Package ‘mlr3benchmark’

October 4, 2021

Title  Analysis and Visualisation of Benchmark Experiments

Version  0.1.3

Description  Implements methods for post-hoc analysis and
visualisation of benchmark experiments, for ‘mlr3’ and beyond.

License  LGPL-3

URL  https://mlr3benchmark.mlr-org.com,
https://github.com/mlr-org/mlr3benchmark

BugReports  https://github.com/mlr-org/mlr3benchmark/issues

Depends  R (>= 3.1.0)

Imports  checkmate, data.table, ggplot2, mlr3misc, R6

Suggests  mlr3, mlr3learners, PMCMRplus, rpart, testthat, xgboost

Encoding  UTF-8

NeedsCompilation  no

RoxygenNote  7.1.1

Author  Sonabend Raphael [cre, aut] (<https://orcid.org/0000-0001-9225-4654>),
Florian Pfisterer [aut] (<https://orcid.org/0000-0001-8867-762X>),
Michel Lang [ctb] (<https://orcid.org/0000-0001-9754-0393>),
Bernd Bischl [ctb] (<https://orcid.org/0000-0001-6002-6980>)

Maintainer  Sonabend Raphael <raphaelsonabend@gmail.com>

Repository  CRAN

Date/Publication  2021-10-04 15:30:02 UTC

R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Documented Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>mlr3benchmark-package</td>
<td>2</td>
</tr>
<tr>
<td>as.BenchmarkAggr</td>
<td>2</td>
</tr>
<tr>
<td>autoplot.BenchmarkAggr</td>
<td>3</td>
</tr>
<tr>
<td>BenchmarkAggr</td>
<td>6</td>
</tr>
<tr>
<td>requireNamespaces</td>
<td>9</td>
</tr>
</tbody>
</table>

Index  10
Description

Implements methods for post-hoc analysis and visualisation of benchmark experiments, for `mlr3` and beyond.

Author(s)

Maintainer: Sonabend Raphael <raphaelsonabend@gmail.com> (ORCID)
Authors:

• Florian Pfisterer <pfistererf@googlemail.com> (ORCID)

Other contributors:

• Michel Lang <michellang@gmail.com> (ORCID) [contributor]
• Bernd Bischl <bernd_bischl@gmx.net> (ORCID) [contributor]

See Also

Useful links:

• https://mlr3benchmark.mlr-org.com
• https://github.com/mlr-org/mlr3benchmark
• Report bugs at https://github.com/mlr-org/mlr3benchmark/issues

as.BenchmarkAggr

Coercions to BenchmarkAggr

Description

Coercion methods to BenchmarkAggr. For mlr3::BenchmarkResult this is a simple wrapper around the BenchmarkAggr constructor called with mlr3::BenchmarkResult$aggregate().

Usage

```r
as.BenchmarkAggr(
  obj,
  task_id = "task_id",
  learner_id = "learner_id",
  independent = TRUE,
  strip_prefix = TRUE,
  ...
)
```
Arguments

obj (mlr3::BenchmarkResult|matrix(1))
Passed to BenchmarkAggr$new().

task_id, learner_id, independent, strip_prefix
See BenchmarkAggr$initialize().

... ANY
Passed to mlr3::BenchmarkResult$aggregate().

Examples

df = data.frame(tasks = factor(rep(c("A", "B"), each = 5),
levels = c("A", "B")),
learners = factor(paste0("L", 1:5)),
RMSE = runif(10), MAE = runif(10))
as.BenchmarkAggr(df, task_id = "tasks", learner_id = "learners")

if (requireNamespaces(c("mlr3", "rpart"))) {
library(mlr3)
task = tsks(c("boston_housing", "mtcars"))
learns = lrns(c("regr.featureless", "regr.rpart"))
bm = benchmark(benchmark_grid(task, learns, rsmp("cv", folds = 2)))

# default measure
as.BenchmarkAggr(bm)

# change measure
as.BenchmarkAggr(bm, measures = msr("regr.rmse"))
}

Description

Generates plots for BenchmarkAggr, all assume that there are multiple, independent, tasks. Choices depending on the argument type:

- "mean" (default): Assumes there are at least two independent tasks. Plots the sample mean of the measure for all learners with error bars computed with the standard error of the mean.
- "box": Boxplots for each learner calculated over all tasks for a given measure.
- "fn": Plots post-hoc Friedman-Nemenyi by first calling BenchmarkAggr$friedman_posthoc and plotting significant pairs in coloured squares and leaving non-significant pairs blank, useful for simply visualising pair-wise comparisons.
• "cd": Critical difference plots (Demsar, 2006). Learners are drawn on the x-axis according to their average rank with the best performing on the left and decreasing performance going right. Any learners not connected by a horizontal bar are significantly different in performance. Critical differences are calculated as:

\[ CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}} \]

Where \( q_\alpha \) is based on the studentized range statistic. See references for further details. It’s recommended to crop white space using external tools, or function image_trim() from package magick.

Usage

```r
## S3 method for class 'BenchmarkAggr'
autoplot(obj, 
  type = c("mean", "box", "fn", "cd"), 
  meas = NULL, 
  level = 0.95, 
  p.value = 0.05, 
  minimize = TRUE, 
  test = "nem", 
  baseline = NULL, 
  style = 1L, 
  ratio = 1/7, 
  col = "red", 
  ...
)
```

Arguments

- **obj** `BenchmarkAggr`
- **type** (character(1)) Type of plot, see description.
- **meas** (character(1)) Measure to plot, should be in `obj$measures`, can be NULL if only one measure is in `obj`.
- **level** (numeric(1)) Confidence level for error bars for type = "mean"
- **p.value** (numeric(1)) What value should be considered significant for type = "cd" and type = "fn".
- **minimize** (logical(1)) For type = "cd", indicates if the measure is optimally minimized. Default is TRUE.
- **test** (character(1))) For type = "cd", critical differences are either computed between all learners
(test = "nemenyi"), or to a baseline (test = "bd"). Bonferroni-Dunn usually yields higher power than Nemenyi as it only compares algorithms to one baseline. Default is "nemenyi".

baseline
(character(1))
For type = "cd" and test = "bd" a baseline learner to compare the other learners to, should be in $learners, if NULL then differences are compared to the best performing learner.

style
(integer(1))
For type = "cd" two ggplot styles are shipped with the package (style = 1 or style = 2), otherwise the data can be accessed via the returned ggplot.

ratio
(numeric(1))
For type = "cd" and style = 1, passed to ggplot2::coord_fixed(), useful for quickly specifying the aspect ratio of the plot, best used with ggsave().

col
(character(1))
For type = "fn", specifies color to fill significant tiles, default is "red".

... ANY
Additional arguments, currently unused.

References


Examples

if (requireNamespaces(c("mlr3learners", "mlr3", "rpart", "xgboost"))) {
library(mlr3)
library(mlr3learners)
library(ggplot2)
set.seed(1)
task = tsks(c("iris", "sonar", "wine", "zoo"))
learns = lrns(c("classif.featureless", "classif.rpart", "classif.xgboost"))
bm = benchmark(benchmark_grid(task, learns, rsmp("cv", folds = 3)))
obj = as.BenchmarkAggr(bm)

# mean and error bars
autoplot(obj, type = "mean", level = 0.95)

if (requireNamespace("PMCMRplus", quietly = TRUE)) {
  # critical differences
  autoplot(obj, type = "cd", style = 1)
  autoplot(obj, type = "cd", style = 2)

  # post-hoc friedman-nemenyi
  autoplot(obj, type = "fn")
}
}
BenchmarkAggr  

Aggregated Benchmark Result Object

Description

An R6 class for aggregated benchmark results.

Details

This class is used to easily carry out and guide analysis of models after aggregating the results after resampling. This can either be constructed using \texttt{mlr3} objects, for example the result of \texttt{mlr3::BenchmarkResult$aggregate} or via \texttt{as.BenchmarkAggr}, or by passing in a custom dataset of results. Custom datasets must include at the very least, a character column for learner ids, a character column for task ids, and numeric columns for one or more measures.

Currently supported for multiple independent datasets only.

Active bindings

\begin{itemize}
\item \texttt{data (data.table::data.table)}
  Aggregated data.
\item \texttt{learners (character())}
  Unique learner names.
\item \texttt{tasks (character())}
  Unique task names.
\item \texttt{measures (character())}
  Unique measure names.
\item \texttt{nlrns (integer())}
  Number of learners.
\item \texttt{ntasks (integer())}
  Number of tasks.
\item \texttt{nmeas (integer())}
  Number of measures.
\item \texttt{nrow (integer())}
  Number of rows.
\item \texttt{col_roles (character())}
  Column roles, currently cannot be changed after construction.
\end{itemize}

Methods

Public methods:

\begin{itemize}
\item \texttt{BenchmarkAggr$new()}
\item \texttt{BenchmarkAggr$print()}
\item \texttt{BenchmarkAggr$summary()}
\end{itemize}
Method `new()`: Creates a new instance of this R6 class.

Usage:
```
BenchmarkAggr$new(
  dt,
  task_id = "task_id",
  learner_id = "learner_id",
  independent = TRUE,
  strip_prefix = TRUE,
  ...
)
```

Arguments:
```
.dt (matrix(1))
  'matrix like object coercable to data.table::data.table, should include column names "task_id" and "learner_id", and at least one measure (numeric). If ids are not already factors then coerced internally.

.task_id (character(1))
  String specifying name of task id column.

.learner_id (character(1))
  String specifying name of learner id column.

.independent (logical(1))
  Are tasks independent of one another? Affects which tests can be used for analysis.

.strip_prefix (logical(1))
  If TRUE (default) then mlr prefixes, e.g. regr., classif., are automatically stripped from the learner_id.

... ANY
  Additional arguments, currently unused.
```

Method `print()`: Prints the internal data via `data.table::print.data.table`.

Usage:
```
BenchmarkAggr$print(...)```

Arguments:
```
... ANY
  Passed to data.table::print.data.table.
```

Method `summary()`: Prints the internal data via `data.table::print.data.table`.

Usage:
```
BenchmarkAggr$summary(...)```

Arguments:
Method rank_data(): Ranks the aggregated data given some measure.

Usage:
BenchmarkAggr$rank_data(meas = NULL, minimize = TRUE, task = NULL, ...)

Arguments:
meas (character(1))
  Measure to rank the data against, should be in $measures. Can be NULL if only one measure in data.
minimize (logical(1))
  Should the measure be minimized? Default is TRUE.
task (character(1))
  If NULL then returns a matrix of ranks where columns are tasks and rows are learners, otherwise returns a one-column matrix of a specified task, should be in $tasks.

Method friedman_test(): Computes Friedman test over all tasks, assumes datasets are independent.

Usage:
BenchmarkAggr$friedman_test(meas = NULL, p.adjust.method = NULL)

Arguments:
meas (character(1))
  Measure to rank the data against, should be in $measures. If no measure is provided then returns a matrix of tests for all measures.
p.adjust.method (character(1))
  Passed to p.adjust if meas = NULL for multiple testing correction. If NULL then no correction applied.

Method friedman_posthoc(): Posthoc Friedman Nemenyi tests. Computed with PMCMRplus::frdAllPairsNemenyiTest. If global $friedman_test is non-significant then this is returned and no post-hocs computed. Also returns critical difference

Usage:
BenchmarkAggr$friedman_posthoc(meas = NULL, p.value = 0.05)

Arguments:
meas (character(1))
  Measure to rank the data against, should be in $measures. Can be NULL if only one measure in data.
p.value (numeric(1))
  p.value for which the global test will be considered significant.

Method subset(): Subsets the data by given tasks or learners. Returns data as data.table::data.table.

Usage:
BenchmarkAggr$subset(task = NULL, learner = NULL)
Arguments:
- task (character())
  Task(s) to subset the data by.
- learner (character())
  Learner(s) to subset the data by.

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

Usage:
BenchmarkAggr$clone(deep = FALSE)

Arguments:
- deep Whether to make a deep clone.

References
't r format_bib("demsar_2006")

Examples
# Not restricted to mlr3 objects
df = data.frame(tasks = factor(rep(c("A", "B"), each = 5),
  levels = c("A", "B")),
  learners = factor(paste0("L", 1:5)),
  RMSE = runif(10), MAE = runif(10))
as.BenchmarkAggr(df, task_id = "tasks", learner_id = "learners")

if (requireNamespaces(c("mlr3", "rpart"))) {
  library(mlr3)
  task = tsks(c("boston_housing", "mtcars"))
  learns = lrns(c("regr.featureless", "regr.rpart"))
  bm = benchmark(benchmark_grid(task, learns, rsmp("cv", folds = 2)))

  # coercion
  as.BenchmarkAggr(bm)
}

Description
Internal helper function for documentation.

Usage
requireNamespaces(x)

Arguments
- x Packages to check.
Index

as.BenchmarkAggr, 2, 6
autoplot.BenchmarkAggr, 3

BenchmarkAggr, 2–4, 6

data.table::data.table, 6–8
data.table::frank(), 8
data.table::print.data.table, 7, 8

ggplot2::coord_fixed(), 5
ggsave(), 5

mlr3::BenchmarkResult, 2, 3, 6
mlr3benchmark (mlr3benchmark-package), 2
mlr3benchmark-package, 2

p.adjust, 8
PMCMRplus::frdAllPairsNemenyiTest, 8

R6, 7
requireNamespaces, 9