Package ‘prcbench’

March 26, 2020

Type Package
Title Testing Workbench for Precision-Recall Curves
Version 0.8.2
Date 2020-03-25
Description A testing workbench for evaluating precision-recall curves under various conditions.
URL http://takayasaito.github.io/prcbench/
      https://github.com/takayasaito/prcbench
BugReports https://github.com/takayasaito/prcbench/issues
Depends R (>= 3.2.3)
License GPL-3
Language en-US
LazyData TRUE
Imports ROCR (>= 1.0-7), PRROC (>= 1.1), precrec (>= 0.1), rJava (>= 0.9-7), R6 (>= 2.1.1), assertthat (>= 0.1), grid, gridExtra (>= 2.0.0), graphics, ggplot2 (>= 2.1.0), methods, memoise (>= 1.0.0)
RoxygenNote 7.1.0
Encoding UTF-8
Suggests microbenchmark (>= 1.4-2.1), PerfMeas (>= 1.2.1), testthat (>= 0.11.0), knitr (>= 1.11), rmarkdown (>= 0.8.1)
VignetteBuilder knitr
NeedsCompilation no
Author Takaya Saito [aut, cre],
      Marc Rehmsmeier [aut]
Maintainer Takaya Saito <takaya.saito@outlook.com>
Repository CRAN
Date/Publication 2020-03-26 06:00:02 UTC
## autoplot.evalcurve

Plot the result of Precision-Recall curve evaluation

### Description

The `plot_eval_results` function validates Precision-Recall curves and creates a plot.

### Usage

```r
## S3 method for class 'evalcurve'
autoplot(
  object,
  base_plot = TRUE,
  ret_grob = FALSE,
  ncol = NULL,
  nrow = NULL,
  use_category = FALSE,
  ...)
```
Arguments

- **object**: An S3 object that contains evaluation results of Precision-Recall curves.
- **base_plot**: A Boolean value to specify whether the base points are plotted.
- **ret_grob**: A Boolean value to specify whether the function returns a grob object.
- **ncol**: An integer used for the column size of multiple panes.
- **nrow**: An integer used for the row size of multiple panes.
- **use_category**: A Boolean value to specify whether the categorical summary instead of the total summary.
- **...**: Not used by this function.

Value

A data frame with validation results.

Examples

```r
library(ggplot2)

## Plot evaluation results on test datasets r1, r2, and r3
testset <- create_testset("curve", c("c1", "c2", "c3"))
toolset <- create_toolset(set_names = "crv5")
eres1 <- run_evalcurve(testset, toolset)
autoplot(erres1)
```

C1DATA

**C1: Pre-calculated Precision-Recall curve**

Description

A list contains scores, labels, and pre-calculated recall and precision values as x and y.

Usage

data(C1DATA)

Format

A list with 5 items.

- **scores**: input scores
- **labels**: input labels
- **bp_x**: pre-calculated recall values for curve evaluation
- **bp_y**: pre-calculated precision values for curve evaluation
- **tp_x**: x position for displaying the test result in a plot
- **tp_y**: y position for displaying the test result in a plot
A list contains scores, labels, and pre-calculated recall and precision values as x and y.

Usage

data(C2DATA)

Format

See C1DATA.

A list contains scores, labels, and pre-calculated recall and precision values as x and y.

Usage

data(C3DATA)

Format

See C1DATA.

A list contains scores, labels, and pre-calculated recall and precision values as x and y.

Usage

data(C4DATA)

Format

See C1DATA.
create_example_func

Create an example for the func argument of the create_usrtool function

Description

The create_example_func function creates an example for the create_usrtool function.

Usage

create_example_func()

Value

A function as an example for create_usrtool

See Also

create_usrtool requires the same format. create_testset for testset.

Examples

## Create a function
func <- create_example_func()
func

create_testset

Create a list of test datasets

Description

The create_testset function creates test datasets either for benchmarking or curve evaluation.

Usage

create_testset(test_type, set_names = NULL)

Arguments

test_type A single string to specify the type of dataset generated by this function.
  "bench" Create test datasets for benchmarking
  "curve" Create test datasets for curve evaluation
set_names A character vector to specify the names of test datasets.
1. For benchmarking (test_type = "bench")
   This function uses a naming convention for randomly generated data for benchmarking. The format is a prefix ('i' or 'b') followed by the number of dataset. The prefix 'i' indicates a balanced dataset, whereas 'b' indicates an imbalanced dataset. The number can be used with a suffix 'k' or 'm', indicating respectively 1000 or 1 million.
   Below are some examples.
   "b100" A balanced data set with 50 positives and 50 negatives.
   "b10k" A balanced data set with 5000 positives and 5000 negatives.
   "b1m" A balanced data set with 500,000 positives and 500,000 negatives.
   "i100" An imbalanced data set with 25 positives and 75 negatives.
   The function returns a list of TestDataB objects.

2. For curve evaluation (test_type = "curve")
   The following three predefined datasets can be specified for curve evaluation.

<table>
<thead>
<tr>
<th>set name</th>
<th>S3 object</th>
<th>data source</th>
</tr>
</thead>
<tbody>
<tr>
<td>c1 or C1</td>
<td>TestDataC</td>
<td>C1DATA</td>
</tr>
<tr>
<td>c2 or C2</td>
<td>TestDataC</td>
<td>C2DATA</td>
</tr>
<tr>
<td>c3 or C3</td>
<td>TestDataC</td>
<td>C3DATA</td>
</tr>
<tr>
<td>c4 or C4</td>
<td>TestDataC</td>
<td>C4DATA</td>
</tr>
</tbody>
</table>

   The function returns a list of TestDataC objects.

Value
A list of R6 test dataset objects.

See Also
run_benchmark and run_evalcurve require the list of the datasets generated by this function. TestDataB for benchmarking test data. TestDataC, C1DATA, C2DATA, C3DATA, and C4DATA for curve evaluation test data. create_usrdata for creating a user-defined test set.

Examples
```r
## Create a balanced data set with 50 positives and 50 negatives
tset1 <- create_testset("bench", "b100")
tset1

## Create an imbalanced data set with 25 positives and 75 negatives
tset2 <- create_testset("bench", "i100")
tset2

## Create P1 dataset
tset3 <- create_testset("curve", "c1")
tset3
```
```r
## Create P1 dataset
```
create_toolset

```r
tset4 <- create_testset("curve", c("c1", "c2"))
tset4
```

---

**create_toolset**

Create a set of tools

**Description**

The create_toolset function takes names of predefined tools and generates a list of wrapper functions for Precision-Recall curve calculations.

**Usage**

```r
create_toolset(
  tool_names = NULL,
  set_names = NULL,
  calc_auc = TRUE,
  store_res = TRUE
)
```

**Arguments**

- `tool_names` A character vector to specify the names of performance evaluation tools. The names for the following five tools can be currently used.
  - ROCR
  - AUCCalculator
  - PerfMeas
  - PRROC
  - precrec
- `set_names` A character vector to specify a predefined set name. Following six sets are currently available.
  - "def5" A set of 5 tools with `calc_auc = TRUE` and `store_res = TRUE`
  - "auc5" A set of 5 tools with `calc_auc = TRUE` and `store_res = FALSE`
  - "crv5" A set of 5 tools with `calc_auc = FALSE` and `store_res = TRUE`
  - "def4" A set of 4 tools with `calc_auc = TRUE` and `store_res = TRUE`
  - "auc4" A set of 4 tools with `calc_auc = TRUE` and `store_res = FALSE`
  - "crv4" A set of 4 tools with `calc_auc = FALSE` and `store_res = TRUE`
- `calc_auc` A Boolean value to specify whether the AUC score should be calculated.
- `store_res` A Boolean value to specify whether the calculated curve is retrieved and stored

**Value**

A list of R6 tool objects.
create_usrdataset

See Also

run_benchmark and run_evalcurve require the list of the tools generated by this function ToolROCR, ToolAUCCalculator, ToolPerfMeas, ToolPRROC, and Toolprecrec as R6 tool classes.

Examples

```r
## Create ROCR and precrec
toolset1 <- create_toolset(c("ROCR", "precrec"))
toolset1

## Create auc5 tools
toolset2 <- create_toolset(set_names = "auc5")
toolset2
```

create_usrdataset Create a user-defined test dataset

Description

The create_usrdataset function creates various types of test datasets.

Usage

```r
create_usrdataset(
  test_type,
  scores = NULL,
  labels = NULL,
  tsname = NULL,
  base_x = NULL,
  base_y = NULL,
  text_x = NULL,
  text_y = NULL,
  text_x2 = text_x,
  text_y2 = text_y
)
```

Arguments

- **test_type**: A single string to specify the type of dataset generated by this function.
  - "bench" Create a test dataset for benchmarking
  - "curve" Create a test dataset for curve evaluation
- **scores**: A numeric vector to set scores.
- **labels**: A numeric vector to set labels.
- **tsname**: A single string to specify the name of the dataset.
- **base_x**: A numeric vector to set pre-calculated recall values for curve evaluation.
base_y  A numeric vector to set pre-calculated precision values for curve evaluation.

text_x  A single numeric value to set the x position for displaying the test result in a plot.

text_y  A single numeric value to set the y position for displaying the test result in a plot.

text_x2 A single numeric value to set the x position for displaying the test result (group into categories) in a plot.

text_y2 A single numeric value to set the y position for displaying the test result (group into categories) in a plot.

Value
A list of R6 test dataset objects.

See Also
create_testset for creating a predefined test set. TestDataB for benchmarking test data. TestDataC for curve evaluation test data.

Examples

## Create a test dataset for benchmarking
testset2 <- create_usrdata("bench", scores = c(0.1, 0.2), labels = c(1, 0),
                        tsname = "m1")
testset2

## Create a test dataset for curve evaluation
testset <- create_usrdata("curve", scores = c(0.1, 0.2), labels = c(1, 0),
                         base_x = c(0, 1.0), base_y = c(0, 0.5))
testset

create_usrtool

Create a set of tools

Description
The create_toolset function takes names of predefined tools and generates a list of wrapper functions for Precision-Recall curve calculations.

Usage

create_usrtool(
  tool_name,
  func,
  calc_auc = TRUE,
  store_res = TRUE,
  x = NA,
  y = NA
)
Arguments

- **tool_name**: A single string to specify the name of a user-defined tool.
- **func**: A function to calculate a Precision-Recall curve and the AUC. It should take an element of the test dataset generated by `create_testset` as an argument. It also should return a list with three elements - 'x', 'y', and 'auc' that represent calculated recall and precision values plus the AUC score. See `create_example_func` for an example.
- **calc_auc**: A Boolean value to specify whether the AUC score should be calculated.
- **store_res**: A Boolean value to specify whether the calculated curve is retrieved and stored.
- **x**: Set pre-calculated recall values.
- **y**: Set pre-calculated precision values.

Value

A list of R6 tool objects.

See Also

- `create_toolset` to create a predefined tool set.
- `create_testset` for testset.
- `create_example_func` to create an example function.

Examples

```r
## Create a new tool interface called "xyz"
efunc <- create_example_func()
toolset1 <- create_usrtool("xyz", efunc)
toolset1

## Example function with a correct argument
testset <- create_usrdata("bench", scores = c(0.1, 0.2), labels = c(1, 0))
retf <- efunc(testset[[1]])
retf
```

---

**Description**

The prcbench package provides four categories of important functions: tool interface, test data interface, benchmarking, and curve evaluation.
run_benchmark

Tool interface

The `create_toolset` function creates a common interface for five different tools that calculate Precision-Recall curves. These tools are ROCR, AUCCalculator, PerfMeas, PRROC, and precrec. The `create_usrtool` function helps users to make the same interface of the predefined ones for their own tools.

Test data interface

The `create_testset` function creates two different types of test data sets. The first type is for benchmarking, and the second type is for curve evaluation. The `create_usrdata` function helps users to make their own test data sets.

Benchmarking

The `run_benchmark` function takes a tool set and a test data set and run microbenchmark for them.

Curve evaluation

The `run_evalcurve` function takes a tool set and a test data set and evaluates the accuracy of Precision-Recall curves for them.

---

**run_benchmark** Run microbenchmark with specified tools and test sets

---

Description

The `run_benchmark` function runs `microbenchmark` for specified tools and test datasets

Usage

```
run_benchmark(testset, toolset, times = 5, unit = "ms", use_sys_time = FALSE)
```

Arguments

- **testset** A character vector to specify a test set generated by `create_testset`
- **toolset** A character vector to specify a tool set generated by `create_toolset`
- **times** The number of iteration used in `microbenchmark`
- **unit** A single string to specify the unit used in `summary.microbenchmark`
- **use_sys_time** A Boolean value to specify `system.time` is used instead of `summary.microbenchmark`

Value

A data frame of microbenchmark results with additional columns.
run_evalcurve

Evaluate Precision-Recall curves with specified tools and test sets

Description

The `run_evalcurve` function runs several tests to evaluate the accuracy of Precision-Recall curves.

Usage

```r
run_evalcurve(testset, toolset, auto_combo = TRUE)
```

Arguments

- **testset**: A character vector to specify a test set generated by `create_testset`.
- **toolset**: A character vector to specify a tool set generated by `create_toolset`.
- **auto_combo**: A Boolean value to specify whether a combination of test and tool sets is automatically created.

Value

A data frame with validation results.

See Also

- `create_testset` to generate a test dataset.
- `create_toolset` to generate a tool set.

Examples

```r
## Evaluate curves for c1, c2, c3 test sets and crv5 tool set
testset <- create_testset("curve", c("c1", "c2", "c3"))
toolset <- create_toolset(set_names = "crv5")
res1 <- run_evalcurve(testset, toolset)
res1
```
Description

TestDataB is a class that contains scores and label for performance evaluation tools. It provides necessary methods for benchmarking.

Format

An R6 class object.

Methods

- `get_tsname()`: Get the dataset name.
- `get_scores()`: Get a vector of scores.
- `get_labels()`: Get a vector of labels.
- `get_fg()`: Get a vector of positive scores.
- `get_bg()`: Get a vector of negative scores.
- `get_fname()`: Get a file name that contains scores and labels.
- `del_file()`: Delete the file with scores and labels.

Public methods:

- `TestDataB$new()`
- `TestDataB$get_tsname()
- `TestDataB$get_scores()
- `TestDataB$get_labels()
- `TestDataB$get_fg()
- `TestDataB$get_bg()
- `TestDataB$get_fname()
- `TestDataB$del_file()
- `TestDataB$print()
- `TestDataB$clone()

Method `new()`:

Usage:

`TestDataB$new(scores = NULL, labels = NULL, tsname = NA)`

Method `get_tsname()`:

Usage:

`TestDataB$get_tsname()`
Method get_scores():
  Usage:
 TestDataB$get_scores()

Method get_labels():
  Usage:
 TestDataB$get_labels()

Method get_fg():
  Usage:
 TestDataB$get_fg()

Method get_bg():
  Usage:
 TestDataB$get_bg()

Method get_fname():
  Usage:
 TestDataB$get_fname()

Method del_file():
  Usage:
 TestDataB$del_file()

Method print():
  Usage:
 TestDataB$print(...)

Method clone(): The objects of this class are cloneable with this method.
  Usage:
  TestDataB$clone(deep = FALSE)
  Arguments:
  deep Whether to make a deep clone.

See Also

create_testset for creating a list of test datasets. TestDataC is derived from this class for curve evaluation.

Examples

## Initialize with scores, labels, and a dataset name
testset <- TestDataB$new(c(0.1, 0.2, 0.3), c(0, 1, 1), "m1")
testset
Description

`TestDataC` is a class that contains scores and label for performance evaluation tools. It provides necessary methods for curve evaluation.

Format

An R6 class object.

Methods

- `set_basepoints_x(x)`: Set pre-calculated recall values for curve evaluation
- `set_basepoints_y(y)`: Set pre-calculated precision values for curve evaluation
- `get_basepoints_x()`: Get pre-calculated recall values for curve evaluation
- `get_basepoints_y()`: Get pre-calculated precision values for curve evaluation
- `set_textpos_x(x)`: Set the x position for displaying the test result in a plot
- `set_textpos_y(y)`: Set the y position for displaying the test result in a plot
- `get_textpos_x()`: Get the x position for displaying the test result in a plot
- `get_textpos_y()`: Get the y position for displaying the test result in a plot

Following seven methods are inherited from `TestDataB`. See `TestDataB` for the method descriptions.

- `get_datname()`
- `get_scores()`
- `get_labels()`
- `get_fg()`
- `get_bg()`
- `get_fname()`
- `del_file()`

Super class

`prcbench::TestDataB` -> `TestDataC`
Methods

Public methods:

- `TestDataC$set_basepoints_x()`
- `TestDataC$set_basepoints_y()`
- `TestDataC$get_basepoints_x()`
- `TestDataC$get_basepoints_y()`
- `TestDataC$set_textpos_x()`
- `TestDataC$set_textpos_y()`
- `TestDataC$set_textpos_x2()`
- `TestDataC$set_textpos_y2()`
- `TestDataC$get_textpos_x()`
- `TestDataC$get_textpos_y()`
- `TestDataC$get_textpos_x2()`
- `TestDataC$get_textpos_y2()`
- `TestDataC$clone()`

**Method** `set_basepoints_x()`:

*Usage:*

`TestDataC$set_basepoints_x(x)`

**Method** `set_basepoints_y()`:

*Usage:*

`TestDataC$set_basepoints_y(y)`

**Method** `get_basepoints_x()`:

*Usage:*

`TestDataC$get_basepoints_x()`

**Method** `get_basepoints_y()`:

*Usage:*

`TestDataC$get_basepoints_y()`

**Method** `set_textpos_x()`:

*Usage:*

`TestDataC$set_textpos_x(x)`

**Method** `set_textpos_y()`:

*Usage:*

`TestDataC$set_textpos_y(y)`

**Method** `set_textpos_x2()`:

*Usage:*

`TestDataC$set_textpos_x2(x)`
Method set_textpos_y2:
Usage:
TestDataC$set_textpos_y2(y)

Method get_textpos_x:
Usage:
TestDataC$get_textpos_x()

Method get_textpos_y:
Usage:
TestDataC$get_textpos_y()

Method get_textpos_x2:
Usage:
TestDataC$get_textpos_x2()

Method get_textpos_y2:
Usage:
TestDataC$get_textpos_y2()

Method clone:
The objects of this class are cloneable with this method.
Usage:
TestDataC$clone(deep = FALSE)
Arguments:
depth Whether to make a deep clone.

See Also
create_testset for creating a list of test datasets. It is derived from TestDataB.

Examples
## Initialize with scores, labels, and a dataset name
testset <- TestDataC$new(c(0.1, 0.2), c(1, 0), "c4")
testset

## Set base points
x <- c(0.13, 0.2)
y <- c(0.5, 0.6)
testset$set_basepoints_x(x)
testset$set_basepoints_y(y)
testset
**ToolAUCCalculator**

**R6 class of the AUCCalculator tool**

**Description**

ToolAUCCalculator is a wrapper class for the AUCCalculator tool, which is a Java library that provides calculations of ROC and Precision-Recall curves.

**Format**

An R6 class object.

**Inheritance**

ToolIFBase

**Methods**

- `set_jarpath(jarpath)` It sets an AUCCalculator jar file.
  
  **jarpath** File path of the AUCCalculator jar file, e.g. "/path1/path2/auc2.jar".

  Following nine methods are inherited from ToolIFBase. See ToolIFBase for the method descriptions.
  
  - call((testset,calc_auc,store_res)
  - get_toolname()
  - set_toolname(toolname)
  - get_setname()
  - set_setname(setname)
  - get_result()
  - get_x()
  - get_y()
  - get_auc()

**Super class**

prcbench::ToolIFBase -> ToolAUCCalculator

**Methods**

**Public methods:**

- ToolAUCCalculator$new()
- ToolAUCCalculator$set_jarpath()
- ToolAUCCalculator$set_curvetype()
- ToolAUCCalculator$set_auctype()
Method `clone()`

Usage:
```
ToolAUCCalculator$clone(deep = FALSE)
```

Arguments:
- `deep` Whether to make a deep clone.

See Also

This class is derived from `ToolIFBase`. `create_toolset` for creating a list of tools.

Examples

```r
## Initialization
toolauccalc <- ToolAUCCalculator$new()

## Show object info
toolauccalc

## create_toolset should be used for benchmarking and curve evaluation
toolauccalc2 <- create_toolset("AUCCalculator")
```
Description

ToolIFBase is an abstract class to provide a uniform interface for performance evaluation tools.

Format

An R6 class object

Methods

• call(testset, calc_auc, store_res): It calls an actual tool to calculate Precision-Recall curves.
  
  testset R6 object generated by the create_testset function.
  calc_auc A Boolean value to specify whether the AUC score should be calculated.
  store_res A Boolean value to specify whether the calculated curve is retrieved and stored.

• get_toolname(): Get the name of the tool.

• set_toolname(toolname): Set the name of the tool.

• get_setname(): Get the name of the tool set.

• set_setname(setname): Set the name of the tool set.

• get_result(): Get a list with curve values and the AUC score.

• get_x(): Get calculated recall values.

• get_y(): Get calculated precision values.

• get_auc(): Get the AUC score.

Methods

Public methods:

• ToolIFBase$new()
• ToolIFBase$call()
• ToolIFBase$get_toolname()
• ToolIFBase$get_toolname()
• ToolIFBase$set_toolname()
• ToolIFBase$set_setname()
• ToolIFBase$get_result()
• ToolIFBase$get_x()
• ToolIFBase$get_y()
• ToolIFBase$get_auc()
• ToolIFBase$print()
• `ToolIFBase$clone()`

**Method new():**

*Usage:*

`ToolIFBase$new(...)`

**Method call():**

*Usage:*

`ToolIFBase$call(testset, calc_auc, store_res)`

**Method get_toolname():**

*Usage:*

`ToolIFBase$get_toolname()`

**Method set_toolname():**

*Usage:*

`ToolIFBase$set_toolname(toolname)`

**Method get_setname():**

*Usage:*

`ToolIFBase$get_setname()`

**Method set_setname():**

*Usage:*

`ToolIFBase$set_setname(setname)`

**Method get_result():**

*Usage:*

`ToolIFBase$get_result()`

**Method get_x():**

*Usage:*

`ToolIFBase$get_x()`

**Method get_y():**

*Usage:*

`ToolIFBase$get_y()`

**Method get_auc():**

*Usage:*

`ToolIFBase$get_auc()`

**Method print():**

*Usage:*

`ToolIFBase$print(...)`

**Method clone():** The objects of this class are cloneable with this method.

*Usage:*

`ToolIFBase$clone(deep = FALSE)`

*Arguments:*

depth Whether to make a deep clone.
See Also

ToolROCR, ToolAUCCalculator, ToolPerfMeas, ToolPRROC, and Toolprecrec are derived from this class. create_toolset for creating a list of tools.

---

ToolPerfMeas  \hspace{1cm} R6 class of the PerfMeas tool

Description

ToolPerfMeas is a wrapper class for the PerfMeas tool, which is an R library that provides several performance measures.

Format

An R6 class object.

Inheritance

ToolIFBase

Methods

Following nine methods are inherited from ToolIFBase. See ToolIFBase for the method descriptions.

- call(testset,calc_auc,store_res)
- get_toolname()
- set_toolname(toolname)
- get_setname()
- set_setname(setname)
- get_result()
- get_x()
- get_y()
- get_auc()

Super class

prcbench::ToolIFBase -> ToolPerfMeas
Methods

Public methods:

• ToolPerfMeas$clone()

Method clone(): The objects of this class are cloneable with this method.

Usage:

ToolPerfMeas$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

See Also

This class is derived from ToolIFBase. create_toolset for creating a list of tools.

Examples

```r
## Initialization
toolperf <- ToolPerfMeas$new()

## Show object info
toolperf

## create_toolset should be used for benchmarking and curve evaluation
toolperf2 <- create_toolset("PerfMeas")
```

Description

Toolprecrec is a wrapper class for the precrec tool, which is an R library that provides calculations of ROC and Precision-Recall curves.

Format

An R6 class object.

Inheritance

ToolIFBase
Methods

Following nine methods are inherited from ToolIFBase. See ToolIFBase for the method descriptions.

- call(testset, calc_auc, store_res)
- get_toolname()
- set_toolname(toolname)
- get_setname()
- set_setname(setname)
- get_result()
- get_x()
- get_y()
- get_auc()

Super class

prcbench::ToolIFBase -> Toolprecrec

Methods

Public methods:

- Toolprecrec$new()
- Toolprecrec$set_x_bins()
- Toolprecrec$clone()

Method new():

Usage:
Toolprecrec$new(...)

Method set_x_bins():

Usage:
Toolprecrec$set_x_bins(x_bins)

Method clone(): The objects of this class are cloneable with this method.

Usage:
Toolprecrec$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

See Also

This class is derived from ToolIFBase. create_toolset for creating a list of tools.
Examples

```r
## Initialization
toolprecrec <- Toolprecrec$new()

## Show object info
toolprecrec

## create_toolset should be used for benchmarking and curve evaluation
toolprecrec2 <- create_toolset("precrec")
```

---

### ToolPRROC

*R6 class of the PRROC tool*

**Description**

ToolPRROC is a wrapper class for the PRROC tool, which is an R library that provides calculations of ROC and Precision-Recall curves.

**Format**

An R6 class object.

**Inheritance**

ToolIFBase

**Methods**

- `set_curve(val)` A Boolean value to specify whether a Precision-Recall curve is calculated.
- `set_minStepSize(val)` A numeric value to specify the minimum step size between two intermediate points.

Following nine methods are inherited from ToolIFBase. See ToolIFBase for the method descriptions.

- `call(testset,calc_auc,store_res)`
- `get_toolname()`
- `set_toolname(toolname)`
- `get_setname()`
- `set_setname(setname)`
- `get_result()`
- `get_x()`
- `get_y()`
- `get_auc()`
Super class

`prcbench::ToolIFBase` -> `ToolPRROC`

Methods

Public methods:

- `ToolPRROC$new()`
- `ToolPRROC$set_curve()`
- `ToolPRROC$set_minStepSize()`
- `ToolPRROC$set_aucType()`
- `ToolPRROC$clone()`

Method `new()`:

Usage:

```r
ToolPRROC$new(...)
```

Method `set_curve()`:

Usage:

```r
ToolPRROC$set_curve(val)
```

Method `set_minStepSize()`:

Usage:

```r
ToolPRROC$set_minStepSize(val)
```

Method `set_aucType()`:

Usage:

```r
ToolPRROC$set_aucType(val)
```

Method `clone()`: The objects of this class are cloneable with this method.

Usage:

```r
ToolPRROC$clone(deep = FALSE)
```

Arguments:

- `deep` Whether to make a deep clone.

See Also

This class is derived from `ToolIFBase`. `create_toolset` for creating a list of tools.

Examples

```r
## Initialization
toolprroc <- ToolPRROC$new()

## Show object info
toolprroc

## create_toolset should be used for benchmarking and curve evaluation
toolprroc2 <- create_toolset("PRROC")
```
ToolROCR  

R6 class of the ROCR tool

Description

ToolROCR is a wrapper class for the ROCR tool, which is an R library that provides calculations of various performance evaluation measures.

Format

An R6 class object.

Inheritance

ToolIFBase

Methods

Following nine methods are inherited from ToolIFBase. See ToolIFBase for the method descriptions.

- call(testset, calc_auc, store_res)
- get_toolname()
- set_toolname(toolname)
- get_setname()
- set_setname(setname)
- get_result()
- get_x()
- get_y()
- get_auc()

Super class

prcbench::ToolIFBase -> ToolROCR

Methods

Public methods:

- ToolROCR$clone()

Method clone(): The objects of this class are cloneable with this method.

Usage:
ToolROCR$clone(deep = FALSE)

Arguments:

deep  Whether to make a deep clone.
See Also

This class is derived from ToolIFBase. `create_toolset` for creating a list of tools.

Examples

```r
## Initialization
toolrocr <- ToolROCR$new()

## Show object info
toolrocr

## create_toolset should be used for benchmarking and curve evaluation
toolrocr2 <- create_toolset("ROCR")
```
Index

*Topic datasets
C1DATA, 3
C2DATA, 4
C3DATA, 4
C4DATA, 4

autoplot.evalcurve, 2
C1DATA, 3, 4, 6
C2DATA, 4, 6
C3DATA, 4, 6
C4DATA, 4, 6
create_example_func, 5, 10
create_testset, 5, 9–12, 14, 17
create_toolset, 7, 10–12, 19, 22–24, 26, 28
create_usrdata, 6, 8, 11
create_usrtool, 5, 9, 11

microbenchmark, 11, 12
prcbench, 10
prcbench::TestDataB, 15
prcbench::ToolIFBase, 18, 22, 24, 26, 27

run_benchmark, 6, 8, 11, 11
run_evalcurve, 6, 8, 11, 12

summary.microbenchmark, 11
system.time, 11

TestDataB, 6, 9, 13, 15, 17
TestDataC, 6, 9, 14, 15
ToolAUCCalculator, 8, 18, 22
ToolIFBase, 18, 19, 20, 22–28
ToolPerfMeas, 8, 22, 22
Toolprecrec, 8, 22, 23
ToolPRROC, 8, 22, 25
ToolROCR, 8, 22, 27