h2omltree — Save decision tree DOT file and display rule set[+]

[+]This command includes features that are part of StataNow.

Description

`h2omltree` saves the decision tree plot in a DOT file and returns the decision rules for a specified tree after the `h2oml` *gbm* and `h2oml` *rf* commands. For details on how to work with DOT files and convert them to images, see [H2OML] **DOT extension**.

Quick start

Save the plot of the second tree as a DOT file after regression

```
h2omltree, id(2) dotsaving(tree.dot)
```

As above, but report the returned results as a rule set, and replace the existing `tree.dot` file

```
h2omltree, id(2) dotsaving(tree.dot, replace) rule
```

Save the plot of the first tree as a DOT file after multiclass classification, and use the second class as the target (reference) class

```
h2omltree, target(2) dotsaving(classtree.dot, replace)
```

As above, but set the direction to horizontal with the tree built left to right

```
h2omltree, target(2) dotsaving(classtree.dot, replace direction(lr))
```

Menu

Statistics > H2O machine learning

Syntax

`h2omltree` [, *options*]

options	Description
* `target(`*class*`)`	specify the target class of the response variable after multiclass classification
`id(`#`)`	specify the number of the tree; default is `id(1)`
`rule`	report the result as a rule set
`dotsaving(`*filename*[, *saveopts*]`)`	specify that the graph be saved as *filename*

*`target()` is required for multiclass classification.

saveopts	Description
`replace`	overwrites the existing file if it already exists
`direction(`*diropts*`)`	sets the direction of tree layout; may be `tb` (the default), `bt`, `lr`, or `rl`
`titile(`*string*`)`	specifies the tree title in the DOT file

Options

`target(`*class*`)` specifies the target class of the response variable for which the decision tree DOT file is to be created. `target()` is required after multiclass classification with `h2oml gbmulticlass` and `h2oml rfmulticlass`.

`id(#)` specifies the number of the tree. The default is the first tree.

`rule` specifies that the tree results be reported as a rule set.

`dotsaving(`*filename*`[ , `*saveopts*`])` specifies that the tree be saved as *filename*. *saveopts* are the following:

`replace` specifies that, if the file already exists, it is okay to replace it.

`direction(`*diropts*`)` sets the direction of the tree layout. *diropts* may be one of the following:

`tb` specifies that the tree is built top to bottom; the default.

`bt` specifies that the tree is built bottom to top.

`lr` specifies that the tree is built left to right.

`rl` specifies that the tree is built right to left.

`title(`*string*`)` specifies the tree title in the DOT file.

stata.com

Remarks and examples

We assume you have read the introduction to decision trees in [H2OML] **Intro**.

Remarks are presented under the following headings:

Example 1: Plotting a classification tree after random forest
Example 2: Plotting a classification tree after gradient boosting machine (GBM)
Example 3: Plotting a regression tree
Example 4: Plotting a tree for multiclass classification

An additional example can be found in *Explaining classification prediction* of [H2OML] **h2oml**.

All decision tree plots in the examples below are produced using Graphviz (https://graphviz.org). See [H2OML] **DOT extension** for more information.

Example 1: Plotting a classification tree after random forest

We plot and interpret binary classification trees produced by random forest.

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.

```
. use https://www.stata-press.com/data/r18/auto
(1978 automobile dataset)
. h2o init
  (output omitted)
. _h2oframe put, into(auto)
Progress (%): 0 100
. _h2oframe change auto
```

For simplicity, we save the predictor names in the global macro `predictors` in Stata. We then perform random forest binary classification with 100 trees and a maximum depth of 5.

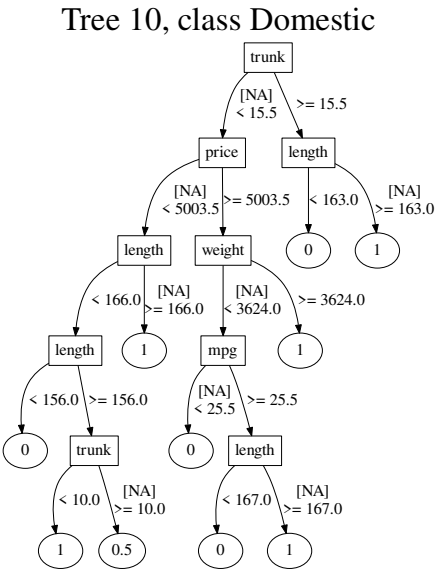
```
. global predictors price mpg trunk weight length
. h2oml rfbinclass foreign $predictors, h2orseed(19) ntrees(100) maxdepth(5)
Progress (%): 0 100
Random forest binary classification using H2O
Response: foreign
Frame:                                  Number of observations:
  Training: auto                                  Training =     74
Model parameters
Number of trees      = 100
              actual = 100
Tree depth:                             Pred. sampling value =     -1
           Input max =   5              Sampling rate        =   .632
                 min =   3              No. of bins cat.     =  1,024
                 avg = 4.8              No. of bins root     =  1,024
                 max =   5              No. of bins cont.    =     20
Min. obs. leaf split =   1              Min. split thresh.   = .00001
Metric summary
```

Metric	Training
Log loss	.3238765
Mean class error	.1223776
AUC	.9160839
AUCPR	.7850033
Gini coefficient	.8321678
MSE	.1089033
RMSE	.330005

Finally, we use the `h2omltree` command to save the 10th tree in the DOT file named `classtreerf.dot`.

```
. h2omltree, id(10) dotsaving(classtreerf, replace)
```

For binary classification, only the base class (the "negative" class) can be chosen as a target or reference class in H2O. In this example, this is the `Domestic` class. The tree plot shown below can be generated and saved as a PDF or another format using the information in `classtreerf.dot` and the Graphviz tool. For more details, refer to [H2OML] **DOT extension**.



The internal nodes in the tree correspond to the predictor names for which the split has occurred and the terminal nodes correspond to $P(\texttt{Domestic} = 1)$. Each internal predictor separates data based on the split. The `NA`'s on the branches indicate the split of the missing values, if any. Based on this tree, for the observations with `length` ≥ 163, the predicted probability of the car being domestic is 1.

Example 2: Plotting a classification tree after gradient boosting machine (GBM)

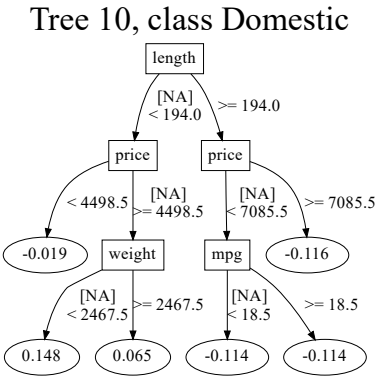
In this example, we plot a classification tree after gradient boosting binary classification. We start by running the `h2oml gbbinclass` command with options `ntrees(100)` and `maxdepth(5)`.

```
. h2oml gbbinclass foreign $predictors, h2orseed(19)  ntrees(100) maxdepth(5)
```
(*output omitted*)

Then we use the `h2omltree` command to save the 10th tree in the DOT file named `classtreegbm.dot`

```
. h2omltree, id(10) dotsaving(classtreegbm, replace)
```

The tree below is generated from the `classtreegbm.dot` file using Graphviz.



Compared with the classification tree in *Example 1: Plotting a classification tree after random forest*, the terminal nodes of the classification tree after GBM contain negative values. This may be surprising because the expected values should be between $[0, 1]$. However, as we explain below, this is the expected behavior.

As discussed in the *Introduction* of [H2OML] ***h2oml gbm***, GBM relies on link functions to determine the loss function. For instance, in binary classification, GBM uses the logit link function. Consequently, for certain postestimation commands, such as `h2omltree` and `h2omlgraph shapvalues`, probabilities are obtained by applying the inverse link function, in this case, the inverse logit function.

For example, the predicted raw value -0.114 in the terminal node corresponds to probability 0.47153083.

```
. display invlogit(-0.114)
.47153083
```

Here the terminal nodes can be explained based on increasing or decreasing probability $P(\texttt{Domestic} = 1)$. Thus, the highest probability corresponds to 0.148 (probability of 0.54) and occurs for the observations with length less than 194, price greater than 4498.5, and weight less than 2467.5.

Example 3: Plotting a regression tree

In this example, we create and save a DOT file and display a regression tree for random forest regression.

We start by redefining the global macro `predictors`. Then we perform random forest regression with 100 trees and a maximum depth of 5 for each tree.

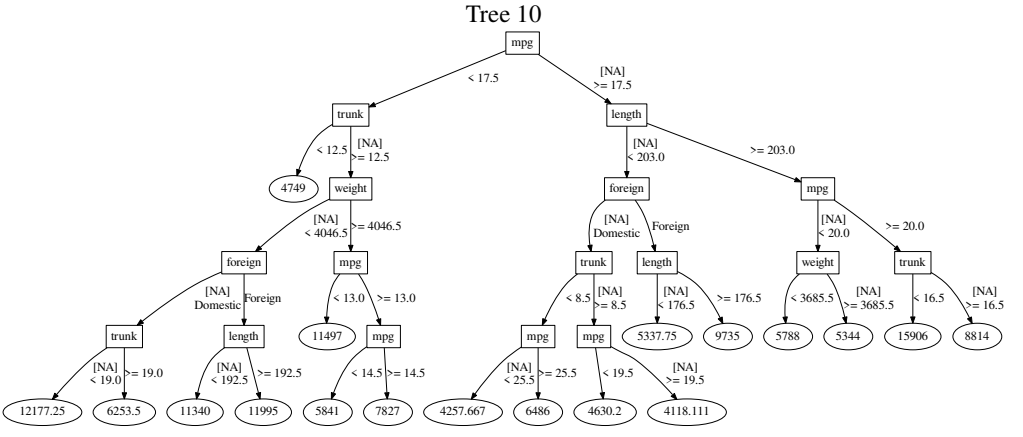
```
. global predictors foreign mpg trunk weight length
. h2oml rfregress price $predictors, h2orseed(19) ntrees(100) maxdepth(5)
Progress (%): 0 100
Random forest regression using H2O
Response: price
Frame:                                  Number of observations:
  Training: auto                                  Training =     74
Model parameters
Number of trees      = 100
             actual = 100
Tree depth:                             Pred. sampling value =     -1
           Input max =   5              Sampling rate        =   .632
                 min =   2              No. of bins cat.     =  1,024
                 avg = 5.0              No. of bins root     =  1,024
                 max =   5              No. of bins cont.    =     20
Min. obs. leaf split =   1              Min. split thresh.   = .00001
Metric summary
```

Metric	Training
Deviance	3129378
MSE	3129378
RMSE	1769.005
RMSLE	.2315556
MAE	1229.955
R-squared	.6353542

We save the regression tree as a DOT file by using the `h2omltree` command.

```
. h2omltree, id(10) dotsaving(regtreerf, replace)
```

The following tree is created from the `regtreerf.dot` file using Graphviz.



From the tree above, the predicted price for the cars with mileage per gallon less than 17.5 and trunk space less than 12.5 cu.ft. is equal to $4,749.

Example 4: Plotting a tree for multiclass classification

In this example, we create a DOT file for a tree for multiclass classification by using the `iris` dataset and random forest. This dataset was used in Fisher (1936) and originally collected by Anderson (1935).

We start by initializing a cluster, opening the dataset in Stata, and importing the dataset as an H2O frame.

```
. use https://www.stata-press.com/data/r18/iris
(Iris data)
. h2o init
  (output omitted)
. _h2oframe put, into(iris)
Progress (%): 0 100
. _h2oframe change iris
```

Next we define the global macro `predictors` to store the name of predictors and perform random forest multiclass classification.

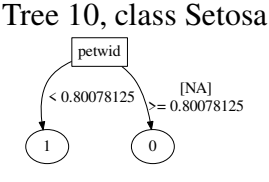
```
. global predictors seplen sepwid petlen petwid
. h2oml rfmulticlass iris $predictors, h2orseed(19) ntrees(100) maxdepth(5)
Progress (%): 0 100
Random forest multiclass classification using H2O
Response: iris                          Number of classes    =      3
Frame:                                  Number of observations:
  Training: iris                                  Training =    150
Model parameters
Number of trees      = 100
             actual = 100
Tree depth:                             Pred. sampling value =     -1
           Input max =   5              Sampling rate        =   .632
                 min =   1              No. of bins cat.     =  1,024
                 avg = 3.4              No. of bins root     =  1,024
                 max =   5              No. of bins cont.    =     20
Min. obs. leaf split =   1              Min. split thresh.   = .00001
Metric summary
```

Metric	Training
Log loss	.1290855
Mean class error	.06
MSE	.0370932
RMSE	.1925959

To save a tree after a multiclass classification, you must specify the option `target()` in the `h2omltree` command. Here we create a DOT file to plot the 10th tree for the class `Setosa`.

```
. h2omltree, id(10) dotsaving(mclasstreerf, replace) target(Setosa)
```

The following tree is created from the `mclasstreerf.dot` file using Graphviz.



References

Anderson, E. 1935. The irises of the Gaspé Peninsula. *Bulletin of the American Iris Society* 59: 2–5.

Fisher, R. A. 1936. The use of multiple measurements in taxonomic problems. *Annals of Eugenics* 7: 179–188. https://doi.org/10.1111/j.1469-1809.1936.tb02137.x.

Also see

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

[H2OML] **DOT extension** — Handling DOT files+