Package 'chaidr'

August 8, 2026

Title CHAID and Exhaustive CHAID Decision Trees

Version 0.1.0

Description An implementation in base 'R' of the CHAID (Chi-squared Automatic Interaction Detection) decision tree algorithm of Kass (1980) <doi:10.2307/2986296> and the Exhaustive CHAID variant of Biggs, de Ville, and Suen (1991) <doi:10.1080/02664769100000005>, as specified in the 'IBM SPSS' Statistics Algorithms documentation. Supports nominal, ordinal (with floating missing category), and continuous predictors, and nominal, ordinal, and continuous response variables using Pearson chi-squared, Goodman row-effects, and one-way ANOVA F tests respectively. Includes prediction, rule extraction, gains and lift analysis, validation on holdout data, and visualization via base graphics, 'Graphviz' DOT, 'plotly', and conversion to 'partykit' objects.

License MIT + file LICENSE

Encoding UTF-8

Language en-GB

Depends R (>= 4.5)

Imports graphics, grDevices, stats, utils

Suggests DiagrammeR, ggparty, ggplot2, htmlwidgets, knitr, partykit, plotly, rmarkdown, rpart, spelling, testthat (>= 3.2.0), withr

VignetteBuilder knitr

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

URL https://github.com/morimotoosamu/chaidr, https://morimotoosamu.github.io/chaidr/

BugReports https://github.com/morimotoosamu/chaidr/issues

NeedsCompilation no

Author Osamu Morimoto [aut, cre, cph]

Maintainer Osamu Morimoto <galactic.supermarket@gmail.com>

Repository CRAN

Date/Publication 2026-08-08 12:30:12 UTC

Contents

`chaid` *Fit a CHAID or Exhaustive CHAID decision tree*

Description

Grows a CHAID (Chi-squared Automatic Interaction Detection) or Exhaustive CHAID decision tree following the IBM SPSS Statistics algorithm specification. Nominal (`factor`), ordinal (`ordered`) and continuous (`numeric`) predictors are supported; continuous predictors are discretised into quantile bins first. The response may be nominal (Pearson or likelihood-ratio chi-squared test), ordinal (Goodman row effects test) or continuous (one-way ANOVA F test).

Usage

```
chaid(
  formula,
  data,
  weights = NULL,
  freq = NULL,
  method = c("chaid", "exhaustive"),
  control = chaid_control(),
  costs = NULL,
  y_scores = NULL
)
```

Arguments

`formula`	A model formula of the form `response ~ predictors`.
`data`	A data frame containing the variables in the formula.

weights	Optional numeric vector of case weights. They only affect the estimation of expected cell frequencies; cases with missing, zero or negative weights are excluded.
freq	Optional numeric vector of frequency weights. They determine observed counts, degrees of freedom and node sizes. Non-integer values are rounded to the nearest integer (IBM specification).
method	"chaid" (default) for the Kass (1980) algorithm or "exhaustive" for Exhaustive CHAID (Biggs, de Ville and Suen, 1991).
control	A "chaid_control" object created by chaid_control().
costs	Optional misclassification cost matrix C[truth, pred] for categorical responses (same convention as the loss matrix of 'rpart': zero diagonal, non-negative entries, dimnames matching the response levels). When supplied, node predictions minimise expected cost instead of taking the majority class. As in SPSS, costs do not affect tree growing or the significance tests.
y_scores	Optional numeric vector of class scores for an ordinal response, in the order of levels(y). Defaults to the class ranks 1..J. Fixed at the start of tree growing and not re-ranked in subtables (IBM specification).

Details

Missing predictor values are handled as in SPSS: for ordinal predictors they form a floating category that may merge with any group, for nominal predictors they form an ordinary extra category. Cases with a missing response, missing/zero/negative weights, or all predictors missing are dropped before fitting.

Value

An object of class "chaid": a list with components call, method, control, response (name, type, levels, scores), predictors (internal coding of each predictor), nodes (list of node records with distribution, prediction and split information), costs, n (number of cases used) and n_dropped (number of excluded cases).

References

Kass, G. V. (1980). An exploratory technique for investigating large quantities of categorical data. *Applied Statistics*, 29(2), 119-127.

Biggs, D., de Ville, B., & Suen, E. (1991). A method of choosing multiway partitions for classification and decision trees. *Journal of Applied Statistics*, 18(1), 49-62.

See Also

chaid_control(), predict.chaid(), chaid_table(), chaid_rules(), plot.chaid()

Examples

```
fit <- chaid(Species ~ ., data = iris,
             control = chaid_control(min_parent = 30, min_child = 10))
print(fit)
predict(fit, head(iris))
```

chaid_as_party *Convert a CHAID tree to a partykit constparty*

Description

Converts a fitted "chaid" object to a partykit::constparty object so that the 'partykit' toolbox (plot(), print(), nodeapply(), 'ggparty', ...) can be used. Binned continuous predictors are represented as ordered factors of the bin interval labels, and all splits become index-type partykit::partysplit objects, so missing values ("<NA>" level) are routed explicitly and split rules display the interval labels.

Usage

```
chaid_as_party(x, data, weights = NULL, freq = NULL, ...)

## S3 method for class 'chaid'
as.party(obj, data, weights = NULL, freq = NULL, ...)
```

Arguments

x, obj	A fitted "chaid" object returned by chaid().
data	The data frame used to fit the tree (the chaid object does not store the data).
weights, freq	The case and frequency weights used in the fit, if any. Required to reproduce the case exclusions of the fit.
...	Ignored.

Details

The returned object is intended for visualisation and structural inspection. For predictions on new data use predict.chaid(); the predict() method of the party object expects the converted data representation, not the original one.

Value

A partykit::constparty object.

See Also

chaid(), predict.chaid()

Examples

```
fit <- chaid(Species ~ ., data = iris,
             control = chaid_control(min_parent = 30, min_child = 10))
pt <- chaid_as_party(fit, data = iris)
print(pt)
```

`chaid_control` *Control parameters for CHAID tree growing*

Description

Collects the algorithm parameters used by `chaid()`. The defaults match the defaults of the IBM SPSS Statistics user interface (alpha 0.05, Bonferroni adjustment on, no re-splitting, maximum depth 3, minimum parent size 100, minimum child size 50, 10 bins for continuous predictors).

Usage

```
chaid_control(
  alpha_merge = 0.05,
  alpha_split = 0.05,
  alpha_split_merge = 0.05,
  resplit = FALSE,
  bonferroni = TRUE,
  stat = c("pearson", "lr"),
  exhaustive_adjust = c("spss", "biggs"),
  max_depth = 3L,
  min_parent = 100,
  min_child = 50,
  min_segment = NULL,
  n_bins = 10L,
  epsilon = 0.001,
  max_iter = 100L,
 adjust_across = c("none", "bonferroni", "holm", "hochberg", "hommel", "BH", "BY")
)
```

Arguments

`alpha_merge`	Significance level for merging predictor categories. Category pairs whose test p-value exceeds this threshold are merged.
`alpha_split`	Significance level for splitting a node. A node is split only if the best (adjusted) p-value is at most this value.
`alpha_split_merge`	Significance level for re-splitting a merged compound category. Only used when `resplit = TRUE`.
`resplit`	Logical. If `TRUE`, compound categories consisting of three or more original categories are considered for a binary re-split during the merge step (SPSS "allow resplitting" option).
`bonferroni`	Logical. If `TRUE` (default), split p-values are Bonferroni-adjusted by the number of ways the predictor categories can be merged into the final groups.
`stat`	Chi-squared statistic for categorical responses: `"pearson"` (default) or `"lr"` (likelihood ratio).

exhaustive_adjust
: Bonferroni multiplier convention used by Exhaustive CHAID: "spss" (default) follows the IBM SPSS algorithm document, "biggs" follows Biggs, de Ville and Suen (1991).

max_depth
: Maximum tree depth (root has depth 0).

min_parent
: Minimum number of cases (frequency-weighted) a node must contain to be considered for splitting.

min_child
: Minimum number of cases (frequency-weighted) in each child node.

min_segment
: Optional minimum size for a merged category group during the merge step (SPSS algorithm step 7). Groups smaller than this are absorbed into the most similar allowable group. NULL (default) disables this step.

n_bins
: Number of quantile bins used to discretise continuous predictors.

epsilon
: Convergence tolerance for the iterative estimation of expected cell frequencies with case weights.

max_iter
: Maximum number of iterations for the same estimation.

adjust_across
: Multiple-comparison adjustment applied across predictors within a node before comparing against alpha_split. "none" (default, as in SPSS) or one of the stats::p.adjust() methods "bonferroni", "holm", "hochberg", "hommel", "BH", "BY". Note that the adjustment family is the node, not the whole tree, and that "holm" always yields the same tree as "bonferroni" because only the minimum adjusted p-value is compared with alpha_split.

Value

An object of class "chaid_control": a list of the validated parameter values, to be passed to the control argument of chaid().

See Also

chaid()

Examples

```
ctl <- chaid_control(max_depth = 2, min_parent = 20, min_child = 5)
fit <- chaid(Species ~ ., data = iris, control = ctl)
fit
```

---

chaid_dot *Render a CHAID tree as Graphviz DOT*

Description

chaid_dot() generates a publication-quality Graphviz DOT description of the tree using base R only. It can be rendered with chaid_graphviz() (which uses 'DiagrammeR', bundling viz.js so no Graphviz binary is needed) or written to a .gv file and rendered externally with dot -Tpng / dot -Tsvg.

Usage

```
chaid_dot(
  fit,
  palette = NULL,
  rankdir = "TB",
  label_len = 28,
  legend = TRUE,
  file = NULL
)

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

chaid_graphviz(fit, ...)
```

Arguments

fit	A fitted "chaid" object returned by chaid().
palette	Vector of class colours for categorical responses (default grDevices::hcl.colors(n, "Dark 3"), matching plot.chaid()). For continuous responses node fills use a white-to-steelblue gradient of the node means.
rankdir	Graph direction: "TB" (top to bottom, default) or "LR" (left to right).
label_len	Maximum number of characters for edge (split group) labels.
legend	Logical. Add a legend node with the class colours (categorical responses only).
file	Optional path; when given, the DOT source is also written there in UTF-8 for use with the external dot command.
x	A "chaid_dot" object.
...	For chaid_graphviz(), arguments passed on to chaid_dot(); ignored by print().

Value

For chaid_dot(), the DOT source as a character string of class "chaid_dot" (returned invisibly; its print() method outputs the source). For chaid_graphviz(), an 'htmlwidget' as returned by DiagrammeR::grViz().

See Also

plot.chaid(), chaid_plotly()

Examples

```
fit <- chaid(Species ~ ., data = iris,
             control = chaid_control(min_parent = 30, min_child = 10))
dot <- chaid_dot(fit)
```

chaid_gains *Gains and lift table for a CHAID tree*

Description

Sorts the terminal nodes by decreasing response rate (node mean for continuous responses) and computes cumulative gains and lift, corresponding to the SPSS gains table.

Usage

```
chaid_gains(fit, data = NULL, target = NULL, weights = NULL, freq = NULL)

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

## S3 method for class 'chaid_gains'
plot(x, type = c("gains", "lift"), ...)
```

Arguments

fit	A fitted "chaid" object returned by chaid().
data	Optional data frame. If NULL (default), node statistics from training are used. If supplied, cases are routed with predict.chaid() and the table is recomputed, e.g. for evaluation on holdout data.
target	Response level of interest for categorical responses. For a binary response the second level is used by default; for three or more levels target is required.
weights, freq	Case and frequency weights for data, when supplied.
x	A "chaid_gains" object.
...	For plot(), further arguments passed to graphics::plot(); ignored by print().
type	"gains" (default) draws the cumulative gains curve (percentage of cases vs. percentage of captured targets, diagonal = random), "lift" draws the cumulative lift curve (1 = random).

Value

An object of class "chaid_gains", a list with the gains table (table), the target level, the response type (ytype), the overall rate (overall) and the data source ("training" or "newdata"). print() and plot() methods are available.

See Also

chaid_table(), chaid_validate()

Examples

```
fit <- chaid(Species ~ ., data = iris,
             control = chaid_control(min_parent = 30, min_child = 10))
g <- chaid_gains(fit, target = "virginica")
print(g)
plot(g)
plot(g, type = "lift")
```

`chaid_importance` *Heuristic variable importance for a CHAID tree*

Description

SPSS defines no official importance measure for CHAID, and the chi-squared and F statistics of different splits are not directly comparable because their degrees of freedom differ. This function therefore uses a p-value based heuristic: for each predictor, `importance` is the sum over its splits of `(node size / root size) * (-log10(adjusted p-value))`, i.e. a variable is important when it splits large nodes with strong significance.

Usage

```
chaid_importance(fit)
```

Arguments

`fit`	A fitted `"chaid"` object returned by `chaid()`.

Value

A data frame with one row per predictor actually used for a split, sorted by decreasing importance: `variable`, `n_splits`, `min_p_adj`, `importance` and `importance_pct` (share of the total importance in percent).

See Also

`chaid()`, `chaid_table()`

Examples

```
fit <- chaid(Species ~ ., data = iris,
             control = chaid_control(min_parent = 30, min_child = 10))
chaid_importance(fit)
```

`chaid_plotly` *Interactive CHAID tree plot with plotly*

Description

Draws the tree as an interactive 'plotly' htmlwidget with zoom, pan and hover information (reaching rule, class distribution and split details for each node). The widget can be embedded directly in R Markdown documents or saved as HTML.

Usage

```
chaid_plotly(fit, palette = NULL, label_len = 20, ...)
```

Arguments

`fit`	A fitted `"chaid"` object returned by `chaid()`.
`palette`	Vector of class colours for categorical responses (default `grDevices::hcl.colors(n, "Dark 3")`, matching `plot.chaid()`).
`label_len`	Maximum number of characters for edge (split group) labels.
`...`	Ignored.

Value

A 'plotly' htmlwidget.

See Also

`plot.chaid()`, `chaid_dot()`

Examples

```
fit <- chaid(Species ~ ., data = iris,
             control = chaid_control(min_parent = 30, min_child = 10))
chaid_plotly(fit)
```

chaid_rules *Extract decision rules from a CHAID tree*

Description

Returns the condition that a case must satisfy to reach each terminal node (or each of the requested nodes) as a rule string.

Usage

```
chaid_rules(fit, nodes = NULL, format = c("text", "sql", "r"))
```

Arguments

fit	A fitted "chaid" object returned by chaid().
nodes	Integer vector of node ids. Defaults to all terminal nodes.
format	"text" (default) for human-readable conditions, "sql" for SQL WHERE clauses, or "r" for R logical expressions that reproduce the node assignment exactly when evaluated against the data.

Value

A data frame with columns node (integer id) and rule (character).

See Also

chaid(), chaid_table()

Examples

```
fit <- chaid(Species ~ ., data = iris,
             control = chaid_control(min_parent = 30, min_child = 10))
chaid_rules(fit)
chaid_rules(fit, format = "sql")
```

chaid_table *Summary table of CHAID terminal nodes*

Description

Builds a segment summary of the terminal nodes. The first row is the root node, which serves as the baseline (index 100). For categorical responses the table contains the predicted class and the class shares, plus, when target is given, the response rate and the index value (response rate relative to the root, times 100). For continuous responses it contains the node mean, standard deviation and the index of the mean.

Usage

```
chaid_table(fit, target = NULL)
```

Arguments

fit	A fitted "chaid" object returned by chaid().
target	Optional response level of interest (categorical responses only). Adds response_rate and index columns.

Details

Note that n is the sum of frequency weights while pct_n and the class shares are computed with case weights included; the two scales coincide unless case weights are used.

Value

A data frame with one row for the root followed by one row per terminal node, including the reaching rule in the rule column.

See Also

chaid_rules(), chaid_gains(), chaid_importance()

Examples

```
fit <- chaid(Species ~ ., data = iris,
             control = chaid_control(min_parent = 30, min_child = 10))
chaid_table(fit)
chaid_table(fit, target = "virginica")
```

---

chaid_validate *Evaluate a CHAID tree on holdout data*

Description

Routes newdata down the fitted tree and compares, for each terminal node, the node share and the response rate (node mean for continuous responses) between training and validation data. Useful to check whether the segments replicate on new data, i.e. whether the tree is overfitting.

Usage

```
chaid_validate(fit, newdata, weights = NULL, freq = NULL)

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

Arguments

fit	A fitted "chaid" object returned by chaid().
newdata	A data frame of validation cases containing the response and all predictors.
weights, freq	Case and frequency weights for newdata.
x	A "chaid_validation" object.
...	Ignored.

Value

An object of class "chaid_validation": a list with nodes (per-node comparison data frame with train_pct_n, test_pct_n, train_rate, test_rate and diff_rate), overall (accuracy for categorical responses; RMSE and R-squared for continuous ones), n_test and ytype. A print() method is available.

See Also

chaid_gains(), predict.chaid()

Examples

```
set.seed(1)
idx <- sample(nrow(iris), 100)
fit <- chaid(Species ~ ., data = iris[idx, ],
             control = chaid_control(min_parent = 30, min_child = 10))
chaid_validate(fit, iris[-idx, ])
```

---

plot.chaid *Plot a CHAID tree with base graphics*

Description

Draws the tree top-down using base graphics only. For categorical responses each node shows the predicted class, its share, the node size and optionally a horizontal bar of the class distribution. Edges are labelled with the (possibly merged) predictor categories.

Usage

```
## S3 method for class 'chaid'
plot(
  x,
  cex = 0.8,
  label_len = 22,
  palette = NULL,
  show_bar = TRUE,
  main = NULL,
  ...
)
```

Arguments

x	A fitted "chaid" object returned by chaid().
cex	Base character expansion for node and edge labels.
label_len	Maximum number of characters for edge (split group) labels; longer labels are truncated.
palette	Vector of class colours for categorical responses. Defaults to grDevices::hcl.colors(n, "Dark 3").
show_bar	Logical. Draw a class distribution bar inside each node (categorical responses only).
main	Plot title. Defaults to the response name and method.
...	Ignored.

Value

The fitted object, invisibly.

See Also

chaid(), chaid_dot() for publication-quality Graphviz output, chaid_plotly() for an interactive version.

Examples

```
fit <- chaid(Species ~ ., data = iris,
             control = chaid_control(min_parent = 30, min_child = 10))
plot(fit)
```

---

predict.chaid *Predict from a fitted CHAID tree*

Description

Routes the rows of newdata down the tree and returns predictions. Factor levels unseen during training, and codes that did not occur in a node when it was split, are routed to the child with the largest node size (for ordinal predictors, to the group with the smallest code distance); a warning is issued when unseen levels are detected.

Usage

```
## S3 method for class 'chaid'
predict(object, newdata, type = c("response", "prob", "node"), ...)
```

Arguments

object	A fitted "chaid" object returned by chaid().
newdata	A data frame containing all predictor variables used in the fit.
type	"response" (default) returns predicted classes (or means for a continuous response), "prob" returns a matrix of class probabilities (categorical responses only), "node" returns the terminal node id for each row.
...	Ignored.

Value

Depending on type: a factor (or numeric vector) of predictions, a numeric matrix of class probabilities with one column per response level, or an integer vector of node ids.

See Also

chaid()

Examples

```
fit <- chaid(Species ~ ., data = iris,
             control = chaid_control(min_parent = 30, min_child = 10))
predict(fit, head(iris))
predict(fit, head(iris), type = "prob")
predict(fit, head(iris), type = "node")
```

---

print.chaid *Print and summarise a CHAID tree*

Description

print() displays the tree structure with one line per node showing the prediction, node size and split information. summary() additionally reports the control settings and a risk estimate on the training data (misclassification rate or expected misclassification cost for categorical responses, weighted within-node variance for continuous responses).

Usage

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

## S3 method for class 'chaid'
summary(object, ...)
```

Arguments

x, object	A fitted "chaid" object returned by chaid().
...	Ignored.

Value

The fitted object, invisibly.

See Also

chaid()

Examples

```
fit <- chaid(Species ~ ., data = iris,
             control = chaid_control(min_parent = 30, min_child = 10))
print(fit)
summary(fit)
```

Index