Package ‘shadowVIMP’

June 19, 2025

Title Covariate Selection Based on VIMP Permutation-Like Testing

Version 1.0.2

Description A statistical method for reducing the number of covariates in an analysis by evaluating Variable Importance Measures (VIMPs) derived from the Random Forest algorithm. It performs statistical tests on the VIMPs and outputs whether the covariate is significant along with the p-values.

License Apache License (>= 2)

Imports dplyr, ggforce, ggplot2, ggpubr, magrittr, parallel, patchwork, ranger, rlang, stats, stringr, tidyr

Suggests knitr, rmarkdown, spelling, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

Language en-GB

RoxygenNote 7.3.2

URL https://github.com/OktawiaStaburo/shadowVIMP

BugReports https://github.com/OktawiaStaburo/shadowVIMP/issues

NeedsCompilation no

Author Tim Mueller [aut],
Oktawia Miluch [aut, cre],
Staburo GmbH [cph, fnd]

Maintainer Oktawia Miluch <oktawia.miluch@staburo.de>

Repository CRAN

Date/Publication 2025-06-19 15:00:02 UTC

Contents

`plot_vimps` *Box plot of VIMPs and corresponding p-values*

Description

Box plot displaying variable importance measures along with unadjusted, FDR-adjusted, and FWER-adjusted p-values obtained from the `shadow_vimp()` function. Colors indicate whether each covariate is informative and specify under which multiple testing adjustment (FWER, FDR, or none) it is deemed informative.

Usage

```
plot_vimps(
  shadow_vimp_out,
  pooled = TRUE,
  filter_vars = NULL,
  helper_legend = TRUE,
  p_val_labels = TRUE,
  text_size = 4,
  legend.position = c("right", "left", "top", "bottom", "none"),
  category_colors = c(`FWER conf.` = "#DD5129FF", `FDR conf.` = "#0F7BA2FF",
    `Unadjusted conf.` = "#43B284FF", `Not significant` = "#898E9FFF"),
  ...
)
```

Arguments

`shadow_vimp_out`
Object of the class "shadow_vimp", the output of the function `shadow_vimp()`.

`pooled` Boolean

- `TRUE` - passed `shadow_vimp_out` contains p-values obtained using the " pooled" approach. Default.
- `FALSE` - passed `shadow_vimp_out` contains p-values obtained using the "per variable" approach.

`filter_vars` Numeric, the number of variables to plot. The default is `NULL`, which means that all variables considered in the last step of the procedure (and included in the `shadow_vimp_out`) will be plotted.

`helper_legend` Boolean. Indicates whether the circle subplot displaying the relationship between the FWER, FDR, and unadjusted p-values should be shown alongside the legend. The default is `TRUE`.

`p_val_labels` Boolean, controls whether the p-value labels should be printed on the plot, default `TRUE`.

`text_size` Numeric, parameter that controls the size of the printed p-values on the plot, default is 4.

`legend.position`
: Character, one of "right", "left", "top", "bottom" or "none". Argument specifying the position of the legend.

`category_colors`
: Character of length 4, containing color assignment for each of four possible outcomes: variable not significant, confirmed by unadjusted, FDR and FWER adjusted p-values. The default colors are color blind friendly.

`...`
: Other options used to control the appearance of the output plot.

Value

ggplot object

Examples

```
data(mtcars)

# When working with real data, increase the value of the `niters` and
# `num.trees` parameters to obtain trustworthy results.

# Function to make sure proper number of cores is specified for multithreading
safe_num_threads <- function(n) {
  available <- parallel::detectCores()
  if (n > available) available else n
}

# Pooled p-values
set.seed(789)
out_pooled <- shadow_vimp(
  data = mtcars, outcome_var = "vs",
  niters = c(10, 20, 30), num.trees = 30,
  num.threads = safe_num_threads(1)
)

# The following 3 lines of code produce identical plots
plot_vimps(shadow_vimp_out = out_pooled, pooled = TRUE, text_size = 4)
plot_vimps(shadow_vimp_out = out_pooled, text_size = 4)
plot_vimps(shadow_vimp_out = out_pooled)

# Plot only top 3 covariates with the lowest p-values
plot_vimps(shadow_vimp_out = out_pooled, filter_vars = 3)

#' # Do not display p-values on the plot
plot_vimps(shadow_vimp_out = out_pooled, p_val_labels = FALSE)

# Change the size of displayed p-values
plot_vimps(shadow_vimp_out = out_pooled, text_size = 6)

# Change the position of the legend, available options: "right", "left",
# "top","bottom", "none"
plot_vimps(shadow_vimp_out = out_pooled, legend.position = "bottom")
plot_vimps(shadow_vimp_out = out_pooled, legend.position = "left")
```

```
# Remove the legend
plot_vimps(shadow_vimp_out = out_pooled, legend.position = "none")

# Remove the subplot that displays the relationship between FWER, FDR, and
# unadjusted p-values
plot_vimps(shadow_vimp_out = out_pooled, helper_legend = FALSE)

# Change colours of the boxes
plot_vimps(shadow_vimp_out = out_pooled, category_colors = c(
  "FWER conf." = "#EE2617FF",
  "FDR conf." = "#F2A241FF",
  "Unadjusted conf." = "#558934FF",
  "Not significant" = "#0E54B6FF"
))

# Per variable p-values plot
out_per_var <- shadow_vimp(
  data = mtcars, outcome_var = "vs",
  niters = c(10, 20, 30), num.trees = 30,
  method = "per_variable", num.threads = safe_num_threads(1)
)

# Set pooled to `FALSE`, otherwise the function will throw an error.
plot_vimps(shadow_vimp_out = out_per_var, pooled = FALSE)
```

print.shadow_vimp *Print shadow_vimp results*

Description

Custom print function to display the key elements of the `shadow_vimp()` results.

Usage

```
## S3 method for class 'shadow_vimp'
print(x, ...)
```

Arguments

x	Object of class ’shadow_vimp’
...	Further arguments passed to or from other methods.

Value

The object `x`, invisibly.

See Also

shadow_vimp

---

shadow_vimp — *Select influential covariates in random forests using multiple testing control*

Description

shadow_vimp() performs variable selection and determines whether each covariate is influential based on unadjusted, FDR-adjusted, and FWER-adjusted p-values.

Usage

```
shadow_vimp(
  alphas = c(0.3, 0.1, 0.05),
  niters = c(30, 120, 1500),
  data,
  outcome_var,
  num.trees = max(2 * (ncol(data) - 1), 10000),
  num.threads = NULL,
  importance = "permutation",
  save_vimp_history = c("all", "last", "none"),
  to_show = c("FWER", "FDR", "unadjusted"),
  method = c("pooled", "per_variable"),
  ...
)
```

Arguments

alphas — Numeric vector, significance level values for each step of the procedure, default c(0.3, 0.10, 0.05).

niters — Numeric vector, number of permutations to be performed in each step of the procedure, default c(30, 120, 1500).

data — Input data frame.

outcome_var — Character, name of the column containing the outcome variable.

num.trees — Numeric, number of trees. Passed to ranger::ranger(), default is max(2 * (ncol(data) - 1), 10000).

num.threads — Numeric. The number of threads used by ranger::ranger() for parallel tree building. If NULL (the default), half of the available CPU threads are used (this is the default behaviour in shadow_vimp(), which is different from the default in ranger::ranger()). See the ranger::ranger() documentation for more details.

importance — Character, the type of variable importance to be calculated for each variable. Argument passed to ranger::ranger(), default is permutation.

`save_vimp_history`
Character, specifies which variable importance measures to save. Possible values are:

- `"all"` (the default) - save variable importance measures from all steps of the procedure (both the pre-selection phase and the final selection step).
- `"last"` - save only the variable importance measures from the final step.
- `"none"` - do not save any variable importance measures.

`to_show`
Character, one of `"FWER"`, `"FDR"` or `"unadjusted"`.

- `"FWER"` (the default) - the output includes unadjusted, Benjamini-Hochberg (FDR) and Holm (FWER) adjusted p-values together with the decision whether the variable is significant or not (1 - significant, 0 means not significant) according to the chosen criterium.
- `"FDR"` - the output includes both unadjusted and FDR adjusted p-values along with the decision.
- `"unadjusted:` - the output contains only raw, unadjusted p-values together with the decision.

`method`
Character, one of `"pooled"` or `"per_variable"`.

- `"pooled"` (the default) - the results of the final step of the procedure show the p-values obtained using the "pooled" approach and the corresponding decisions.
- `"per_variable"` - the results of the final step of the procedure show the p-values obtained using the "per variable" approach and the corresponding decisions.

`...`
Additional parameters passed to `ranger::ranger()`.

Details

The `shadow_vimp()` function by default performs variable selection in multiple steps. Initially, it prunes the set of predictors using a relaxed (higher) alpha threshold in a pre-selection stage. Variables that pass this stage then undergo a final evaluation using the target (lower) alpha threshold and more iterations. This stepwise approach distinguishes informative from uninformative covariates based on their VIMPs and enhances computational efficiency. The user can also perform variable selection in a single step, without a pre-selection phase.

Value

Object of the class "shadow_vimp" with the following entries:

- `call` - the call formula used to generate the output.
- `alpha` - numeric, significance level used in the algorithm.
- `step_all_covariates_removed` - integer. If > 0, the step number at which all candidate covariates were deemed insignificant and removed. If 0, at least one covariate survived the pre-selection until the last step of the procedure.
- `final_dec_pooled` (the default) or `final_dec_per_variable` - a data frame that contains, depending on the specified value of the `to_show` parameter, p-values and corresponding decisions (in columns with names ending in `confirmed`) if the variable is deemed informative

at the final step of the procedure: 1 = covariate considered informative in the last step; 0 = not informative. If all covariates were dropped in the pre-selection, i.e. none reached the final step, then all p-values are NA and all decisions are set to 0.

- `vimp_history`- if `save_vimp_history` is set to `"all"` or `"last"` then it is a data frame with VIMPs of covariates and their shadows from the last step of the procedure. If `save_vimp_history` is set to `"none"`, then it is `NULL`.
- `time_elapsed` - list containing the runtime of each step and the total time taken to execute the code.
- `pre_selection` - list in which the results of the pre-selection are stored. The exact form of this element depends on the chosen value of the `save_vimp_history` parameter.

Examples

```
data(mtcars)

# When working with real data, use higher values for the niters and num.trees
# parameters --> here these parameters are set to small values to reduce the
# runtime.

# Function to make sure proper number of cores is specified
safe_num_threads <- function(n) {
  available <- parallel::detectCores()
  if (n > available) available else n
}

# Standard use
out1 <- shadow_vimp(
  data = mtcars, outcome_var = "vs",
  niters = c(10, 20, 30), num.trees = 30, num.threads = safe_num_threads(1)
)


# `num.threads` sets the number of threads for multithreading in
# `ranger::ranger`. By default, the `shadow_vimp` function uses half the
# available CPU threads.
out2 <- shadow_vimp(
  data = mtcars, outcome_var = "vs",
  niters = c(10, 20, 30), num.threads = safe_num_threads(2),
  num.trees = 30
)

# Save variable importance measures only from the final step of the
# procedure
out4 <- shadow_vimp(
  data = mtcars, outcome_var = "vs",
  niters = c(10, 20, 30), save_vimp_history = "last", num.trees = 30,
  num.threads = safe_num_threads(1)
)

# Print unadjusted and FDR-adjusted p-values together with the corresponding
# decisions
```

```
out5 <- shadow_vimp(
  data = mtcars, outcome_var = "vs",
  niters = c(10, 20, 30), to_show = "FDR", num.trees = 30,
  num.threads = safe_num_threads(1)
)

# Use per-variable p-values to decide in the final step whether a covariate
# is informative or not. Note that pooled p-values are always used in the
# pre-selection (first two steps).
out6 <- shadow_vimp(
  data = mtcars, outcome_var = "vs",
  niters = c(10, 20, 30), method = "per_variable", num.trees = 30,
  num.threads = safe_num_threads(1)
)

# Perform variable selection in a single step, without a pre-selection phase
out7 <- shadow_vimp(
  data = mtcars, outcome_var = "vs", alphas = c(0.05),
  niters = c(30), num.trees = 30,
  num.threads = safe_num_threads(1)
)
```

Index