# Package 'FFTrees'

January 6, 2023

**Type** Package

**Title** Generate, Visualise, and Evaluate Fast-and-Frugal Decision Trees

**Version** 1.8.0

**Date** 2023-01-06

**Maintainer** Hansjoerg Neth <h.neth@uni.kn>

**Description** Create, visualize, and test fast-and-frugal decision trees (FFTs) using the algorithms and methods described in Phillips, Neth, Woike & Gaissmaier (2017).
FFTs are simple and transparent decision trees for solving binary classification problems.
FFTs can be preferable to more complex algorithms because they require very little information, are easy to communicate, and are robust against overfitting.

**LazyData** true

**Encoding** UTF-8

**Depends** R(>= 3.5.0)

**Imports** caret, rpart, randomForest, e1071, cli, graphics, progress,
scales, dplyr, knitr, magrittr, stringr, testthat, tibble,
tidyselect

**Suggests** rmarkdown, spelling

**License** CC0

**URL** https://CRAN.R-project.org/package=FFTrees,
https://github.com/ndphillips/FFTrees/

**BugReports** https://github.com/ndphillips/FFTrees/issues

**VignetteBuilder** knitr

**RoxygenNote** 7.2.1

**Language** en-US

**NeedsCompilation** no

**Author** Nathaniel Phillips [aut] (<https://orcid.org/0000-0002-8969-7013>),
Hansjoerg Neth [aut, cre] (<https://orcid.org/0000-0001-5427-3141>),
Jan Woike [aut] (<https://orcid.org/0000-0002-6816-121X>),
Wolfgang Gaissmaier [aut] (<https://orcid.org/0000-0001-6273-178X>)

**Repository** CRAN

**Date/Publication** 2023-01-06 09:00:18 UTC

# R topics documented:

## `add_stats` *Add decision statistics to data (containing counts of a 2x2 contingency table)*

### Description

`add_stats` assumes the input of essential 2x2 frequency counts (as a data frame `data` with variable names `"hi"`, `"fa"`, `"mi"`, and `"cr"`) and uses them to compute various decision accuracy measures.

### Usage

```
add_stats(
  data,
  sens.w = 0.5,
  cost.each = NULL,
  cost.outcomes = list(hi = 0, fa = 1, mi = 1, cr = 0)
)
```

### Arguments

`data` A data frame with (integer) values named `"hi"`, `"fa"`, `"mi"`, and `"cr"`.

`sens.w` numeric. Sensitivity weight (for computing weighted accuracy, `wacc`). Default: `sens.w = .50`.

`cost.each` numeric. An optional fixed cost added to all outputs (e.g.; the cost of the cue).

`cost.outcomes` list. A list of length 4 named `"hi"`, `"fa"`, `"mi"`, `"cr"`, and specifying the costs of a hit, false alarm, miss, and correct rejection, respectively. E.g.; `cost.outcomes = listc("hi" = 0, "fa" = 10, "mi" = 20, "cr" = 0)` means that a false alarm and miss cost 10 and 20 units, respectively, while correct decisions incur no costs.

### Details

Providing numeric values for `cost.each` (as a vector) and `cost.outcomes` (as a named list) allows computing cost information for the counts of corresponding classification decisions.

### Value

A data frame with variables of computed accuracy and cost measures (but dropping inputs).

## blood *Blood donation data*

### Description

Blood donation data

### Usage

```
blood
```

### Format

A data frame containing 748 rows and 5 columns.

**recency** Months since last donation

**frequency** Total number of donations

**total** Total blood donated in c.c.

**time** Months since first donation

**donation.crit** Did he/she donated blood in March 2007?

...

### Source

https://archive.ics.uci.edu/ml/datasets/Blood+Transfusion+Service+Center

### See Also

Other datasets: `breastcancer`, `car`, `contraceptive`, `creditapproval`, `fertility`, `forestfires`, `heart.cost`, `heart.test`, `heart.train`, `heartdisease`, `iris.v`, `mushrooms`, `sonar`, `titanic`, `voting`, `wine`

## breastcancer *Physiological data of patients tested for breast cancer*

### Description

Physiological data of patients tested for breast cancer

### Usage

```
breastcancer
```

### Format

A data frame containing 699 patients (rows) and 9 variables (columns).

**thickness** Clump Thickness

**cellsize.unif** Uniformity of Cell Size

**cellshape.unif** Uniformity of Cell Shape

**adhesion** Marginal Adhesion

**epithelial** Single Epithelial Cell Size

**nuclei.bare** Bare Nuclei

**chromatin** Bland Chromatin

**nucleoli** Normal Nucleoli

**mitoses** Mitoses

**diagnosis** Is cancer present? TRUE or FALSE

...

### Source

https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Original)

### See Also

Other datasets: `blood`, `car`, `contraceptive`, `creditapproval`, `fertility`, `forestfires`, `heart.cost`, `heart.test`, `heart.train`, `heartdisease`, `iris.v`, `mushrooms`, `sonar`, `titanic`, `voting`, `wine`

---

## `car` *Car acceptability data*

---

### Description

A dataset on car evaluations based on basic features, derived from a simple hierarchical decision model.

### Usage

```
car
```

### Format

A data frame containing 1728 cars (rows) and 7 variables (columns).

**buying.price** Numeric

**maint.price** Factor

**doors** Factor

**persons** Numeric

**luggage** Numeric

**safety** Factor

**acceptability** Factor

...

## Details

The (yet to be binarized) criterion variable is a car's `acceptability` rating.

## Source

http://archive.ics.uci.edu/ml/datasets/Car+Evaluation

## References

Bohanec, M., Rajkovic, V. (1990): Expert system for decision making. Sistemica 1 (1), pp. 145-157.

## See Also

Other datasets: `blood`, `breastcancer`, `contraceptive`, `creditapproval`, `fertility`, `forestfires`, `heart.cost`, `heart.test`, `heart.train`, `heartdisease`, `iris.v`, `mushrooms`, `sonar`, `titanic`, `voting`, `wine`

---

# `classtable` *Compute classification statistics for binary prediction and criterion (e.g.; truth) vectors*

---

## Description

The main input are 2 logical vectors of prediction and criterion values.

## Usage

```
classtable(
  prediction_v = NULL,
  criterion_v = NULL,
  sens.w = NULL,
  cost.v = NULL,
  correction = 0.25,
  cost.outcomes = list(hi = 0, fa = 1, mi = 1, cr = 0),
  na_prediction_action = "ignore"
)
```

## Arguments

prediction_v  logical. A logical vector of predictions.

criterion_v  logical. A logical vector of (TRUE) criterion values.

sens.w  numeric. Sensitivity weight parameter (from 0 to 1, for computing wacc). Default: sens.w = NULL (to enforce that actual value is being passed by the calling function).

cost.v  list. An optional list of additional costs to be added to each case.

correction  numeric. Correction added to all counts for calculating dprime.

cost.outcomes  list. A list of length 4 with names 'hi', 'fa', 'mi', and 'cr' specifying the costs of a hit, false alarm, miss, and correct rejection, respectively. For instance, cost.outcomes = listc("hi" = 0, "fa" = 10, "mi" = 20, "cr" = 0) means that a false alarm and miss cost 10 and 20, respectively, while correct decisions have no cost.

na_prediction_action

What happens when no prediction is possible? (experimental).

## Details

The primary confusion matrix is computed by confusionMatrix of the **caret** package.

# comp_pred *A wrapper for competing classification algorithms.*

## Description

comp_pred provides the main wrapper for running alternative classification algorithms, such as CART (rpart::rpart), logistic regression (glm), support vector machines (svm::svm), and random forests (randomForest::randomForest).

## Usage

```
comp_pred(
  formula,
  data.train,
  data.test = NULL,
  algorithm = NULL,
  model = NULL,
  sens.w = NULL,
  new.factors = "exclude"
)
```

### Arguments

| | |
|---|---|
| formula | A formula (usually x$formula, for an FFTrees object x). |
| data.train | A training dataset (as data frame). |
| data.test | A testing dataset (as data frame). |
| algorithm | character string. An algorithm in the set: "lr" – logistic regression; "rlr" – regularized logistic regression; "cart" – decision trees; "svm" – support vector machines; "rf" – random forests. |
| model | model. An optional existing model, applied to the test data. |
| sens.w | Sensitivity weight parameter (from 0 to 1, required to compute wacc). |
| new.factors | string. What should be done if new factor values are discovered in the test set? "exclude" = exclude (i.e.; remove these cases), "base" = predict the base rate of the criterion. |

## contraceptive *Contraceptive use data*

### Description

A subset of the 1987 National Indonesia Contraceptive Prevalence Survey.

### Usage

```
contraceptive
```

### Format

A data frame containing 1473 rows and 10 columns.

**wife.age** Numeric

**wife.edu** Factor

**hus.ed** Factor

**children** Numeric

**wife.rel** Numeric

**wife.work** Factor

**hus.occ** Factor

**sol** Factor

**media** Numeric

**cont.crit** numeric

...

### Details

The samples describe married women who were either not pregnant or do not know if they were at the time of interview.

The problem consists in predicting a woman's current contraceptive method choice (here: binarized `cont.crit`) based on her demographic and socio-economic characteristics.

### Source

https://archive.ics.uci.edu/ml/datasets/Contraceptive+Method+Choice

### See Also

Other datasets: `blood`, `breastcancer`, `car`, `creditapproval`, `fertility`, `forestfires`, `heart.cost`, `heart.test`, `heart.train`, `heartdisease`, `iris.v`, `mushrooms`, `sonar`, `titanic`, `voting`, `wine`

---

## `creditapproval` *Credit approval data*

---

### Description

Credit approval data

### Usage

```
creditapproval
```

### Format

A data frame containing 690 rows and 15 columns

### Source

https://archive.ics.uci.edu/ml/datasets/Credit+Approval

### See Also

Other datasets: `blood`, `breastcancer`, `car`, `contraceptive`, `fertility`, `forestfires`, `heart.cost`, `heart.test`, `heart.train`, `heartdisease`, `iris.v`, `mushrooms`, `sonar`, `titanic`, `voting`, `wine`

## `fact_clean` *Clean factor variables in prediction data*

### Description

Clean factor variables in prediction data

### Usage

```
fact_clean(data.train, data.test, show.warning = T)
```

### Arguments

| | |
|---|---|
| `data.train` | A training dataset |
| `data.test` | A testing dataset |
| `show.warning` | logical |

## `fertility` *Fertility data*

### Description

This dataset describes a sample of 100 volunteers providing a semen sample that was analyzed according to the WHO 2010 criteria.

### Usage

```
fertility
```

### Format

A data frame containing 100 rows and 10 columns.

### Details

Sperm concentration are related to socio-demographic data, environmental factors, health status, and life habits.

The binary criterion variable is `diagnosis`: Normal (N) vs. altered (O).

### Source

https://archive.ics.uci.edu/ml/datasets/Fertility

## See Also

Other datasets: blood, breastcancer, car, contraceptive, creditapproval, forestfires, heart.cost, heart.test, heart.train, heartdisease, iris.v, mushrooms, sonar, titanic, voting, wine

---

## FFTrees *Main function to create and apply fast-and-frugal trees (FFTs)*

---

### Description

FFTrees is the workhorse function of the **FFTrees** package for creating fast-and-frugal trees (FFTs).

FFTs are decision algorithms for solving binary classification tasks, i.e., they predict the values of a binary criterion variable based on 1 or multiple predictor variables (cues).

Using FFTrees on data usually generates a range of FFTs and corresponding summary statistics (as an FFTrees object) that can then be printed, plotted, and examined further.

The criterion and predictor variables are specified in formula notation. Based on the settings of data and data.test, FFTs are trained on a (required) training dataset and tested on an (optional) test dataset.

If an existing FFTrees object object or tree.definitions are provided as inputs, no new FFTs are created. When both arguments are provided, tree.definitions take priority over the FFTs in an existing object. Specifically,

- If tree.definitions are provided, these are assigned to the FFTs of x.
- If no tree.definitions are provided, but an existing FFTrees object object is provided, the trees from object are assigned to the FFTs of x.

Create and evaluate fast-and-frugal trees (FFTs).

### Usage

```
FFTrees(
  formula = NULL,
  data = NULL,
  data.test = NULL,
  algorithm = "ifan",
  max.levels = NULL,
  sens.w = 0.5,
  cost.outcomes = NULL,
  cost.cues = NULL,
  stopping.rule = "exemplars",
  stopping.par = 0.1,
  goal = NULL,
  goal.chase = NULL,
  goal.threshold = "bacc",
  numthresh.method = "o",
  numthresh.n = 10,
  decision.labels = c("False", "True"),
  main = NULL,
  train.p = 1,
  rounding = NULL,
  repeat.cues = TRUE,
  my.tree = NULL,
  object = NULL,
  tree.definitions = NULL,
  do.comp = TRUE,
  do.cart = TRUE,
  do.lr = TRUE,
  do.rf = TRUE,
  do.svm = TRUE,
  quiet = FALSE,
  comp = NULL,
  force = NULL,
  rank.method = NULL,
  store.data = NULL,
  verbose = NULL
)
```

## Arguments

formula
: formula. A formula specifying a binary criterion variable (as logical) as a function of 1 or more predictor variables (cues).

data
: dataframe. A training dataset.

data.test
: dataframe. An optional testing dataset with the same structure as data.

algorithm
: character. The algorithm used to create FFTs. Can be 'ifan', 'dfan'.

max.levels
: integer. The maximum number of levels considered for the trees. Because all permutations of exit structures are considered, the larger max.levels is, the more trees will be created.

sens.w
: numeric. A number from 0 to 1 indicating how to weight sensitivity relative to specificity when goal = 'wacc'. Default: sens.w = .50.

cost.outcomes
: A list of length 4 specifying the cost value for one of the 4 possible classification outcomes. The list elements must have names 'hi', 'fa', 'mi', and 'cr' (for specifying the costs of a hit, false alarm, miss, and correct rejection, respectively) and provide a numeric cost value. E.g.; cost.outcomes = listc("hi" = 0, "fa" = 10, "mi" = 20, "cr" = 0) means that a false alarm and miss cost 10 and 20, respectively, while correct decisions have no costs.

cost.cues
: A list containing the cost of each cue (in some unit). Each list element must have a name corresponding to a cue (i.e., a column in data), and should be a single (positive) number. Cues not present in cost.cues are assumed to have no costs (i.e., a cost value of 0).

stopping.rule
: character. A string indicating the method to stop growing trees. "levels" means the tree grows until a certain level; "exemplars" means the tree grows

until a certain number of unclassified exemplars remain; "statdelta" means the tree grows until the change in the criterion statistic is less than a specified level.

stopping.par numeric. A number indicating the parameter for the stopping rule. For stopping.rule "levels", this is the number of levels. For stopping rule "exemplars", this is the smallest percentage of exemplars allowed in the last level.

goal character. A string indicating the statistic to maximize when selecting final trees: "acc" = overall accuracy, "bacc" = balanced accuracy, "wacc" = weighted accuracy.

goal.chase character. A string indicating the statistic to maximize when constructing trees: "acc" = overall accuracy, "bacc" = balanced accuracy, "wacc" = weighted accuracy, "cost" = cue costs.

goal.threshold character. A string indicating the statistic to maximize when calculating cue thresholds: "acc" = overall accuracy, "bacc" = balanced accuracy, "wacc" = weighted accuracy. Default: goal.threshold = "bacc".

numthresh.method

character. How should thresholds for numeric cues be determined? "o" will optimize thresholds, while "m" will always use the median.

numthresh.n integer. Number of numeric thresholds to try.

decision.labels

string. A vector of strings of length 2 indicating labels for negative and positive cases. E.g.; decision.labels = c("Healthy", "Diseased").

main string. An optional label for the dataset. Passed on to other functions, like plot.FFTrees, and print.FFTrees.

train.p numeric. What percentage of the data to use for training when data.test is not specified? For example, train.p = .50 will randomly split data into a 50% training set and a 50% test set. The default of train.p = 1 uses all data for training.

rounding integer. An integer indicating digit rounding for non-integer numeric cue thresholds. The default of rounding = NULL implies no rounding. A value of 0 rounds all possible thresholds to the nearest integer, 1 rounds to the nearest decade (.10), etc.

repeat.cues logical. May cues occur multiple times within a tree? Default: repeat.cues = TRUE.

my.tree An optional character string. A a verbal description of an FFT, i.e., an FFT in words. For example, my.tree = "If age > 20, predict TRUE. If sex = {m}, predict FALSE. Otherwise, predict TRUE."

object An optional existing FFTrees object. When specified, no new FFTs are fitted, but existing trees are applied to data and data.test. When formula, data or data.test are not specified, the current values of object are used.

tree.definitions

An optional data.frame of hard-coded FFT definitions (in the format of x$trees$definitions of an FFTrees object x). If specified, no new FFTs are fitted, but the tree definitions provided are used to re-evaluate the current FFTrees object.

do.comp, do.cart, do.lr, do.rf, do.svm
: logical. Should alternative algorithms be used for comparison? All options set to TRUE by default. Available options are: cart = regular (non-frugal) trees with **rpart**; lr = logistic regression with **glm**; rf = random forests with **randomForest**; svm = support vector machines with **e1071**. Specifying do.comp = FALSE sets all available options to FALSE.

quiet
: logical. Should progress reports be suppressed? Setting quiet = FALSE is helpful for diagnosing errors. Default: quiet = FALSE (i.e., show progress).

comp, force, rank.method, store.data, verbose
: Deprecated arguments (unused or replaced, to be retired in future releases).

## Value

An FFTrees object with the following elements:

**criterion_name** The name of the binary criterion variable (as character).

**cue_names** The names of all potential predictor variables (cues) in the data (as character).

**formula** The formula specified when creating the FFTs.

**trees** A list of FFTs created, with further details contained in n, best, definitions, inwords, stats, level_stats, and decisions.

**data** The original training and test data (if available).

**params** A list of defined control parameters (e.g.; algorithm, goal, sens.w, as well as various thresholds, stopping rule, and cost parameters).

**competition** Models and classification statistics for competitive classification algorithms: Logistic regression, CART, random forests RF, and SVM.

**cues** A list of cue information, with further details contained in thresholds and stats.

## See Also

print.FFTrees for printing FFTs; plot.FFTrees for plotting FFTs; summary.FFTrees for summarizing FFTs; inwords for obtaining a verbal description of FFTs; showcues for plotting cue accuracies.

## Examples

```
# 1. Create fast-and-frugal trees (FFTs) for heart disease:
heart.fft <- FFTrees(formula = diagnosis ~ .,
                     data = heart.train,
                     data.test = heart.test,
                     main = "Heart Disease",
                     decision.labels = c("Healthy", "Diseased")
                     )

# 2. Print a summary of the result:
heart.fft  # same as:
# print(heart.fft, data = "train", tree = "best.train")
# 3. Plot an FFT applied to training data:
plot(heart.fft)  # same as:
# plot(heart.fft, what = "all", data = "train", tree = "best.train")

# 4. Apply FFT to (new) testing data:
plot(heart.fft, data = "test")            # predict for Tree 1
plot(heart.fft, data = "test", tree = 2)  # predict for Tree 2

# 5. Predict classes and probabilities for new data:
predict(heart.fft, newdata = heartdisease)
predict(heart.fft, newdata = heartdisease, type = "prob")

# 6. Create a custom tree (from verbal description) with my.tree:
custom.fft <- FFTrees(
  formula = diagnosis ~ .,
  data = heartdisease,
  my.tree = "If age < 50, predict False.
             If sex = 1, predict True.
             If chol > 300, predict True, otherwise predict False.",
  main = "My custom FFT")

# Plot the (pretty bad) custom tree:
plot(custom.fft)
```

## `FFTrees.guide` *Open the* **FFTrees** *package guide*

### Description

Open the **FFTrees** package guide

### Usage

```
FFTrees.guide()
```

### Value

No return value, called for side effects.

## `fftrees_cuerank` *Calculate thresholds that optimize some statistic (goal) for cues in data*

### Description

`fftrees_cuerank` takes an `FFTrees` object `x` and optimizes its `goal.threshold` (from `x$params`) for all cues in a dataset `newdata` (of some `data` type).

### Usage

```
fftrees_cuerank(x = NULL, newdata = NULL, data = "train", rounding = NULL)
```

### Arguments

| | |
|---|---|
| x | An FFTrees object. |
| newdata | The dataset to with cues to be ranked (as data frame). |
| data | The type of data with cues to be ranked (as character: 'train', 'test', or 'dynamic'). Default: data = 'train'. |
| rounding | Number of digits used to round (as integer). Default: rounding = NULL. |

### Details

fftrees_cuerank creates a data frame cuerank_df that is added to x$cues$stats.

Note that the cue directions and thresholds computed by **FFTrees** always predict positive criterion values (i.e., TRUE or signal, rather than FALSE or noise). Using these thresholds for negative exits (i.e., for predicting instances of FALSE or noise) usually requires a reversal (e.g., negating cue direction).

fftrees_cuerank is called (twice) by the fftrees_grow_fan algorithm to grow fast-and-frugal trees (FFTs).

### Value

A modified FFTrees object (with cue rank information for the current data type in x$cues$stats).

## fftrees_ffttowords *Describe a fast-and-frugal tree (FFT) in words*

### Description

fftrees_ffttowords provides a verbal description of tree definition (as defined in an FFTrees object). Thus, fftrees_ffttowords translates an abstract FFT definition into natural language output.

fftrees_ffttowords is the complement function to fftrees_wordstofftrees, which parses a verbal description of an FFT into the abstract tree definition of an FFTrees object.

The final sentence (or tree node) of the FFT's description always predicts positive criterion values (i.e., TRUE instances) first, before predicting negative criterion values (i.e., FALSE instances). Note that this may require a reversal of exit directions, if the final cue predicted FALSE instances.

Note that the cue directions and thresholds computed by **FFTrees** always predict positive criterion values (i.e., TRUE or signal, rather than FALSE or noise). Using these thresholds for negative exits (i.e., for predicting instances of FALSE or noise) usually requires a reversal (e.g., negating cue direction).

### Usage

```
fftrees_ffttowords(x = NULL, mydata = "train", digits = 2)
```

### Arguments

| | |
|---|---|
| `x` | An `FFTrees` object created with `FFTrees`. |
| `mydata` | The type of data to which a tree is being applied (as character string "train" or "test"). Default: `mydata = "train"`. |
| `digits` | How many digits to round numeric values (as integer)? |

### Value

A modified `FFTrees` object `x` with `x$trees$inwords` containing a list of string vectors.

### See Also

`fftrees_wordstofftrees` for converting a verbal description of an FFT into an `FFTrees` object; `fftrees_create` for creating `FFTrees` objects; `fftrees_grow_fan` for creating FFTs by applying algorithms to data; `print.FFTrees` for printing FFTs; `plot.FFTrees` for plotting FFTs; `summary.FFTrees` for summarizing FFTs; `FFTrees` for creating FFTs from and applying them to data.

### Examples

```
heart.fft <- FFTrees(diagnosis ~ .,
  data = heartdisease,
  decision.labels = c("Healthy", "Disease")
)

inwords(heart.fft)
```

---

## `fftrees_fitcomp` *Fit competitive algorithms*

---

### Description

`fftrees_fitcomp` fits competitive algorithms for binary classification tasks (e.g., LR, CART, RF, SVM) to the data and parameters specified in an `FFTrees` object.

`fftrees_fitcomp` is called by the main `FFTrees` function when creating FFTs from and applying them to data (unless `do.comp = FALSE`).

### Usage

```
fftrees_fitcomp(x)
```

### Arguments

| | |
|---|---|
| x | An FFTrees object. |

### See Also

FFTrees for creating FFTs from and applying them to data.

## fftrees_grow_fan *Grow fast-and-frugal trees (FFTs) using the fan algorithms*

### Description

fftrees_grow_fan is called by fftrees_define to create new FFTs by applying the fan algorithms (specifically, either ifan or dfan) to data.

### Usage

```
fftrees_grow_fan(x, repeat.cues = TRUE)
```

### Arguments

| | |
|---|---|
| x | An FFTrees object. |
| repeat.cues | Can cues be considered/used repeatedly (as logical)? Default: repeat.cues = TRUE, but only relevant when using the dfan algorithm. |

### See Also

fftrees_create for creating FFTrees objects; fftrees_define for defining FFTs; fftrees_grow_fan for creating FFTs by applying algorithms to data; fftrees_wordstofftrees for creating FFTs from verbal descriptions; FFTrees for creating FFTs from and applying them to data.

## fftrees_ranktrees *Rank FFTs by current goal*

### Description

fftrees_ranktrees ranks trees in an FFTrees object x based on the current goal (either "cost" or as specified in x$params$goal).

fftrees_ranktrees is called by the main FFTrees function when creating FFTs from and applying them to (training) data.

### Usage

```
fftrees_ranktrees(x, data = "train")
```

### Arguments

| | |
|---|---|
| `x` | An `FFTrees` object. |
| `data` | The type of data to be used (as character). Default: `data = "train"`. |

### See Also

`FFTrees` for creating FFTs from and applying them to data.

---

## `fftrees_threshold_factor_grid`

*Perform a grid search over factor and return accuracy statistics for a given factor cue*

---

### Description

Perform a grid search over factor and return accuracy statistics for a given factor cue

### Usage

```
fftrees_threshold_factor_grid(
  thresholds = NULL,
  cue_v = NULL,
  criterion_v = NULL,
  directions = "=",
  sens.w = 0.5,
  cost.outcomes = list(hi = 0, fa = 1, mi = 1, cr = 0),
  cost.each = 0,
  goal.threshold = "bacc"
)
```

### Arguments

| | |
|---|---|
| `thresholds` | numeric. A vector of factor thresholds to consider. |
| `cue_v` | numeric. Feature/cue values. |
| `criterion_v` | logical. A logical vector of (TRUE) criterion values. |
| `directions` | character. Character vector of threshold directions to consider. |
| `sens.w` | numeric. Sensitivity weight parameter (from 0 to 1, for computing `wacc`). Default: `sens.w = .50`. |
| `cost.outcomes` | list. A list of length 4 with names 'hi', 'fa', 'mi', and 'cr' specifying the costs of a hit, false alarm, miss, and correct rejection, respectively. For instance, `cost.outcomes = listc("hi" = 0, "fa" = 10, "mi" = 20, "cr" = 0)` means that a false alarm and miss cost 10 and 20, respectively, while correct decisions have no cost. |
| `cost.each` | numeric. |
| `goal.threshold` | character. |

### Value

A data frame containing accuracy statistics for several factor thresholds

### See Also

fftrees_threshold_numeric_grid for numeric cues.

---

## fftrees_threshold_numeric_grid

*Perform a grid search over thresholds and return accuracy statistics for a given numeric cue*

---

### Description

Perform a grid search over thresholds and return accuracy statistics for a given numeric cue

### Usage

```
fftrees_threshold_numeric_grid(
  thresholds,
  cue_v,
  criterion_v,
  directions = c(">", "<="),
  sens.w = 0.5,
  cost.each = 0,
  cost.outcomes = list(hi = 0, fa = 1, mi = 1, cr = 0),
  goal.threshold = "bacc"
)
```

### Arguments

thresholds numeric. A vector of thresholds to consider.

cue_v numeric. Feature values.

criterion_v logical. A logical vector of (TRUE) criterion values.

directions character. Possible directions to consider.

sens.w numeric. Sensitivity weight parameter (from 0 to 1, for computing wacc). Default: sens.w = .50.

cost.each numeric. Cost to add to each value (e.g.; cost of the cue).

cost.outcomes list. A list of length 4 with names 'hi', 'fa', 'mi', and 'cr' specifying the costs of a hit, false alarm, miss, and correct rejection, respectively. For instance, cost.outcomes = listc("hi" = 0, "fa" = 10, "mi" = 20, "cr" = 0) means that a false alarm and miss cost 10 and 20, respectively, while correct decisions have no cost.

goal.threshold character. A string indicating the statistic to maximize when calculting cue thresholds: "acc" = overall accuracy, "wacc" = weighted accuracy, "bacc" = balanced accuracy.

## Value

A data frame containing accuracy statistics for several numeric thresholds.

## See Also

`fftrees_threshold_factor_grid` for factor cues.

---

`fftrees_wordstofftrees` *Convert a verbal description of an FFT into an `FFTrees` object*

---

## Description

`fftrees_wordstofftrees` converts a verbal description of an FFT (provided as a string of text) into a tree definition (of an `FFTrees` object). Thus, `fftrees_wordstofftrees` provides a simple natural language parser for FFTs.

`fftrees_wordstofftrees` is the complement function to `fftrees_ffttowords`, which converts an abstract tree definition (of an `FFTrees` object) into a verbal description (i.e., provides natural language output).

To increase robustness, the parsing of `fftrees_wordstofftrees` allows for lower- or uppercase spellings (but not typographical variants) and ignores the else-part of the final sentence (i.e., the part beginning with "otherwise").

## Usage

```
fftrees_wordstofftrees(x, my.tree)
```

## Arguments

| | |
|---|---|
| `x` | An `FFTrees` object. |
| `my.tree` | A character string. A verbal description (as a string of text) defining an FFT. |

## Value

An `FFTrees` object with a new tree definition as described by `my.tree`.

## See Also

`fftrees_ffttowords` for converting FFTs into verbal descriptions; `print.FFTrees` for printing FFTs; `plot.FFTrees` for plotting FFTs; `summary.FFTrees` for summarizing FFTs; `FFTrees` for creating FFTs from and applying them to data.

## `forestfires` *Forest fires data*

### Description

A dataset of forest fire statistics.

### Usage

```
forestfires
```

### Format

A data frame containing 517 rows and 13 columns.

**X** Integer -x-axis spatial coordinate within the Montesinho park map: 1 to 9

**Y** Integer - y-axis spatial coordinate within the Montesinho park map: 2 to 9

**month** Factor - month of the year: "jan" to "dec"

**day** Factor -day of the week: "mon" to "sun"

**FFMC** Numeric -FFMC index from the FWI system: 18.7 to 96.20

**DMC** Numeric - DMC index from the FWI system: 1.1 to 291.3

**DC** Numeric - DC index from the FWI system: 7.9 to 860.6

**ISI** Numeric - ISI index from the FWI system: 0.0 to 56.10

**temp** Numeric - temperature in Celsius degrees: 2.2 to 33.30

**RH** Numeric - relative humidity in percent: 15.0 to 100

**wind** Numeric - wind speed in km/h: 0.40 to 9.40

**rain** Numeric - outside rain in mm/m2 : 0.0 to 6.4

**area** Numeric - the burned area of the forest (in ha): 0.00 to 1090.84

...

### Source

http://archive.ics.uci.edu/ml/datasets/Forest+Fires

### See Also

Other datasets: `blood`, `breastcancer`, `car`, `contraceptive`, `creditapproval`, `fertility`, `heart.cost`, `heart.test`, `heart.train`, `heartdisease`, `iris.v`, `mushrooms`, `sonar`, `titanic`, `voting`, `wine`

## heart.cost *Cue costs for the heartdisease data*

### Description

This data further characterizes the variables (cues) in the heartdisease dataset.

### Usage

```
heart.cost
```

### Format

A data frame containing 153 rows and 14 columns.

**cue** The name of the cue

**cost** The cost of the cue

...

### Source

https://archive.ics.uci.edu/ml/machine-learning-databases/heart-disease/costs/

### See Also

heartdisease dataset.

Other datasets: blood, breastcancer, car, contraceptive, creditapproval, fertility, forestfires, heart.test, heart.train, heartdisease, iris.v, mushrooms, sonar, titanic, voting, wine

## heart.test *Heart disease testing data*

### Description

Testing data for a heartdisease data. This subset is used to test the prediction performance of a model trained on the heart.train data. The dataset heartdisease contains both datasets.

### Usage

```
heart.test
```

### Format

A data frame containing 153 rows and 14 columns (see heartdisease for details).

### Source

https://archive.ics.uci.edu/ml/datasets/Heart+Disease

### See Also

heartdisease dataset.

Other datasets: blood, breastcancer, car, contraceptive, creditapproval, fertility, forestfires, heart.cost, heart.train, heartdisease, iris.v, mushrooms, sonar, titanic, voting, wine

## heart.train *Heart disease training data*

### Description

Training data for a binary prediction model (here: FFT) on (a subset of) the heartdisease data. The complementary subset for model testing is heart.test. The data in heartdisease contains both subsets.

### Usage

```
heart.train
```

### Format

A data frame containing 150 rows and 14 columns (see heartdisease for details).

### Source

https://archive.ics.uci.edu/ml/datasets/Heart+Disease

### See Also

heartdisease dataset.

Other datasets: blood, breastcancer, car, contraceptive, creditapproval, fertility, forestfires, heart.cost, heart.test, heartdisease, iris.v, mushrooms, sonar, titanic, voting, wine

## heartdisease *Heart disease data*

### Description

A dataset predicting the diagnosis of 303 patients tested for heart disease.

### Usage

```
heartdisease
```

### Format

A data frame containing 303 rows and 14 columns, with the following variables:

**diagnosis** True value of binary criterion: TRUE = Heart disease, FALSE = No Heart disease

**age** Age (in years)

**sex** Sex, 1 = male, 0 = female

**cp** Chest pain type: ta = typical angina, aa = atypical angina, np = non-anginal pain, a = asymptomatic

**trestbps** Resting blood pressure (in mm Hg on admission to the hospital)

**chol** Serum cholestoral in mg/dl

**fbs** Fasting blood sugar > 120 mg/dl: 1 = true, 0 = false

**restecg** Resting electrocardiographic results. "normal" = normal, "abnormal" = having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV), "hypertrophy" = showing probable or definite left ventricular hypertrophy by Estes' criteria.

**thalach** Maximum heart rate achieved

**exang** Exercise induced angina: 1 = yes, 0 = no

**oldpeak** ST depression induced by exercise relative to rest

**slope** The slope of the peak exercise ST segment.

**ca** Number of major vessels (0-3) colored by flourosopy

**thal** "normal" = normal, "fd" = fixed defect, "rd" = reversible defect

...

### Source

https://archive.ics.uci.edu/ml/datasets/Heart+Disease

### See Also

heart.cost dataset for cost information.

Other datasets: blood, breastcancer, car, contraceptive, creditapproval, fertility, forestfires, heart.cost, heart.test, heart.train, iris.v, mushrooms, sonar, titanic, voting, wine

## inwords *Provide a verbal description of an FFT*

### Description

`inwords` generates and provides a verbal description of a fast-and-frugal tree (FFT) from an `FFTrees` object.

When `data` remains unspecified, `inwords` will only look up `x$trees$inwords`. When `data` is set to either "train" or "test", `inwords` first employs `fftrees_ffttowords` to re-generate the verbal descriptions of FFTs in `x`.

### Usage

```
inwords(x, data = NULL, tree = 1)
```

### Arguments

| | |
|---|---|
| `x` | An `FFTrees` object. |
| `data` | The type of data to which a tree is being applied (as character string "train" or "test"). Default: `data = NULL` will only look up `x$trees$inwords`. |
| `tree` | The tree to display (as an integer). |

### Value

A verbal description of an FFT (as a character string).

### See Also

`fftrees_ffttowords` for converting FFTs into verbal descriptions; `print.FFTrees` for printing FFTs; `plot.FFTrees` for plotting FFTs; `summary.FFTrees` for summarizing FFTs; `FFTrees` for creating FFTs from and applying them to data.

## iris.v *Iris data*

### Description

A famous dataset from R.A. Fisher (1936) simplified to predict only the virginica class (i.e., as a binary classification problem).

### Usage

```
iris.v
```

## Format

A data frame containing 150 rows and 4 columns.

## Source

`https://archive.ics.uci.edu/ml/datasets/Iris`

## References

Fisher, R.A. (1936): The use of multiple measurements in taxonomic problems. Annual Eugenics, 7, Part II, pp. 179–188.

## See Also

Other datasets: `blood`, `breastcancer`, `car`, `contraceptive`, `creditapproval`, `fertility`, `forestfires`, `heart.cost`, `heart.test`, `heart.train`, `heartdisease`, `mushrooms`, `sonar`, `titanic`, `voting`, `wine`

---

# `mushrooms` *Mushrooms data*

---

## Description

Data describing poisonous vs. non-poisonous mushrooms.

## Usage

```
mushrooms
```

## Format

A data frame containing 8,124 rows and 23 columns.

See `http://archive.ics.uci.edu/ml/machine-learning-databases/mushroom/agaricus-lepiota.names` for column descriptions.

**poisonous** logical criterion variable

**cshape** character

**csurface** character

**ccolor** character

**bruises** character

**odor** numeric

**gattach** character

**gspace** characte

**gsize** character

**gcolor** character

**sshape** character

**sroot** character

**ssaring** character

**ssbring** character

**scaring** character

**scbring** character

**vtype** character

**vcolor** character

**ringnum** character

**ringtype** character

**sporepc** character

**population** character

**habitat** character

...

## Details

This dataset includes descriptions of hypothetical samples corresponding to 23 species of gilled mushrooms in the Agaricus and Lepiota Family. Each species is classified as `poisonous` (True or False). The Guide clearly states that there is no simple rule for determining the edibility of a mushroom; no rule like “leaflets three, let it be” for Poisonous Oak and Ivy.

## Source

`https://archive.ics.uci.edu/ml/datasets/Mushroom`

## References

Mushroom records drawn from The Audubon Society Field Guide to North American Mushrooms (1981). G.H. Lincoff (Pres.), New York: A.A. Knopf.

## See Also

Other datasets: `blood`, `breastcancer`, `car`, `contraceptive`, `creditapproval`, `fertility`, `forestfires`, `heart.cost`, `heart.test`, `heart.train`, `heartdisease`, `iris.v`, `sonar`, `titanic`, `voting`, `wine`

## plot.FFTrees *Plot an FFTrees object*

### Description

plot.FFTrees visualizes an FFTrees object created by the FFTrees function.

plot.FFTrees is the main plotting function of the **FFTrees** package and called when evaluating the generic plot on an FFTrees object.

plot.FFTrees visualizes a selected FFT, key data characteristics, and various aspects of classification performance.

As x may not contain test data, plot.FFTrees by default plots the performance characteristics for training data (i.e., fitting), rather than for test data (i.e., for prediction). When test data is available, specifying data = "test" plots prediction performance.

Whenever the sensitivity weight (sens.w) is set to its default of sens.w = 0.50, a level shows *balanced* accuracy (bacc). If, however, sens.w deviates from its default, the level shows the tree's *weighted* accuracy value (wacc) and the current sens.w value (below the level).

Many aspects of the plot (e.g., its panels) and the FFT's appearance (e.g., labels of its nodes and exits) can be customized by setting corresponding arguments.

### Usage

```
## S3 method for class 'FFTrees'
plot(
  x = NULL,
  data = "train",
  what = "all",
  tree = 1,
  main = NULL,
  cue.labels = NULL,
  decision.labels = NULL,
  cue.cex = NULL,
  threshold.cex = NULL,
  decision.cex = 1,
  comp = TRUE,
  show.header = NULL,
  show.tree = NULL,
  show.confusion = NULL,
  show.levels = NULL,
  show.roc = NULL,
  show.icons = NULL,
  show.iconguide = NULL,
  hlines = TRUE,
  label.tree = NULL,
  label.performance = NULL,
  n.per.icon = NULL,
  level.type = "bar",
  which.tree = NULL,
  decision.names = NULL,
  stats = NULL,
  ...
)
```

## Arguments

`x` An FFTrees object created by the FFTrees function.

`data` The type of data in x to be plotted (as a string) or a test dataset (as a data frame).

- A valid data string must be either 'train' (for fitting performance) or 'test' (for prediction performance).
- For a valid data frame, the specified tree is evaluated and plotted for this data (as 'test' data), but the global FFTrees object x remains unchanged unless it is re-assigned.

By default, data = 'train' (as x may not contain test data).

`what` What should be plotted (as a string)? Valid options are:

**'all'** Plot the tree diagram with all corresponding guides and performance statistics, but excluding cue accuracies.

**'cues'** Plot only the marginal accuracy of cues in ROC space. Note that cue accuracies are *not* shown when calling what = 'all' and use the showcues function.

**'icontree'** Plot tree diagram with icon arrays on exit nodes. Consider also setting n.per.icon and show.iconguide.

**'tree'** Plot only the tree diagram.

**'roc'** Plot only the performance of tree(s) (and comparison algorithms) in ROC space.

Default: what = 'all'.

`tree` The tree to be plotted (as an integer, only valid when the corresponding tree argument is non-empty). Default: tree = 1. To plot the best training or best test tree with respect to the goal specified during FFT construction, use 'best.train' or 'best.test', respectively.

`main` The main plot label (as a character string).

`cue.labels` An optional string of labels for the cues / nodes (as character vector).

`decision.labels`
A character vector of length 2 indicating the content-specific names for noise and signal predictions/exits.

`cue.cex` The size of the cue labels (as numeric).

`threshold.cex` The size of the threshold labels (as numeric).

`decision.cex` The size of the decision labels (as numeric).

`comp` Should the performance of competitive algorithms (e.g.; logistic regression, random forests, etc.) be shown in the ROC plot (if available, as logical)?

`show.header` Show header with basic data properties (in top panel, as logical)?

`show.tree` Show nodes and exits of FFT (in middle panel, as logical)?

`show.confusion` Show 2x2 confusion matrix (in bottom panel, as logical)?

`show.levels` Show performance levels (in bottom panel, as logical)?

`show.roc` Show ROC curve (in bottom panel, as logical)?

`show.icons` Show exit cases as icon arrays (in middle panel, as logical)?

`show.iconguide` Show icon guide (in middle panel, as logical)?

`hlines` Show horizontal panel separation lines (as logical)? Default: `hlines = TRUE`.

`label.tree` Label for the FFT (optional, as character string).

`label.performance`
Labels for the performance section (optional, as character string).

`n.per.icon` Number of cases per icon (as numeric).

`level.type` How should bottom levels be drawn (as a string)? Can be `"bar"` (the default) or `"line"`.

`which.tree` Deprecated argument. Use `tree` instead.

`decision.names` Deprecated argument. Use `decision.labels` instead.

`stats` Deprecated argument. Should statistical information be plotted (as logical)? Use `what = "all"` to include performance statistics and `what = "tree"` to plot only a tree diagram.

`...` Graphical parameters (passed to text of panel titles, to `showcues` when `what = 'cues'`, or to `title` when `what = 'roc'`).

## Value

An invisible `FFTrees` object `x` and a plot visualizing and describing an FFT (as side effect).

## See Also

`showcues` for plotting cue accuracies; `print.FFTrees` for printing FFTs; `summary.FFTrees` for summarizing FFTs; `FFTrees` for creating FFTs from and applying them to data.

Other plot functions: `showcues()`

## Examples

```
# Create FFTs (for heartdisease data):
heart_fft <- FFTrees(formula = diagnosis ~ .,
                     data = heart.train)

# Visualize the default FFT (Tree #1, what = 'all'):
plot(heart_fft, main = "Heart disease",
     decision.labels = c("Absent", "Present"))

# Visualize cue accuracies (in ROC space):
plot(heart_fft, what = "cues",  main = "Cue accuracies for heart disease data")

# Visualize tree diagram with icon arrays on exit nodes:
plot(heart_fft, what = "icontree", n.per.icon = 2,
     main = "Diagnosing heart disease")

# Visualize performance comparison in ROC space:
plot(heart_fft, what = "roc", main = "Performance comparison for heart disease data")

# Visualize predictions of FFT #2 (for new test data) with custom options:
plot(heart_fft, tree = 2, data = heart.test,
     main = "Predicting heart disease",
     cue.labels = c("1. thal?", "2. cp?", "3. ca?", "4. exang"),
     decision.labels = c("ok", "sick"), n.per.icon = 2,
     show.header = TRUE, show.confusion = FALSE, show.levels = FALSE, show.roc = FALSE,
     hlines = FALSE, font = 3, col = "steelblue")

# # For details, see
# vignette("FFTrees_plot", package = "FFTrees")
```

## predict.FFTrees — *Predict classification outcomes or probabilities from data*

### Description

predict.FFTrees predicts binary classification outcomes or their probabilities from newdata for an FFTrees object.

### Usage

```
## S3 method for class 'FFTrees'
predict(
  object = NULL,
  newdata = NULL,
  tree = 1,
  type = "class",
  sens.w = NULL,
  method = "laplace",
  data = NULL,
  ...
)
```

### Arguments

| | |
|---|---|
| object | An FFTrees object created by the FFTrees function. |
| newdata | dataframe. A data frame of test data. |
| tree | integer. Which tree in the object should be used? By default, tree = 1 is used. |
| type | string. What should be predicted? Can be "class", which returns a vector of class predictions, "prob" which returns a matrix of class probabilities, or "both" which returns a matrix with both class and probability predictions. |

`sens.w, data` deprecated

`method` string. Method of calculating class probabilities. Either 'laplace', which applies the Laplace correction, or 'raw' which applies no correction.

`...` Additional arguments passed on to `predict`.

## Value

Either a logical vector of predictions, or a matrix of class probabilities.

## See Also

`print.FFTrees` for printing FFTs; `plot.FFTrees` for plotting FFTs; `summary.FFTrees` for summarizing FFTs; `FFTrees` for creating FFTs from and applying them to data.

## Examples

```
# Create training and test data:
set.seed(100)
breastcancer <- breastcancer[sample(nrow(breastcancer)), ]
breast.train <- breastcancer[1:150, ]
breast.test  <- breastcancer[151:303, ]

# Create an FFTrees object from the training data:
breast.fft <- FFTrees(
  formula = diagnosis ~ .,
  data = breast.train
)

# Predict classes for test data:
breast.fft.pred <- predict(breast.fft,
  newdata = breast.test
)

# Predict class probabilities for test data:
breast.fft.pred <- predict(breast.fft,
  newdata = breast.test,
  type = "prob"
)
```

---

`print.FFTrees` *Print basic information of fast-and-frugal trees (FFTs)*

---

## Description

`print.FFTrees` prints basic information on FFTs for an `FFTrees` object `x`.

As `x` may not contain test data, `print.FFTrees` by default prints the performance characteristics for training data (i.e., fitting), rather than for test data (i.e., for prediction). When test data is available, specify `data = "test"` to print prediction performance.

### Usage

```
## S3 method for class 'FFTrees'
print(x = NULL, tree = 1, data = "train", ...)
```

### Arguments

x
: An FFTrees object created by FFTrees.

tree
: The tree to be printed (as an integer, only valid when the corresponding tree argument is non-empty). Default: tree = 1. To print the best training or best test tree with respect to the goal specified during FFT construction, use "best.train" or "best.test", respectively.

data
: The type of data in x to be printed (as a string) or a test dataset (as a data frame).

  - A valid data string must be either 'train' (for fitting performance) or 'test' (for prediction performance).
  - For a valid data frame, the specified tree is evaluated and printed for this data (as 'test' data), but the global FFTrees object x remains unchanged unless it is re-assigned.

  By default, data = 'train' (as x may not contain test data).

...
: additional arguments passed to print.

### Value

An invisible FFTrees object x and summary information on an FFT printed to the console (as side effect).

### See Also

plot.FFTrees for plotting FFTs; summary.FFTrees for summarizing FFTs; inwords for obtaining a verbal description of FFTs; FFTrees for creating FFTs from and applying them to data.

## select_best_tree *Select the best tree (from the current set)*

### Description

select_best_tree selects (looks up and identifies) the best tree from the set (or “fan”) of FFTs contained in the current FFTrees object x, an existing type of data ('train' or 'test'), and a goal for which corresponding statistics are available in the designated data type (in x$trees$stats).

### Usage

```
select_best_tree(x, data, goal)
```

## Arguments

| | |
|---|---|
| x | An FFTrees object. |
| data | The type of data to consider (as character: either 'train' or 'test'). |
| goal | character. A goal to maximize or minimize when selecting a tree from an existing x (for which values exist in x$trees$stats). |

## Details

Importantly, select_best_tree only identifies and selects from the set of *existing* trees with known statistics, rather than creating new trees or computing new cue thresholds. More specifically, goal is used for identifying and selecting the best of an existing set of FFTs, but not for computing new cue thresholds (see goal.threshold and fftrees_cuerank()) or creating new trees (see goal.chase and fftrees_ranktrees()).

## Value

An integer denoting the tree that maximizes/minimizes goal in data.

## See Also

FFTrees for creating FFTs from and applying them to data.

---

# showcues *Visualize cue accuracies (as points in ROC space)*

---

## Description

showcues plots the cue accuracies of an FFTrees object created by the FFTrees function (as points in ROC space).

If the optional arguments cue.accuracies and alt.goal are specified, their values take precedence over the corresponding settings of an FFTrees object x (but do not change x).

showcues is called when the main plot.FFTrees function is set to what = "cues".

## Usage

```
showcues(
  x = NULL,
  cue.accuracies = NULL,
  alt.goal = NULL,
  main = NULL,
  top = 5,
  quiet = FALSE,
  ...
)
```

## Arguments

| | |
|---|---|
| x | An FFTrees object created by the FFTrees function. |
| cue.accuracies | An optional data frame specifying cue accuracies directly (without specifying FFTrees object x). |
| alt.goal | An optional alternative goal to sort the current cue accuracies (without using the goal of FFTrees object x). |
| main | A main plot title (as character string). |
| top | How many of the top cues should be highlighted (as an integer)? |
| quiet | Should user feedback messages be printed (as logical)? Default: quiet = FALSE (i.e., show messages). |
| ... | Graphical parameters (passed to plot). |

## Value

A plot showing cue accuracies (of an FFTrees object) (as points in ROC space).

## See Also

print.FFTrees for printing FFTs; plot.FFTrees for plotting FFTs; summary.FFTrees for summarizing FFTs; FFTrees for creating FFTs from and applying them to data.

Other plot functions: plot.FFTrees()

## Examples

```
# Create fast-and-frugal trees (FFTs) for heart disease:
heart.fft <- FFTrees(formula = diagnosis ~ .,
                     data = heart.train,
                     data.test = heart.test,
                     main = "Heart Disease",
                     decision.labels = c("Healthy", "Diseased")
                     )

# Show cue accuracies (in ROC space):
showcues(heart.fft,
         main = "Predicting heart disease")
```

# sonar *Sonar data*

## Description

Sonar data

## Usage

```
sonar
```

## Format

A data frame containing 208 rows and 60 columns.

## Source

https://archive.ics.uci.edu/ml/datasets/Connectionist+Bench+(Sonar,+Mines+vs.+Rocks)

## See Also

Other datasets: blood, breastcancer, car, contraceptive, creditapproval, fertility, forestfires, heart.cost, heart.test, heart.train, heartdisease, iris.v, mushrooms, titanic, voting, wine

---

# summary.FFTrees *Summarize an* FFTrees *object*

---

## Description

summary.FFTrees summarizes key contents of an FFTrees object.

## Usage

```
## S3 method for class 'FFTrees'
summary(object, tree = NULL, ...)
```

## Arguments

| | |
|---|---|
| object | An FFTrees object. |
| tree | The tree to summarize (as an integer, but may be a vector). If tree = NULL (as per default) or exceeding the possible range 1:object$trees$n, information on all trees in object is returned. |
| ... | Additional arguments (currently ignored). |

## Details

summary.FFTrees returns an invisible list containing two elements:

1. definitions and corresponding performance measures of trees;
2. stats on decision frequencies, derived probabilities, and costs (separated by train and test).

A header prints descriptive information of the FFTrees object (to the console): Its main title, number of trees (object$trees$n), and the name of the criterion variable (object$criterion_name).

Per default, information on all available trees is shown and returned. Specifying tree filters the output list elements for the corresponding tree(s). When only a single tree is specified, the printed header includes a verbal description of the corresponding tree.

While summary.FFTrees provides key details about the specified tree(s), the individual decisions (stored in object$trees$decisions) are not shown or returned.

## Value

An invisible list with elements containing the `definitions` and performance `stats` of the FFT(s) specified by `tree`(s).

## See Also

`print.FFTrees` for printing FFTs; `plot.FFTrees` for plotting FFTs; `inwords` for obtaining a verbal description of FFTs; `FFTrees` for creating FFTs from and applying them to data.

---

# `titanic` *Titanic survival data*

---

## Description

Data indicating who survived on the Titanic.

## Usage

```
titanic
```

## Format

A data frame containing 2,201 rows and 4 columns.

**class** Factor - Class (first, second, third, or crew)

**age** Factor - Age group (child or adult)

**sex** Factor - Sex (male or female)

**survived** Factor - Whether the passenger survived (1) or not (0)

...

## Details

See `Titanic` of the R **datasets** package for details and the same data (in a 4-dimensional `table`).

## Source

`https://www.encyclopedia-titanica.org`

## References

Dawson, Robert J. MacG. (1995), The ‘Unusual Episode’ Data Revisited. Journal of Statistics Education, 3. doi: 10.1080/10691898.1995.11910499.

## See Also

Other datasets: `blood`, `breastcancer`, `car`, `contraceptive`, `creditapproval`, `fertility`, `forestfires`, `heart.cost`, `heart.test`, `heart.train`, `heartdisease`, `iris.v`, `mushrooms`, `sonar`, `voting`, `wine` `voting` *Voting data*

### Description

A dataset of votes for each of the U.S. House of Representatives Congressmen on the 16 key votes identified by the CQA.

### Usage

```
voting
```

### Format

A data frame containing 435 rows and 16 columns.

### Details

The CQA lists nine different types of votes: voted for, paired for, and announced for (these three simplified to yea), voted against, paired against, and announced against (these three simplified to nay), voted present, voted present to avoid conflict of interest, and did not vote or otherwise make a position known (these three simplified to an unknown disposition).

The binary criterion variable used here is `party.crit`.

### Source

https://archive.ics.uci.edu/ml/datasets/Congressional+Voting+Records

### References

Congressional Quarterly Almanac, 98th Congress, 2nd session 1984, Volume XL: Congressional Quarterly Inc. Washington, D.C., 1985.

### See Also

Other datasets: `blood`, `breastcancer`, `car`, `contraceptive`, `creditapproval`, `fertility`, `forestfires`, `heart.cost`, `heart.test`, `heart.train`, `heartdisease`, `iris.v`, `mushrooms`, `sonar`, `titanic`, `wine`

`wine` *Wine tasting data*

### Description

Chemical and tasting data from wines in North Portugal.

### Usage

```
wine
```

### Format

A data frame containing 6497 rows and 13 columns.

### Source

`http://archive.ics.uci.edu/ml/datasets/Wine+Quality`

### See Also

Other datasets: `blood`, `breastcancer`, `car`, `contraceptive`, `creditapproval`, `fertility`, `forestfires`, `heart.cost`, `heart.test`, `heart.train`, `heartdisease`, `iris.v`, `mushrooms`, `sonar`, `titanic`, `voting`

# Index