h2omlgraph shapvalues — Produce SHAP values plot for individual observations[+]

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

Description

`h2omlgraph shapvalues` plots the Shapley additive explanation (SHAP) values for an individual observation after regression or binary classification performed by `h2oml gbregress`, `h2oml rfregress`, `h2oml gbbinclass`, or `h2oml rfbinclass`. SHAP values indicate the contributions of predictors to the prediction for a given observation. SHAP values are considered a unified measure for variable importance and machine learning model explanation.

Quick start

Plot individual SHAP values for the third observation

```
h2omlgraph shapvalues, obs(3)
```

As above, but use H2O frame `myframe` and predictors `x1`, `x2`, and `x3`

```
h2omlgraph shapvalues x1-x3, obs(3) frame(myframe)
```

As above, but instead of `x1`, `x2`, and `x3`, plot the top 4 SHAP-important predictors

```
h2omlgraph shapvalues, obs(3) frame(myframe) top(4)
```

As above, but save the result in the `shapval3.dta` file

```
h2omlgraph shapvalues, obs(3) frame(myframe) top(4) ///
  savedata(shapval3, replace)
```

Menu

Statistics > H2O machine learning

Syntax

```
h2omlgraph shapvalues [predictors], obs(#) [options]
```

options	Description
Main	
* `obs(#)`	specify the observation number for which SHAP will be computed
`impplot`	plot SHAP values as zero-based importance—as deviations from zero rather than deviations from average prediction
`top(#)`	display the top # highest SHAP-important predictors; default is `top(20)`
`savedata(`*filename*`[ , replace])`	save plot data to *filename*
Plot options	
`norefline`	suppress reference line at zero for zero-based importance
`rlopts(`*line_options*`)`	affect rendition of reference line for zero-based importance
`nopredline`	suppress prediction line
`predlineopts(`*line_options*`)`	affect rendition of prediction line
`nopredlabel`	suppress label of prediction line
`predlabelopts(`*textbox_options*`)`	affect labeling of prediction line
`nobiasline`	suppress bias line
`biaslineopts(`*line_options*`)`	affect rendition of bias line that identifies the expected model prediction
`nobiaslabel`	suppress label of bias line
`biaslabelopts(`*textbox_options*`)`	affect labeling of bias line
`noboundarylines`	suppress boundary lines for SHAP contribution bars
`boundarylineopts(`*line_options*`)`	affect rendition of boundary lines for SHAP contribution bars
`novaluelabel`	suppress labels of SHAP values
`valuelabelopts(`*label_opts*`)`	affect labeling of SHAP values
`poscolor(`*colorstyle*`)`	affect color for positive SHAP values
`negcolor(`*colorstyle*`)`	affect color for negative SHAP values
`bar#opts(`*bar_opts*`)`	affect rendition of the bar for the #th SHAP-important predictor
`baropts(`*bar_opts*`)`	affect rendition of all bars for the SHAP plot
`barwidth(#)`	specify the bar width; default is `barwidth(0.9)`
Y axis, X axis, Titles, Legend, Overall	
twoway_options	any option other than `by()` documented in [G-3] ***twoway_options***
`train`	specify that SHAP values be reported using training results
`valid`	specify that SHAP values be reported using validation results
`test`	specify that SHAP values be computed using testing frame
`test(`*framename*`)`	specify that SHAP values be computed using data in testing frame *framename*
`frame(`*framename*`)`	specify that SHAP values be computed using data in H2O frame *framename*
`framelabel(`*string*`)`	label frame as *string* in the output

*`obs()` is required.

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

Options

Main

`obs(#)` specifies the observation number for which SHAP will be computed. # must be a positive integer. `obs()` is required.

`impplot` plots SHAP values as deviations from zero rather than deviations from the average model prediction. `impplot` is not allowed with any of options `predlineopts()`, `predlabelopts()`, `biaslineopts()`, `biaslabelopts()`, `valuelabelopts()`, or `boundarylineopts()`.

`top(#)` specifies the number of highest SHAP-important predictors to be included in the plot. Up to 20 top important predictors are included by default. `top()` is not allowed if *predictors* are specified.

`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

`norefline` suppresses the reference line at zero when zero-based importance is plotted. `norefline` may be specified with only option `impplot`. The reference line is included by default.

`rlopts(`*line_options*`)` affects the rendition of the reference line at zero for zero-based importance. `rlopts()` must be specified with the option `impplot`. See [G-3] ***line_options***.

`nopredline` suppresses prediction line identifying the predicted value for regression or the predicted probability for classification. When gradient boosting machine is used, the predicted values correspond to the raw predictions of the model before applying the inverse link function.

`predlineopts(`*line_options*`)` affects rendition of prediction line. See [G-3] ***line_options***. `predlineopts()` is not allowed with `impplot`.

`nopredlabel` suppresses the label for prediction line.

`predlabelopts(`*textbox_options*`)` affects labeling of prediction line. See [G-3] ***textbox_options***. `predlabelopts()` is not allowed with `impplot`.

`nobiasline` suppresses bias line identifying the expected model response—the contribution of the model without any predictors. When gradient boosting machine is used, the bias value corresponds to the raw prediction of the model before applying the inverse link function.

`biaslineopts(`*line_options*`)` affects rendition of bias line. See [G-3] ***line_options***. `biaslineopts()` is not allowed with `impplot`.

`nobiaslabel` suppresses the label for the bias line.

`biaslabelopts(`*textbox_options*`)` affects labeling of bias line. See [G-3] ***textbox_options***. `biaslabelopts()` is not allowed with `impplot`.

`noboundarylines` suppresses the boundary lines for the SHAP contribution bars.

`boundarylineopts(`*line_options*`)` affects the rendition of the lines on the boundaries of the bars for the SHAP contributions. `boundarylineopts()` is not allowed with `impplot`. See [G-3] ***line_options***.

`novaluelabel` suppresses labeling of the SHAP contributions for each predictor.

`valuelabelopts(`*label_opts*`)` affects labeling of the SHAP values for each predictor. See [G-3] ***marker_label_options***. The labels are numbers that show the SHAP values. `valuelabel()` is not allowed with `impplot`.

`poscolor(`*colorstyle*`)` affects the bar color of the positive SHAP contributions. See [G-4] ***colorstyle***.

`negcolor(`*colorstyle*`)` affects the bar color of the negative SHAP contributions. See [G-4] ***colorstyle***.

`bar#opts(`*bar_opts*`)` affects rendition of the bar for the SHAP-important predictor #. In an `h2omlgraph shapvalues` plot, the order of the predictors is based on SHAP importance. The predictor with largest magnitude of SHAP values will be the first and so on. For example, to change the rendition of the bar for the third-ranked predictor, we need to specify `bar3opts()`. See [G-2] **graph twoway bar**.

`baropts(`*bar_opts*`)` affects rendition of all bars for the SHAP plot. See [G-2] **graph twoway bar**.

`barwidth(#)` specifies the width of the bar. The default is `barwidth(0.9)`.

Y axis, X axis, Titles, Legend, Overall

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 shapvalues` but are not shown in the dialog box:

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

`train` specifies that SHAP values be reported using training results. This is the default when validation is not performed during estimation and when a postestimation frame has not been set with `h2omlpostestframe`.

`valid` specifies that SHAP values 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*.

`test` specifies that SHAP values 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 by using `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 SHAP values 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, it is more computationally efficient and convenient to specify the testing frame by using `h2omlpostestframe` instead of specifying `test(`*framename*`)` with individual postestimation commands.

`frame(`*framename*`)` specifies that SHAP values be computed using the data in H2O frame *framename*.

`framelabel(`*string*`)` specifies the label to be used for the frame in the output.
stata.com

Remarks and examples

We assume you have read the introduction to explainable machine learning in *Interpretation and explanation* in [H2OML] **Intro**.

SHAP values are used to explain the predictions of a model by measuring the contribution of each predictor to those predictions. Specifically, for a given prediction, the SHAP value measures the contribution of a predictor to the deviation of that prediction from a base prediction, typically from the average prediction our model makes (Štrumbelj and Kononenko 2010, 2013; Lundberg and Lee 2017).

In a traditional linear regression with no interaction terms, the computation of SHAP has a simple closed-form solution. For example, the contribution of predictor X_1 to the prediction is simply the estimated coefficient on X_1 multiplied by the observed value x_{1i}. However, for a typical machine learning model, no such coefficients are available, so computing the contributions requires an alternative approach.

In this entry, we focus on local SHAP explanation, which allows us to explain the effect of predictors for one observation at a time. The `h2omlgraph shapvalues` command plots this type of local SHAP values. For global SHAP explanations, the `h2omlgraph shapsummary` command uses the Kernel SHAP algorithm (Lundberg and Lee 2017) and produces a beeswarm plot that summarizes how each predictor affects predictions across many observations.

For intuition on SHAP values, suppose we have trained a machine learning model, such as random forest, to predict the price of a car using three predictors: mileage (M), number of accidents (A), and the presence of add-on features (F). A new car then arrives with mileage equal to 6,000 miles, a history of 1 accident, and with add-on features. In the `h2omlgraph shapvalues` command, we specify the observation number for this new car with the `obs()` option. Finally, suppose the predicted price for the car is \$32,000 and the average predicted price for all cars is \$29,000. Our goal then is to measure the contribution of each predictor (M, A, and F) to the \$32,000 − \$29,000 = \$3,000 by which the predicted price of the new car deviates from the average predicted price.

The general idea of SHAP values is to imagine that the three predictors collaborate with each other to achieve the predicted value. For example, suppose for the newly arrived car we start by adding the predictor M into our model and observe that it contributes \$7,000 to the prediction, then add the number of accidents A predictor and see that it contributes −\$5,000. Finally, the presence of add-on features F contributes \$1,000 to the so-called coalition of predictors {M, A}. The contribution of all predictors then adds up to the \$3,000, the deviation we computed above. Unfortunately, the contribution of each predictor depends on the order at which it enters the model; that is, it depends on the coalition of the previously entered predictors. Notice that the coalition S of predictors that entered the model before M could be one of four:

$$\mathrm{S} \in \{\{\emptyset\}, \{\mathrm{A}\}, \{\mathrm{F}\}, \{\mathrm{A}, \mathrm{F}\}\}$$

And there are eight possible coalitions of predictors:

$$\mathrm{C} = \{\mathrm{M}, \mathrm{A}, \mathrm{F}\} : \{\emptyset\}, \{\mathrm{M}\}, \{\mathrm{A}\}, \{\mathrm{F}\}, \{\mathrm{M}, \mathrm{A}\}, \{\mathrm{M}, \mathrm{F}\}, \{\mathrm{A}, \mathrm{F}\}, \{\mathrm{M}, \mathrm{A}, \mathrm{F}\}$$

Therefore, the SHAP contribution of M is a weighted average of the differences of contributions of a coalition with M, denoted $v_x(\mathrm{S} \cup \mathrm{M})$, and a coalition excluding M, denoted $v_x(\mathrm{S})$, for each possible scenario of S. Here $v_x(\mathrm{S})$ is defined as a conditional expectation of the prediction given the observed values of predictors in the coalition S,

$$v_x(\mathrm{S}) = E(\hat{f}(\mathbf{x})|\mathbf{x}_\mathrm{S})$$

where $\hat{f}(\mathbf{x})$ is the prediction for a specific observation $\mathbf{x}$. For more details, see Lundberg and Lee (2017) and Aas, Jullum, and Løland (2021).

For machine learning methods, there is no simple form for the weighted average and with many predictors, direct computation becomes intractable. Therefore, H2O uses the TREESHAP algorithm, introduced in Lundberg, Erion, and Lee (2018), which is an efficient procedure for the exact computation of the SHAP values.

SHAP values have desirable properties (Molnar 2022, chap. 9). For instance, the efficiency property is

$$\hat{f}(\mathbf{x}) = \phi_0 + \sum_{j=1}^{p} \phi_j$$

where $\phi_0 = E\{\hat{f}(\mathbf{x})\}$ is the average predicted contribution and ϕ_j, $j = 1, \ldots, p$ is the SHAP value of each predictor. The prediction for each observation is the sum of the average prediction plus the SHAP values for all predictors.

We can also define SHAP predictor importance (Molnar 2022, chap. 9.6), which is based on the idea that important predictors are associated with large absolute SHAP values. Thus, the global importance for predictors $j = 1, \ldots, p$ can be computed by averaging their absolute SHAP values over the observations

$$I_j = \frac{1}{N} \sum_{i=1}^{n} |\phi_j^{(i)}|$$

In `h2omlgraph shapvalues`, you can specify that only a given number of highest SHAP-important predictors to be included in the graph with the `top()` option.

▷ Example 1: Interpreting SHAP values

In this example, we interpret SHAP values after performing random forest regression.

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. 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 change auto
```

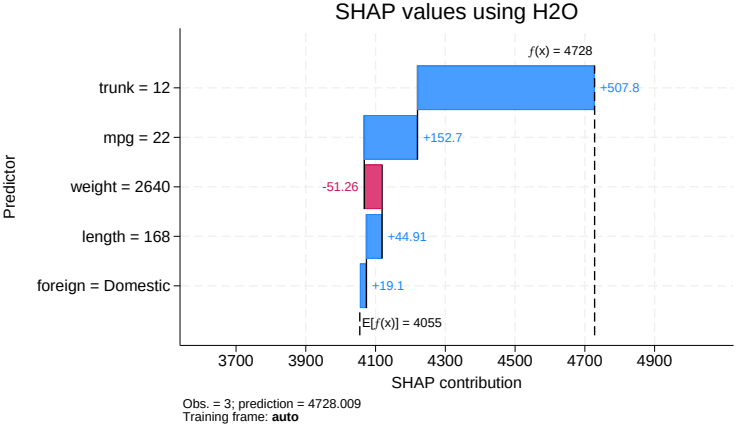
For simplicity, we save the predictor names in the global macro `predictors` in Stata. We then perform random forest regression with 100 trees and limit the maximum depth of the trees to 5.

```
. 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

Finally, we use the `h2omlgraph shapvalues` command to plot SHAP values for the third observation.

```
. h2omlgraph shapvalues, obs(3)
```

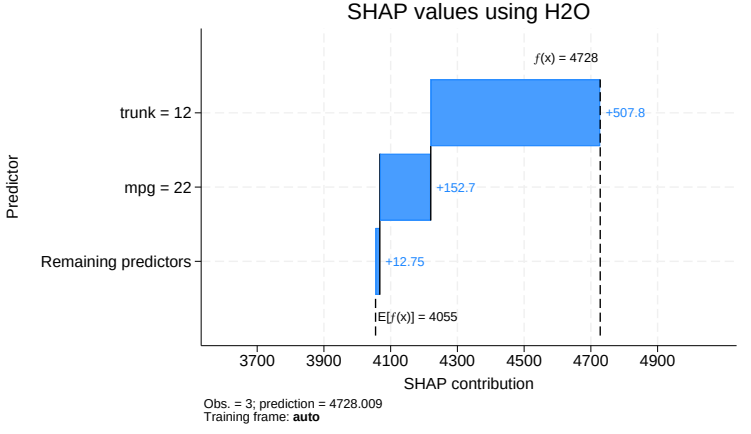


In this case, the predicted car price is 4728. We wish to explain the contribution of each predictor to this predicted price. In the plot, the contributions are plotted bottom to top, starting from the baseline value, which is the average prediction of 4055. We can see from the top blue bar that `trunk = 12` has a

positive SHAP value, which means it increases the predicted price. On the other hand, `weight = 2640` has a negative contribution to the predicted price as indicated by the red bar in the center of the graph. The sum of the bars in the plot is equal to the difference of the predicted price and the bias term 4728 − 4055.

If we wish to display contributions of a subset of predictors, for example, `trunk` and `mpg`, the plot can be customized to show contributions of this subset by specifying the names of the predictors in the `h2omlgraph shapvalues` command.

```
. h2omlgraph shapvalues trunk mpg, obs(3)
```



In this case, the bottom bar in the plot shows the total contribution of the remaining predictors. The order of the predictors is determined based on the magnitude of their SHAP values.

Example 2: Explaining voting behavior

In this example, we consider the social pressure dataset described in example 1 of [H2OML] ***h2oml rf***. The goal is to explain how the predictors affect the probability of voting in the August 2006 primary election. As with most explainable machine learning methods, caution is advised when interpreting the results.

We start by opening the simulated `socialpressure.dta` dataset in Stata and then putting it into an H2O frame.

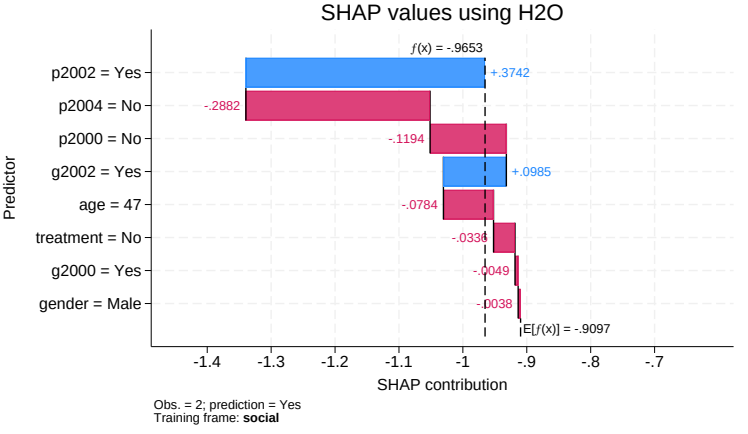
```
. use https://www.stata-press.com/data/r18/socialpressure
(Social pressure data)
. h2o init
. _h2oframe _put, into(social)
Progress (%): 0 100
. _h2oframe _change social
```

For convenience, we create a global macro, `predictors`, in Stata that contains the predictor names and perform gradient boosting binary classification with a learning rate of 0.05, a maximum tree depth of 6, and 70 trees.

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

. h2oml gbbinclass voted $predictors, h2orseed(19) lrate(0.05)
> maxdepth(6) ntrees(70)

Progress (%): 0 1.4 4.2 27.1 38.5 64.2 74.2 100

Gradient boosting binary classification using H2O

Response: voted
Loss:     Bernoulli
Frame:                                  Number of observations:
  Training: social                                Training = 229,461

Model parameters

Number of trees      =  70              Learning rate       =     .05
             actual =  70               Learning rate decay =       1
Tree depth:                             Pred. sampling rate =       1
           Input max =   6              Sampling rate       =       1
                 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 =  10              Min. split thresh.  =  .00001

Metric summary

------------------------------
          Metric |   Training
-----------------+------------
       Log loss  |   .5695804
 Mean class error|   .3907184
             AUC |   .6771573
           AUCPR |   .4761226
Gini coefficient |   .3543147
             MSE |   .1934469
            RMSE |    .439826
------------------------------
```

We display SHAP values for the second observation of the dataset by using the `h2omlgraph shapvalues` command with the option `obs(2)`. The option `xlabel()` improves the display of the figure by setting the range of the x axis to a convenient interval.

```
. h2omlgraph shapvalues, obs(2) xlabel(-1.4(0.1)-0.7)
```



The second observation corresponds to a male who voted in the primary election, so our goal is to explain why the prediction of his vote is "Yes" based on predictors. We can see that the subject being male has a very small effect on the probability of voting. On the other hand, as expected, voting in the primary election in 2002 (p2002) has a substantial positive effect on the probability of voting.

Note that the reported SHAP values after `h2oml gbbinclass` are reported as raw predictions. To interpret these values as probabilities, we need to apply the inverse logit transformation to the values shown in the graph. Similarly, for SHAP values reported after `h2oml gbregress` with a loss other than Gaussian, an appropriate transformation may be needed for interpretation. Nonetheless, the graph still allows us to infer the direction and magnitude of the predictions directly.

◁

References

Aas, K., M. Jullum, and A. Løland. 2021. Explaining individual predictions when features are dependent: More accurate approximations to Shapley values. *Artificial Intelligence* 298: art. 103502. https://doi.org/10.1016/j.artint.2021.103502.

Lundberg, S. M., G. G. Erion, and S. Lee. 2018. Consistent individualized feature attribution for tree ensembles. arXiv:1802.03888 [cs.LG], https://doi.org/10.48550/arXiv.1802.03888.

Lundberg, S. M., and S. Lee. 2017. "A unified approach to interpreting model predictions". In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, vol. 30: 4768–4777. Red Hook, NY: Curran Associates.

Molnar, C. 2022. *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable*. 2nd ed. https://christophm.github.io/interpretable-ml-book.

Štrumbelj, E., and I. Kononenko. 2010. An efficient explanation of individual classifications using game theory. *Journal of Machine Learning Research* 11: 1–18.

———. 2013. Explaining prediction models and individual predictions with feature contributions. *Knowledge and Information Systems* 41: 647–665. https://doi.org/10.1007/s10115-013-0679-x.

Also see

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

[H2OML] **h2omlgraph shapsummary** — Produce SHAP beeswarm plot+