graph below.



The black reference line is the ROC curve for a method that randomly classifies with probability equal to 0.5. Therefore, a model that has a ROC curve that lies below the reference line performs worse than a random guess. Similarly, the further a model's ROC curve lies above the reference line, the better the model performs over a random guess.

We can also use ROC curves to compare models. The ROC curve located closest to the upper-left corner has the best performance. If ROC curves of two models overlap, then the higher AUC may indicate a better performance. In `h2omlgraph roc`, we can compare models by specifying the `models()` option with the names of two or more stored results.

◁

▷ Example 2: ROC for one model

In this example, we plot and interpret the ROC curve after performing random forest binary classification.

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 use the `_h2oframe split` command to randomly split the `auto` frame into a training frame (80% of observations) and a testing frame (20% of observations), which we name `train` 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/r19/auto
(1978 automobile dataset)

. h2o init
  (output omitted)

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

. _h2oframe split auto, into(train test) split(0.8 0.2) rseed(19)

. _h2oframe change train
```

Next we perform random forest binary classification with 3-fold cross-validation and store the estimation results by using the `h2omlest store` command.

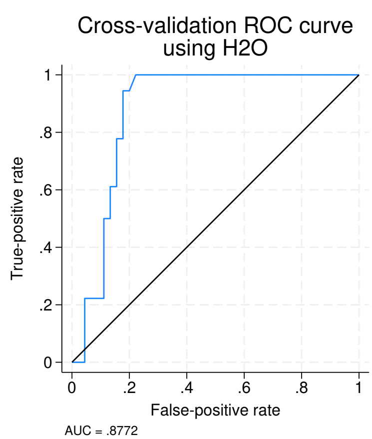
```
. global predictors price mpg trunk weight length
. h2oml rfbinclass foreign $predictors, h2orseed(19) cv(3, modulo)
Progress (%): 0 78.5 100
Random forest binary classification using H2O
Response: foreign
Frame:                                    Number of observations:
  Training: train                                    Training =     63
                                             Cross-validation =     63
Cross-validation: Modulo                  Number of folds      =      3
Model parameters
Number of trees        =  50
              actual =  50
Tree depth:                               Pred. sampling value =     -1
           Input max =  20                Sampling rate        =   .632
                 min =   4                No. of bins cat.     =  1,024
                 avg = 5.3                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	Cross-validation
Log loss	.8986088	.4191571
Mean class error	.1166667	.1166667
AUC	.8851852	.8771605
AUCPR	.590704	.5771737
Gini coefficient	.7703704	.754321
MSE	.1331692	.144763
RMSE	.3649235	.3804774

```
. h2omlest store RF
```

Finally, we plot the ROC curve by using the `h2omlgraph roc` command.

```
. h2omlgraph roc
```



Because the cv() option was specified and cross-validation was performed during the estimation, the default reported results correspond to the metrics calculated using cross-validation. The closer the curve is to the upper-left corner, the better the performance. This model performs substantially better than the reference line corresponding to random guessing.

◁

▷ Example 3: Comparing models using ROC

In example 2, we plotted the ROC curve for the random forest binary classification. In practice, the ROC curve is often used to compare the performance of different models on a testing frame. In this example, we compare the ROC curve for the random forest method with the one for the gradient boosting machine (GBM) method.

We use the h2omlpostestframe command to set the testing frame for the random forest model estimated in example 2.

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

Then we perform gradient boosting binary classification, set the testing frame for this model, and store the estimation results.

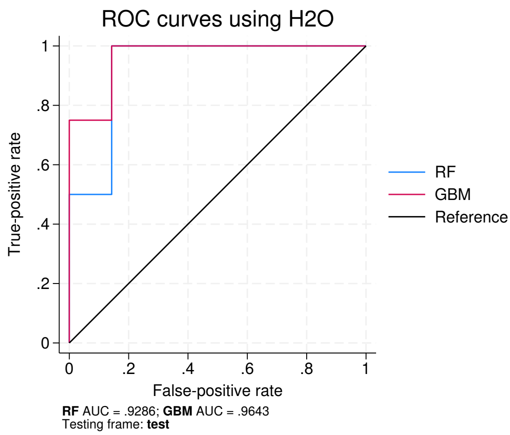
```
. h2oml gbbinclass foreign $predictors, h2orseed(19) cv(3, modulo)
Progress (%): 0 100
Gradient boosting binary classification using H2O
Response: foreign
Loss:     Bernoulli
Frame:                                     Number of observations:
  Training: train                                     Training =     63
                                              Cross-validation =     63
Cross-validation: Modulo                   Number of folds      =      3
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 = 3.5                 No. of bins root     =  1,024
                 max =   5                 No. of bins cont.    =     20
Min. obs. leaf split =  10                 Min. split thresh.   = .00001
Metric summary
```

Metric	Training	Cross-validation
Log loss	.0931244	.2803522
Mean class error	.0111111	.0666667
AUC	.9975309	.9259259
AUCPR	.9938208	.7733418
Gini coefficient	.9950617	.8518519
MSE	.0211802	.096305
RMSE	.1455344	.3103305

```
. h2omlpostestframe test
(testing frame test is now active for h2oml postestimation)
. h2omlest store GBM
```

To compare the ROC curves of the GBM and random forest models, with default hyperparameters, we use `h2omlgraph roc` with the `models()` option.

```
. h2omlgraph roc, models(RF GBM)
```



Based on the graph above, GBM performs better than random forest.

◁

Also see

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

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