Package: detectXOR (via r-universe)

October 30, 2026

Title XOR Pattern Detection and Visualization

Version 0.2.0

Description Provides tools for detecting XOR-like patterns in variable pairs in two-class data sets. Includes visualizations for pattern exploration and reporting capabilities with both text and HTML output formats.

License GPL-3

Encoding UTF-8

LazyData true

Imports dplyr (>= 1.1.0), ggplot2 (>= 3.4.0), ggh4x (>= 0.2.3), tibble (>= 3.1.8), reshape2 (>= 1.4.4), glue (>= 1.6.0), magrittr (>= 2.0.0), stats, ggthemes, DescTools (>= 0.99.50), utils, methods, grDevices, knitr, kableExtra, htmltools, base64enc

Suggests testthat (>= 3.0.0), rmarkdown, doParallel, foreach, parallel (>= 4.2.0), future (>= 1.28.0), future.apply (>= 1.10.0), pbmcapply (>= 1.5.0)

Config/testthat/edition 3

SystemRequirements GNU make

Depends R (>= 3.5.0)

URL https://github.com/JornLotsch/detectXOR

BugReports https://github.com/JornLotsch/detectXOR/issues

Config/roxygen2/version 8.1.1

Config/pak/sysreqs cmake libfontconfig1-dev libfreetype6-dev libfribidi-dev make libharfbuzz-dev libicu-dev libpng-dev libuv1-dev libxml2-dev libssl-dev libx11-dev zlib1g-dev

Repository https://jornlotsch.r-universe.dev

Date/Publication 2026-10-30 11:27:22 UTC

RemoteUrl https://github.com/jornlotsch/detectxor

RemoteRef HEAD

RemoteSha 8c0ddff3ca4a409617d8840a38f0296905eb8abf

Contents

detectXOR-package 2
detect_xor 3
generate_spaghetti_plot_from_results 6
generate_xor_reportConsole 8
generate_xor_reportHTML 8
generate_xy_plot_from_results 9
XOR_data 11

Index **13**

`detectXOR-package` *XOR Pattern Detection and Visualization*

Description

Provides tools for detecting XOR-like patterns in variable pairs in two-class data sets. Includes visualizations for pattern exploration and reporting capabilities with both text and HTML output formats.

Details

Core Features:

1. Statistical detection using chi-square tests and Kendall's tau
2. Spaghetti plots and xy plot for pattern visualization

Main Functions:

- `detect_xor`: Core detection algorithm
- `generate_spaghetti_plot_from_results`: Line plots
- `generate_xy_plot_from_results`: Plot for pattern visualization

Author(s)

Jorn Lotsch <j.lotsch@em.uni-frankfurt.de>

References

Methodological foundations:

- Pattern detection in machine learning
- Statistical dependency measures (Kendall's tau)

See Also

Useful links:

- https://github.com/JornLotsch/detectXOR
- Report bugs at https://github.com/JornLotsch/detectXOR/issues

Related packages:

- DescTools for statistical tests
- ggh4x for advanced plotting
- future for parallel processing

Examples

```
# Basic workflow with included dataset
data(XOR_data)

# Detect XOR patterns
results <- detect_xor(XOR_data, class_col = "class")

# Generate visualizations
generate_spaghetti_plot_from_results(
  results$results_df,
  XOR_data,
  class_col = "class"
)

generate_xy_plot_from_results(
  results$results_df,
  XOR_data,
  class_col = "class"
)
```

`detect_xor` *Detect XOR-like patterns in pairwise variable relationships*

Description

Screens all pairs of numeric variables in a two-class data set for XOR-like patterns. Each pair is partitioned into a 3 x 3 tertile grid, and the class composition of the four critical tiles of the default (corner tiles) and rotated (edge-center tiles) XOR manifestation is evaluated.

Usage

```
detect_xor(
  data,
  class_col = "class",
  check_tau = TRUE,
  compute_axes_parallel_significance = FALSE,
  p_threshold = 0.05,
  tau_threshold = 0.3,
  purity_threshold = 0.8,
  min_tile_share = 0.05,
  n_perm = 999,
  p_adjust_method = "maxT",
  split_method = "quantile",
  split_quantiles = c(1/3, 2/3),
  max_cores = 1,
  extreme_handling = "winsorize",
  winsor_limits = c(0.05, 0.95),
  scale_data = TRUE,
  use_complete = TRUE,
  balance_classes = FALSE,
  abs_diff_threshold = NULL
)
```

Arguments

`data`	Data frame containing variables and class column.
`class_col`	Character. Name of the class column (default: "class").
`check_tau`	Logical. Whether to require opposite class-wise Kendall tau correlations for a positive XOR call (default: TRUE).
`compute_axes_parallel_significance`	Logical. Whether to compute the four group-wise Wilcoxon tests along the axes of the critical tiles and require all of them to be significant for a positive XOR call (default: FALSE). The tests are not applied to pairs of binary variables.
`p_threshold`	Numeric. Significance level for the adjusted permutation p-value, the one-sided lower confidence bound of the XOR purity, and the Wilcoxon tests (default: 0.05).
`tau_threshold`	Numeric. Minimum absolute class-wise Kendall tau (default: 0.3).
`purity_threshold`	Numeric in (0.5, 1). Minimum XOR purity that the lower confidence bound must reach (default: 0.8).
`min_tile_share`	Numeric in [0, 0.5). Minimum share of a class' points in the critical tiles that each of its two expected tiles must contain (default: 0.05). Rejects incomplete XOR patterns with an empty or nearly empty quadrant, while tolerating unequal tile populations, e.g., for binary variables with unbalanced value frequencies.
`n_perm`	Integer. Number of class-label permutations for the null distribution (default: 999).

`p_adjust_method`
: Character. Multiple-testing adjustment across all variable pairs: "maxT" (Westfall-Young single-step max-T, controls the family-wise error rate; default) or any method of `stats::p.adjust` (e.g., "BH", "holm", "none").

`split_method`
: Character. Method for splitting tiles ("quantile" or "range") (default: "quantile").

`split_quantiles`
: Numeric vector of length 2. Lower and upper quantile of the tile boundaries for `split_method = "quantile"` (default: tertiles, c(1/3, 2/3)).

`max_cores`
: Integer. Maximum number of cores to use for parallel processing (default: 1).

`extreme_handling`
: Character. Method for handling extreme values: "winsorize", "trim" (remove rows with extreme values; "remove" is accepted as a synonym), or "none" (default: "winsorize").

`winsor_limits`
: Numeric vector of length 2. Lower and upper quantiles for winsorization or trimming (default: c(0.05, 0.95)).

`scale_data`
: Logical. Whether to scale variables to unit variance (default: TRUE).

`use_complete`
: Logical. Whether to use only complete cases (default: TRUE).

`balance_classes`
: Logical. Whether to randomly down-sample the larger class to the size of the smaller class before screening (default: FALSE). Recommended for marked class imbalance, under which the quantile-based tile boundaries can misrepresent rotated XOR patterns.

`abs_diff_threshold`
: Deprecated and ignored (used by detectXOR < 0.2.0).

Details

For a given XOR manifestation, the XOR rule assigns one class to two diagonally opposite critical tiles and the other class to the remaining two. The XOR purity is the balanced accuracy of this rule among the observations in the critical tiles, maximized over both label assignments (so that the result does not depend on class coding) and over the two manifestations. It is 0.5 for no XOR structure and 1 for a perfect XOR pattern.

A pair is called XOR-positive if all of the following hold:

1. the purity exceeds chance: the permutation p-value, adjusted for the number of variable pairs, is below `p_threshold`. All pairs share the same label permutations, which allows max-T adjustment;
2. the purity is large: its one-sided `1 - p_threshold` lower confidence bound is at least `purity_threshold`;
3. the pattern is complete: each class populates both of its expected tiles with at least `min_tile_share` of its critical-tile points;
4. if `check_tau`, the class-wise Kendall tau correlations have opposite signs and absolute values of at least `tau_threshold`;
5. if `compute_axes_parallel_significance`, all four group-wise Wilcoxon tests are significant (continuous variables only).

Criteria 4 and 5 are evaluated only for pairs meeting criteria 1 to 3. Permutations are drawn from the session's random number generator; call `set.seed()` beforehand for reproducible p-values. Parallel processing is used when `max_cores > 1` and the required packages are available.

Value

List containing:

- results_df: Data frame summarizing XOR detection results for all variable pairs
- pair_list: List of detailed results for each variable pair
- settings: List of the parameter values used

Author(s)

Jorn Lotsch, Alfred Ultsch

See Also

generate_spaghetti_plot_from_results, generate_xy_plot_from_results

Examples

```
data(XOR_data)
set.seed(42)
result <- detect_xor(XOR_data, class_col = "class", n_perm = 199)
result$results_df[, c("var1", "var2", "xor_shape_detected", "xor_pattern", "xor_purity")]
```

---

generate_spaghetti_plot_from_results

Generate XOR Spaghetti Plots

Description

Creates connected line plots for variable pairs showing XOR patterns.

Usage

```
generate_spaghetti_plot_from_results(
  results,
  data,
  class_col,
  scale_data = TRUE
)
```

Arguments

results	Either a data frame from detect_xor()$results_df or the full list object returned by detect_xor()
data	Original dataset containing variables and classes
class_col	Character string specifying the name of the class column
scale_data	Logical indicating whether to scale variables before plotting (default: TRUE)

Details

This function creates spaghetti plots (connected line plots) for variable pairs that have been flagged as showing XOR patterns by `detect_xor()`. The function automatically handles both original and rotated XOR patterns, applying the appropriate coordinate transformation when necessary.

The function accepts either the full results object returned by `detect_xor()` or just the `results_df` component extracted from it. Variable pairs are separated using "||" as the delimiter in plot labels.

If no XOR patterns are detected, an empty plot with an appropriate message is returned.

To save the plot, use `ggplot2::ggsave()` or other standard R plotting save methods.

Value

Returns a ggplot object. No files are saved automatically.

See Also

`detect_xor` for XOR pattern detection, `generate_xy_plot_from_results` for scatter plots

Examples

```
# Using full results object (recommended)
data(XOR_data)
results <- detect_xor(data = XOR_data, class_col = "class")
spaghetti_plot <- generate_spaghetti_plot_from_results(
  results = results,
  data = XOR_data,
  class_col = "class"
)

# Display the plot
print(spaghetti_plot)

# Save the plot if needed
# ggplot2::ggsave("my_spaghetti_plot.png", spaghetti_plot)

# Using extracted results_df (also works)
xy_plot <- generate_spaghetti_plot_from_results(
  results = results$results_df,
  data = XOR_data,
  class_col = "class"
)
```

`generate_xor_reportConsole` *Generate XOR Detection Report (Console-friendly)*

Description

Creates a report with formatted table and plots for XOR pattern detection results.

Usage

```
generate_xor_reportConsole(
  results,
  data,
  class_col,
  scale_data = TRUE,
  show_plots = TRUE,
  quantile_lines = c(1/3, 2/3),
  line_method = "quantile"
)
```

Arguments

`results`	Either a data frame from `detect_xor$results_df` or the full list returned by `detect_xor`.
`data`	Original dataset containing variables and classes.
`class_col`	Character specifying the class column name.
`scale_data`	Logical indicating whether to scale variables in plots. Default: TRUE.
`show_plots`	Logical indicating whether to display plots. Default: TRUE.
`quantile_lines`	Numeric vector of quantiles for reference lines in XY plots. Default: c(1/3, 2/3).
`line_method`	Method for boundary calculation ("quantile" or "range"). Default: "quantile".

Value

Invisibly returns a list containing the formatted table and plots (if generated).

`generate_xor_reportHTML` *Generate XOR Detection HTML Report*

Description

Creates a HTML report with formatted table and plots for XOR pattern detection results.

Usage

```
generate_xor_reportHTML(
  results,
  data,
  class_col,
  output_file = "xor_detection_report.html",
  open_browser = TRUE,
  scale_data = TRUE,
  quantile_lines = c(1/3, 2/3),
  line_method = "quantile"
)
```

Arguments

results	Either a data frame from detect_xor$results_df or the full list returned by detect_xor.
data	Original dataset containing variables and classes.
class_col	Character specifying the class column name.
output_file	Character specifying the output HTML file name. Default: "xor_detection_report.html".
open_browser	Logical indicating whether to open the report in browser automatically. Default: TRUE.
scale_data	Logical indicating whether to scale variables in plots. Default: TRUE.
quantile_lines	Numeric vector of quantiles for reference lines in XY plots. Default: c(1/3, 2/3).
line_method	Method for boundary calculation ("quantile" or "range"). Default: "quantile".

Value

Invisibly returns the file path of the generated HTML report.

---

generate_xy_plot_from_results

Generate XOR Scatter Plots

Description

Creates scatterplots with decision boundaries for variable pairs showing XOR patterns.

Usage

```
generate_xy_plot_from_results(
  results,
  data,
  class_col,
  scale_data = TRUE,
  quantile_lines = c(1/3, 2/3),
  line_method = "quantile"
)
```

Arguments

`results`	Either a data frame from `detect_xor()$results_df` or the full list object returned by `detect_xor()`
`data`	Original dataset containing variables and classes
`class_col`	Character string specifying the name of the class column
`scale_data`	Logical indicating whether to scale variables before plotting (default: TRUE)
`quantile_lines`	Numeric vector of length 2 specifying quantiles for reference lines (default: c(1/3, 2/3))
`line_method`	Character string specifying the boundary calculation method, either "quantile" or "range" (default: "quantile")

Details

This function creates scatter plots for variable pairs that have been flagged as showing XOR patterns by `detect_xor()`. The plots include dashed reference lines that help visualize the decision boundaries used in XOR pattern detection.

The function automatically handles both original and rotated XOR patterns, applying the appropriate coordinate transformation when necessary. Variable pairs are separated using "||" as the delimiter in plot labels.

The `line_method` parameter controls how reference lines are calculated:

- "quantile": Lines are placed at the specified quantiles of the data distribution
- "range": Lines divide the data range into three equal parts

If no XOR patterns are detected, an empty plot with an appropriate message is returned.

To save the plot, use `ggplot2::ggsave()` or other standard R plotting save methods.

Value

Returns a ggplot object. No files are saved automatically.

See Also

`detect_xor` for XOR pattern detection, `generate_spaghetti_plot_from_results` for spaghetti plots

Examples

```
# Using full results object (recommended)
data(XOR_data)
results <- detect_xor(data = XOR_data, class_col = "class")
xy_plot <- generate_xy_plot_from_results(
  results = results,
  data = XOR_data,
  class_col = "class"
)
```

```
# Display the plot
print(xy_plot)

# Using different boundary method
xy_plot_range <- generate_xy_plot_from_results(
  results = results,
  data = XOR_data,
  class_col = "class",
  line_method = "range"
)

# Save the plot if needed
# ggplot2::ggsave("my_xy_plot.png", xy_plot)

# Using extracted results_df (also works)
xy_plot_df <- generate_xy_plot_from_results(
  results = results$results_df,
  data = XOR_data,
  class_col = "class"
)
```

XOR_data *Synthetic XOR Pattern Dataset*

Description

Simulated classification dataset containing 400 observations with 5 features demonstrating XOR patterns, linear class differences, and random noise.

Usage

```
data("XOR_data")
```

Format

A data frame with 400 rows and 6 variables:

class Binary class labels (1 or 2)

Variable_A Normally distributed with subtle class difference (delta mu=0.25)

Variable_B High-variance normal distribution (sigma=3) with moderate class separation (delta mu=-0.7)

Variable_C XOR pattern component 1 (mu=3 vs 10 between classes)

Variable_D XOR pattern component 2 (mu=3 vs 10 between classes)

Variable_E Uniform noise (1-10)

Source

Synthetic data generated with rnorm() and runif()

Examples

```
data(XOR_data)
str(XOR_data)
summary(XOR_data)
```

Index

∗ **XOR**
    detectXOR-package, 2
    generate_spaghetti_plot_from_results, 6
    generate_xy_plot_from_results, 9
∗ **datasets**
    XOR_data, 11
∗ **machine learning**
    detectXOR-package, 2
∗ **package**
    detectXOR-package, 2
∗ **pattern detection**
    detectXOR-package, 2
∗ **scatter**
    generate_xy_plot_from_results, 9
∗ **spaghetti**
    generate_spaghetti_plot_from_results, 6
∗ **visualization**
    generate_spaghetti_plot_from_results, 6
    generate_xy_plot_from_results, 9

DescTools, *3*
detect_xor, *2*, 3, *7*, *10*
detectXOR-package, 2

future, *3*

generate_spaghetti_plot_from_results, *2*, 6, *6*, *10*
generate_xor_reportConsole, 8
generate_xor_reportHTML, 8
generate_xy_plot_from_results, *2*, *6*, *7*, 9
ggh4x, *3*

XOR_data, 11