Package ‘mlr3viz’

August 15, 2022

Title  Visualizations for 'mlr3'

Version  0.5.10

Description  Provides visualizations for 'mlr3' objects such as tasks, predictions, resample results or benchmark results via the autoplot() generic of 'ggplot2'. The returned 'ggplot' objects are intended to provide sensible defaults, yet can easily be customized to create camera-ready figures. Visualizations include barplots, boxplots, histograms, ROC curves, and Precision-Recall curves.

License  LGPL-3


BugReports  https://github.com/mlr-org/mlr3viz/issues

Depends  R (>= 3.1.0)

Imports  checkmate, data.table, ggplot2 (>= 3.3.0), mlr3misc (>= 0.7.0), scales, utils, viridis

Suggests  bbotk, cluster, factoextra, GGally, ggfortify (>= 0.4.11), ggpdata, glmnet, knitr, lgr, mlr3 (>= 0.6.0), mlr3cluster, mlr3filters, mlr3learners, mlr3tuning (>= 0.9.0), paradox, partykit, patchwork (>= 1.1.1), precrec, ranger, rpart, stats, testthat (>= 3.0.0), vdiffr (>= 1.0.2), xgboost

Config/testthat/edition  3

Config/testthat/parallel  true

Encoding  UTF-8

NeedsCompilation  no

RoxygenNote  7.2.1

Collate  'BenchmarkResult.R' 'Filter.R' 'LearnerClassifCVGlmnet.R'
  'LearnerClassifGlmnet.R' 'LearnerClassifRpart.R'
  'LearnerClustHierarchical.R' 'LearnerRegrCVGlmnet.R'
  'LearnerRegrGlmnet.R' 'LearnerRegrRpart.R'
  'OptimInstanceSingleCrit.R' 'Prediction.R'
  'PredictionClassif.R' 'PredictionClust.R' 'PredictionRegr.R'
  'ResampleResult.R' 'Task.R' 'TaskClassif.R' 'TaskClust.R'

1
mlr3viz-package

mlr3viz: Visualizations for 'mlr3'

Description

Provides visualizations for 'mlr3' objects such as tasks, predictions, resample results or benchmark results via the autoplot() generic of 'ggplot2'. The returned 'ggplot' objects are intended to provide sensible defaults, yet can easily be customized to create camera-ready figures. Visualizations include barplots, boxplots, histograms, ROC curves, and Precision-Recall curves.
Author(s)

Maintainer: Michel Lang <michellang@gmail.com> (ORCID)

Authors:

• Patrick Schratz <patrick.schratz@gmail.com> (ORCID)
• Raphael Sonabend <raphael.sonabend.15@ucl.ac.uk> (ORCID)
• Marc Becker <marcbecker@posteo.de> (ORCID)
• Jakob Richter <jakob1richter@gmail.com> (ORCID)

Other contributors:

• Damir Pulatov <dpulatov@uwyo.edu> [contributor]

See Also

Useful links:

• https://mlr3viz.mlr-org.com
• https://github.com/mlr-org/mlr3viz
• Report bugs at https://github.com/mlr-org/mlr3viz/issues

as_precrec

Convert to 'precrec' Format

Description

Converts to a format which is understood by \texttt{precrec::evalmod()} of package \texttt{precrec}.

Usage

as_precrec(object)

## S3 method for class 'PredictionClassif'
as_precrec(object)

## S3 method for class 'ResampleResult'
as_precrec(object)

## S3 method for class 'BenchmarkResult'
as_precrec(object)

Arguments

object (any)
Object to convert.
autoplot.BenchmarkResult

Value

Object as created by `precrec::mmdata()`.

References


---

**autoplot.BenchmarkResult**

*Plot for BenchmarkResult*

---

**Description**

Generates plots for `mlr3::BenchmarkResult`, depending on argument `type`:

- "boxplot" (default): Boxplots of performance measures, one box per `mlr3::Learner` and one facet per `mlr3::Task`.
- "roc": ROC curve (1 - specificity on x, sensitivity on y). The `mlr3::BenchmarkResult` may only have a single `mlr3::Task` and a single `mlr3::Resampling`. Note that you can subset any `mlr3::BenchmarkResult` with its `$filter()` method (see examples). Requires package `precrec`. Additional arguments will be passed down to the respective `autoplot()` function in package `precrec`. Arguments `calc_avg` and `cb_alpha` are passed to `precrec::evalmod()`.
- "prc": Precision recall curve. See "roc".

**Usage**

```r
## S3 method for class 'BenchmarkResult'
autoplot(object, type = "boxplot", measure = NULL, ...)
```

**Arguments**

- `object`: (`mlr3::BenchmarkResult`).
- `type`: (character(1)): Type of the plot. See description.
- `measure`: (`mlr3::Measure`) Performance measure to use.
- `...`: (any): Additional arguments, passed down to the respective `geom` or plotting function.

**Value**

`ggplot2::ggplot()` object.
Theme

The `theme_mlr3()` and viridis color maps are applied by default to all `autoplot()` methods. To change this behavior set `options(mlr3.theme = FALSE)`.

References


Examples

```r
if (requireNamespace("mlr3")) {
  library(mlr3)
  library(mlr3viz)

  tasks = tsks(c("pima", "sonar"))
  learner = lrns(c("classif.featureless", "classif.rpart"),
                 predict_type = "prob")
  resampling = rsmps("cv")
  object = benchmark(benchmark_grid(tasks, learner, resampling))

  head(fortify(object))
  autoplot(object)
  autoplot(object$clone(deep = TRUE)$filter(task_ids = "pima", type = "roc")
}
```

autoplott.Filter  
*Plot for Filter Scores*

Description

Generates plots for `mlr3filters::Filter`, depending on argument `type`:
- "barplot" (default): Bar plot of filter scores.

Usage

```r
## S3 method for class 'Filter'
autoplott(object, type = "boxplot", n = Inf, ...)
```

Arguments

- `object` *(mlr3filters::Filter)*.
- `type` *(character(1))*: Type of the plot. See description.
- `n` *(integer(1))*: Only include the first `n` features with highest importance. Defaults to all features.
- `...` *(any)*: Additional argument, passed down to the respective `geom`. 
Value

`ggplot2::ggplot()` object.

Theme

The `theme_mlr3()` and viridis color maps are applied by default to all `autoplot()` methods. To change this behavior set `options(mlr3.theme = FALSE)`.

Examples

```r
if (requireNamespace("mlr3")) {
  library(mlr3)
  library(mlr3viz)
  library(mlr3filters)

  task = tsk("mtcars")
  f = flt("correlation")
  f$calculate(task)

  head(fortify(f))
  autoplot(f, n = 5)
}
```

Description

Visualizations for `mlr3learners::mlr_learners_classif.glmnet`, `mlr3learners::mlr_learners_regr.glmnet`, `mlr3learners::mlr_learners_classif.cv_glmnet` and `mlr3learners::mlr_learners_regr.cv_glmnet` using the package `ggfortify`.

Note that learner-specific plots are experimental and subject to change.

Usage

```r
# S3 method for class 'LearnerClassifCVglmnet'
autoplot(object, ...)

# S3 method for class 'LearnerClassifGlmnet'
autoplot(object, ...)

# S3 method for class 'LearnerRegrCVglmnet'
autoplot(object, ...)

# S3 method for class 'LearnerRegrGlmnet'
autoplot(object, ...)
```
Arguments

object (mlr3learners::LearnerClassifGlmnet | mlr3learners::LearnerRegrGlmnet | mlr3learners::LearnerRegrCVGlmnet | mlr3learners::LearnerRegrCVGlmnet).

... (any): Additional arguments, passed down to ggparty::autoplot.party().

Value

ggplot2::ggplot() object.

Theme

The theme_mlr3() and viridis color maps are applied by default to all autoplot() methods. To change this behavior set options(mlr3.theme = FALSE).

References


Examples

## Not run:
library(mlr3)
library(mlr3viz)
library(mlr3learners)

# classification
task = tsk("sonar")
learner = lrn("classif.glmnet")
learner$train(task)
autoplot(learner)

# regression
task = tsk("mtcars")
learner = lrn("regr.glmnet")
learner$train(task)
autoplot(learner)

## End(Not run)
Description

Visualize trees for `mlr3::mlr_learners_classif.rpart` and `mlr3::mlr_learners_regr.rpart` using the package `ggparty`.

Contrary to `ggparty`, boxplots are shown in the terminal nodes for regression trees.
Note that learner-specific plots are experimental and subject to change.

Usage

```r
## S3 method for class 'LearnerClassifRpart'
autoplot(object, ...)

## S3 method for class 'LearnerRegrRpart'
autoplot(object, ...)
```

Arguments

- **object** (`mlr3::LearnerClassifRpart` | `mlr3::LearnerRegrRpart`).
- **...** (`any`): Additional arguments, passed down to `ggparty::autoplot.party()`.

Value

`ggplot2::ggplot()` object.

Theme

The `theme_mlr3()` and viridis color maps are applied by default to all `autoplot()` methods. To change this behavior set `options(mlr3.theme = FALSE)`.

Examples

```r
if (requireNamespace("mlr3")) {
  library(mlr3)
  library(mlr3viz)

  # classification
  task = tsk("iris")
  learner = lrn("classif.rpart", keep_model = TRUE)
  learner$train(task)
  autoplot(learner)

  # regression
  task = tsk("mtcars")
  learner = lrn("regr.rpart", keep_model = TRUE)
  learner$train(task)
  autoplot(learner)
}
```
autoplot.LearnerClustHierarchical

Plot for Hierarchical Clustering Learners

Description
Generates plots for hierarchical clusterers, depending on argument type:

- "dend" (default): dendrograms using factoextra package.
- "scree": scree plot that shows the number of possible clusters on x-axis and the height on the y-axis.

Note that learner-specific plots are experimental and subject to change.

Usage
## S3 method for class 'LearnerClustHierarchical'
autoplot(object, type = "dend", ...)

Arguments
- object (mlr3cluster::LearnerClustAgnes | mlr3cluster::LearnerClustDiana | mlr3cluster::LearnerClustHclust).
- type (character(1)): Type of the plot. See description.
- ... (any): Additional arguments, passed down to function factoextra::fviz_dend() in package factoextra.

Value
ggplot2::ggplot() object.

Theme
The theme_mlr3() and viridis color maps are applied by default to all autoplot() methods. To change this behavior set options(mlr3.theme = FALSE).

Examples
if (requireNamespace("mlr3")) {
  library(mlr3)
  library(mlr3cluster)
  library(mlr3viz)

  task = mlr_tasks$get("usarrests")

  # agnes clustering
  learner = mlr_learners$get("clust.agnes")
  learner$train(task)
autoplot(OptimInstanceSingleCrit)

Plot for OptimInstanceSingleCrit

Description
Generates plots for bbotk::OptimInstanceSingleCrit.

Usage
## S3 method for class 'OptimInstanceSingleCrit'
autoplot(
  object,
  type = "marginal",
  cols_x = NULL,
  trafo = FALSE,
  learner = mlr3::lrn("regr.ranger"),
  grid_resolution = 100,
  batch = NULL,
  ...
)

Arguments

  object  (bbotk::OptimInstanceSingleCrit)

  type     (character(1)): Type of the plot. Available choices:
               •  "marginal": scatter plots of x versus y The colour of the points shows the batch number.
               •  "performance": scatter plots of batch number versus y
               •  "parameter": scatter plots of batch number versus input. The colour of the points shows the y values.
"parallel" parallel coordinates plot. x values are rescaled by \((x - \text{mean}(x)) / \text{sd}(x)\).
- "points" - scatter plot of two x dimensions versus y. The colour of the points shows the y values.
- "surface": surface plot of two x dimensions versus y values. The y values are interpolated with the supplied \texttt{mlr3::Learner}.
- "pairs": plots all x and y values against each other.

- \texttt{cols_x} (character())
  Column names of x values. By default, all untransformed x values from the search space are plotted. Transformed hyperparameters are prefixed with \texttt{x\_domain\_}.
- \texttt{trafo} (logical(1))
  Determines if untransformed (FALSE) or transformed (TRUE) x values are plotted.
- \texttt{learner} (\texttt{mlr3::Learner})
  Regression learner used to interpolate the data of the surface plot.
- \texttt{grid_resolution} (numeric())
  Resolution of the surface plot.
- \texttt{batch} (integer())
  The batch number(s) to limit the plot to. Default is all batches.
- \ldots{} (any): Additional arguments, possibly passed down to the underlying plot functions.

\textbf{Value}

\texttt{ggplot2::ggplot()} object.

\textbf{Theme}

The \texttt{theme\_mlr3()} and viridis color maps are applied by default to all \texttt{autoplot()} methods. To change this behavior set \texttt{options(mlr3.theme = FALSE)}.

\textbf{Examples}

```r
if (requireNamespace("mlr3") && requireNamespace("bbotk") && requireNamespace("patchwork")) {
  library(bbotk)
  library(paradox)

  fun = function(xs) {
    c(y = -(xs[[1]] - 2)^2 - (xs[[2]] + 3)^2 + 10)
  }
  domain = ps(
    x1 = p_dbl(-10, 10),
    x2 = p_dbl(-5, 5)
  )
  codomain = ps(
    y = p_dbl(tags = "maximize")
  )
  obfun = ObjectiveRFun$new(
```
fun = fun,
    domain = domain,
    codomain = codomain
)

instance = OptimInstanceSingleCrit$new(objective = obfun, terminator = trm("evals", n_evals = 20))

optimizer = opt("random_search", batch_size = 2)
optimizer$optimize(instance)

# plot y versus batch number
autoplot(instance, type = "performance")

# plot x1 values versus performance
autoplot(instance, type = "marginal", cols_x = "x1")

# plot parallel coordinates plot
autoplot(instance, type = "parallel")

# plot pairs
autoplot(instance, type = "pairs")

autoplot.PredictionClassif

Plot for PredictionClassif

Description

Generates plots for mlr3::PredictionClassif, depending on argument type:

- "stacked" (default): Stacked barplot of true and estimated class labels.
- "roc": ROC curve (1 - specificity on x, sensitivity on y). Requires package prec roc.
- "prc": Precision recall curve. Requires package prec roc.
- "threshold": Systematically varies the threshold of the mlr3::PredictionClassif object and plots the resulting performance as returned by measure. Arguments in ... are passed down to the score function of the mlr3::Measure.

Usage

## S3 method for class 'PredictionClassif'
autoplot(object, type = "stacked", measure = NULL, ...)

Arguments

- **object** (mlr3::PredictionClassif).
- **type** (character(1)):
  Type of the plot. See description.
autoclip.PredictionClust

Description

Generates plots for mlr3cluster::PredictionClust, depending on argument type:

- "scatter" (default): scatterplot with correlation values and colored cluster assignments.
- "sil": Silhouette plot with mean silhouette value as a reference line. Requires package ggfortify.
- "pca": Perform PCA on data and color code cluster assignments. Inspired by and uses ggfortify::autoplot.kmeans.
Usage

```r
## S3 method for class 'PredictionClust'
autoplot(object, task, row_ids = NULL, type = "scatter", ...)```

Arguments

- `object` (mlr3cluster::PredictionClust).
- `task` (mlr3cluster::TaskClust).
- `row_ids` row ids to subset task data to ensure that only the data used to make predictions are shown in plots.
- `type` (character(1)): Type of the plot. See description.
- `...` (any): Additional arguments, passed down to the respective geom.

Value

`ggplot2::ggplot()` object.

Theme

The `theme_mlr3()` and viridis color maps are applied by default to all `autoplot()` methods. To change this behavior set `options(mlr3.theme = FALSE)`.

References


Examples

```r
if (requireNamespace("mlr3")) {
  library(mlr3)
  library(mlr3cluster)
  library(mlr3viz)
  task = tsk("usarrests")
  learner = lrn("clust.kmeans", centers = 3)
  object = learner$train(task)$predict(task)
  head(fortify(object))
  autoplot(object, task)
}
```
Description

Generates plots for `mlr3::PredictionRegr`, depending on argument type:

- "xy" (default): Scatterplot of "true" response vs. "predicted" response. By default a linear model is fitted via `geom_smooth(method = "lm")` to visualize the trend between x and y (by default colored blue).
  - In addition `geom_abline()` with slope = 1 is added to the plot.
  - Note that `geom_smooth()` and `geom_abline()` may overlap, depending on the given data.
- "histogram": Histogram of residuals: \( r = y - \hat{y} \).
- "residual": Plot of the residuals, with the response \( \hat{y} \) on the "x" and the residuals on the "y" axis.
  - By default a linear model is fitted via `geom_smooth(method = "lm")` to visualize the trend between x and y (by default colored blue).

Usage

```r
## S3 method for class 'PredictionRegr'
autoplot(object, type = "xy", ...)
```

Arguments

- `object` (mlr3::PredictionRegr).
- `type` (character(1)): Type of the plot. See description.
- `...` (any): Additional arguments, passed down to the respective `geom`.

Value

`ggplot2::ggplot` object.

Theme

The `theme_mlr3()` and `viridis` color maps are applied by default to all `autoplot()` methods. To change this behavior set `options(mlr3.theme = FALSE)`.
Examples

```r
if (requireNamespace("mlr3")) {
  library(mlr3)
  library(mlr3viz)

  task = tsk("boston_housing")
  learner = lrn("regr.rpart")
  object = learner$train(task)$predict(task)

  head(fortify(object))
  autoplot(object)
  autoplot(object, type = "histogram", binwidth = 1)
  autoplot(object, type = "residual")
}
```

---

**autoplot.ResampleResult**

*Plot for ResampleResult*

Description

Generates plots for `mlr3::ResampleResult`, depending on argument `type`:

- "boxplot" (default): Boxplot of performance measures.
- "histogram": Histogram of performance measures.
- "roc": ROC curve (1 - specificity on x, sensitivity on y). The predictions of the individual `mlr3::Resamplings` are merged prior to calculating the ROC curve (micro averaged). Requires package `precrec`. Additional arguments will be passed down to the respective `autoplot()` function in package `precrec`. Arguments `calc_avg` and `cb_alpha` are passed to `precrec::evalmod()`.
- "prc": Precision recall curve. See "roc".
- "prediction": Plots the learner prediction for a grid of points. Needs models to be stored. Set `store_models = TRUE` for `mlr3::resample`. For classification, we support tasks with exactly two features and learners with `predict_type` set to "response" or "prob". For regression, we support tasks with one or two features. For tasks with one feature we can print confidence bounds if the predict type of the learner was set to "se". For tasks with two features the predict type will be ignored.

Usage

```r
## S3 method for class 'ResampleResult'
autoplot(object, type = "boxplot", measure = NULL, predict_sets = "test", ...)
```
**Arguments**

- **object** ([mlr3::ResampleResult]).
- **type** (character(1)): Type of the plot. See description.
- **measure** ([mlr3::Measure]) Performance measure to use.
- **predict_sets** (character()): Only for type set to "prediction". Which points should be shown in the plot? Can be a subset of ("train", "test") or empty.
- ... (any): Additional arguments, passed down to the respective geom or plotting function.

**Value**

`ggplot2::ggplot()` object.

**Theme**

The `theme_mlr3()` and viridis color maps are applied by default to all `autoplot()` methods. To change this behavior set options(`mlr3.theme = FALSE`).

**References**


**Examples**

```r
if (requireNamespace("mlr3")) {
  library(mlr3)
  library(mlr3viz)

  task = tsk("sonar")
  learner = lrn("classif.rpart", predict_type = "prob")
  resampling = rsmp("cv")
  object = resample(task, learner, resampling)

  head(fortify(object))

  # Default: boxplot
  autoplot(object)

  # Histogram
  autoplot(object, type = " histogram", bins = 30)

  # ROC curve, averaged over resampling folds:
  autoplot(object, type = "roc")

  # ROC curve of joint prediction object:
```
autoplot(object$prediction(), type = "roc")

# Precision Recall Curve
autoplot(object, type = "prc")

# Prediction Plot
task = tsk("iris")$select(c("Sepal.Length", "Sepal.Width"))
resampling = rsmp("cv", folds = 3)
object = resample(task, learner, resampling, store_models = TRUE)
autoplot(object, type = "prediction")

---

**autoplot.TaskClassif**  
*Plot for Classification Tasks*

**Description**

Generates plots for **mlr3::TaskClassif**, depending on argument type:

- "target" (default): Bar plot of the target variable (default).
- "duo": Passes data and additional arguments down to `GGally::ggduo()`. columnsX is target, columnsY is features.
- "pairs": Passes data and additional arguments down to `GGally::ggpairs()`. Color is set to target column.

**Usage**

```r
## S3 method for class 'TaskClassif'
autoplot(object, type = "target", ...)
```

**Arguments**

- `object` *(mlr3::TaskClassif)*
- `type` *(character(1))*: Type of the plot. See description.
- `...` *(any)*: Additional argument, possibly passed down to the underlying plot functions.

**Value**

`ggplot2::ggplot` object.

**Theme**

The `theme_mlr3()` and viridis color maps are applied by default to all `autoplot()` methods. To change this behavior set `options(mlr3.theme = FALSE)`. 
Examples

```r
if (requireNamespace("mlr3")) {
  library(mlr3)
  library(mlr3viz)

  task = tsk("iris")

  head(fortify(task))
  autoplot(task)
  autoplot(task$clone()$select(c("Sepal.Length", "Sepal.Width"),
                             type = "pairs")
  autoplot(task, type = "duo")
}
```

---

### autoplot.TaskClust

#### Plot for Clustering Tasks

#### Description

Generates plots for `mlr3cluster::TaskClust`, depending on argument `type`:

- "pairs": Passes data and additional arguments down to `GGally::ggpairs()` (default).

#### Usage

```r
## S3 method for class 'TaskClust'
autoplot(object, type = "pairs", ...)
```

#### Arguments

- `object` (`mlr3cluster::TaskClust`).
- `type` (character(1)): Type of the plot. See description.
- `...` (any): Additional argument, passed down to the underlying `geom` or `plot` functions.

#### Value

`ggplot2::ggplot()` object.

#### Theme

The `theme_ml3()` and `viridis` color maps are applied by default to all `autoplot()` methods. To change this behavior set `options(mlr3.theme = FALSE)`. 
Examples

```r
if (requireNamespace("mlr")) {
  library(mlr)
  library(mlr3cluster)
  library(mlr3viz)

  task = mlr_tasks$get("usarrests")

  head(fortify(task))
  autoplot(task)
}
```

### autoplot.TaskRegr

**Plot for Regression Tasks**

#### Description

Generates plots for `mlr3::TaskRegr`, depending on argument type:

- "target": Box plot of target variable (default).
- "pairs": Passes data and additional arguments down to `GGally::ggpairs()`. Color is set to target column.

#### Usage

```r
## S3 method for class 'TaskRegr'
autoplot(object, type = "target", ...)
```

#### Arguments

- `object` (`mlr3::TaskRegr`).
- `type` (`character(1)`): Type of the plot. See description.
- `...` (`any`): Additional argument, passed down to the underlying `geom` or `plot` functions.

#### Value

`ggplot2::ggplot()` object.

#### Theme

The `theme_mlr3()` and viridis color maps are applied by default to all `autoplot()` methods. To change this behavior set `options(mlr3.theme = FALSE)`. 
Examples

```r
if (requireNamespace("mlr3")) {
  library(mlr3)
  library(mlr3viz)

  task = tsk("mtcars")
  task$select(c("am", "carb"))

  head(fortify(task))
  autoplot(task)
  autoplot(task, type = "pairs")
}
```

Description

Generates plots for `mlr3tuning::TuningInstanceSingleCrit`.

Usage

```r
## S3 method for class 'TuningInstanceSingleCrit'
autoplot(
  object,
  type = "marginal",
  cols_x = NULL,
  trafo = FALSE,
  learner = mlr3::lrn("regr.ranger"),
  grid_resolution = 100,
  ...)
```

Arguments

- `object` (mlr3tuning::TuningInstanceSingleCrit): Type of the plot. Available choices:
  - "marginal": scatter plots of hyperparameter versus performance. The color of the points shows the batch number.
  - "performance": scatter plots of batch number versus performance.
  - "parameter": scatter plots of batch number versus hyperparameter. The color of the points shows the performance.
  - "parallel": parallel coordinates plot. Parameter values are rescaled by `(x - mean(x)) / sd(x)`.
  - "points": scatter plot of two hyperparameters versus performance. The color of the points shows the performance.
• "surface": surface plot of 2 hyperparameters versus performance. The performance values are interpolated with the supplied mlr3::Learner.
• "pairs": plots all hyperparameters and performance values against each other.

cols_x (character())
Column names of hyperparameters. By default, all untransformed hyperparameters are plotted. Transformed hyperparameters are prefixed with x_domain_.

trafo (logical(1))
Determines if untransformed (FALSE) or transformed (TRUE) hyperparameters are plotted.

learner (mlr3::Learner)
Regression learner used to interpolate the data of the surface plot.

grid_resolution (numeric())
Resolution of the surface plot.

... (any): Additional arguments, possibly passed down to the underlying plot functions.

Value

ggplot2::ggplot() object.

Theme

The theme_mlr3() and viridis color maps are applied by default to all autoplot() methods. To change this behavior set options(mlr3.theme = FALSE).

Examples

if (requireNamespace("mlr3tuning") && requireNamespace("patchwork")) {
  library(mlr3tuning)

  learner = lrn("classif.rpart")
  learner$param_set$values$cp = to_tune(0.001, 0.1)
  learner$param_set$values$minsplit = to_tune(1, 10)

  instance = TuningInstanceSingleCrit$new(
    task = tsk("iris"),
    learner = learner,
    resampling = rsmp("holdout"),
    measure = msr("classif.ce"),
    terminator = trm("evals", n_evals = 10))

  tuner = tnr("random_search")
  tuner$optimize(instance)

  # plot performance versus batch number
  autoplot(instance, type = "performance")}
# plot cp values versus performance
autoplot(instance, type = "marginal", cols_x = "cp")

# plot transformed parameter values versus batch number
autoplot(instance, type = "parameter", trafo = TRUE)

# plot parallel coordinates plot
autoplot(instance, type = "parallel")

# plot pairs
autoplot(instance, type = "pairs")

plot_learner_prediction

## Plot for Learner Predictions

### Description
Generates a plot for the mlr3::Prediction of a single mlr3::Learner on a single mlr3::Task.

- For classification we support tasks with exactly two features and learners with predict_type set to "response" or "prob".
- For regression we support tasks with one or two features. For tasks with one feature we print confidence bounds if the predict type of the learner was set to "se". For tasks with two features the predict type will be ignored.

Note that this function is a wrapper around autoplot.ResampleResult() for a temporary mlr3::ResampleResult using mlr3::mlr_resamplings_holdout with ratio 1 (all observations in training set).

### Usage

```r
plot_learner_prediction(learner, task, grid_points = 100L, expand_range = 0)
```

### Arguments

- **learner** (mlr3::Learner).
- **task** (mlr3::Task).
- **grid_points** (integer(1))
  Resolution of the grid. For factors, ordered and logicals this value is ignored.
- **expand_range** (numeric(1))
  Expand the prediction range for numerical features.

### Value

ggplot2::ggplot() object.
```r
if (requireNamespace("mlr3")) {
  library(mlr3)
  library(mlr3viz)

  task = mlr3::tsk("pima")$select(c("age", "glucose"))
  learner = lrn("classif.rpart", predict_type = "prob")
  p = plot_learner_prediction(learner, task)
  print(p)
}
```

**predict_grid**

*Generates a data.table of evenly distributed points.*

**Description**

For each point we have the predicted class / regression value in column `response`. If the learner predicts probabilities, a column `.prob.response` is added that contains the probability of the predicted class.

**Usage**

`predict_grid(learners, task, grid_points, expand_range)`

**Arguments**

- `learners` list of trained learners, each learner belongs to one resampling iteration
- `task` the task all learners are trained on
- `grid_points` (int): see sequenize
- `expand_range` see sequenize

**theme_mlr3**

*mlr-org ggplot2 theme*

**Description**

The theme is heavily influenced and partly based on `ggpubr::theme_pubr()`. This theme is applied by default to all `autoplot()` methods in the mlr3 ecosystem. If you do not like it and want to use the default ggplot2 theme, you can add `+ theme_gray()` to the `autoplot()` call.
theme_mlr3

Usage

theme_mlr3(
    base_size = 12,
    base_family = "",
    border = FALSE,
    margin = TRUE,
    legend = c("top", "bottom", "left", "right", "none"),
    x.text.angle = 0
)

Arguments

base_size [integer]
Text font size.

base_family [character]
Font family.

border [logical]
If TRUE, adds a panel border.

margin [logical]
If FALSE, reduces the plot margin(s).

legend [character]
Specifies the legend position. Allowed values are one of c("top", "bottom", "left", "right", "none"). Default is "top".

x.text.angle [numeric]
Rotation angle of x axis tick labels. Default value is 0. Use 90 for vertical text.

Examples

library("ggplot2")
p = ggplot(mtcars, aes(x = wt, y = mpg)) +
    geom_point(aes(color = gear))

# Default plot
p

# theme_mlr3()
p + theme_mlr3()
Index

as_precrec, 3
autoplot(), 4, 16
autoplot::BenchmarkResult, 4
autoplot::Filter, 5
autoplot::LearnerClassifCVGlmnet, 6
autoplot::LearnerClassifGlmnet
  (autoplot::LearnerClassifCVGlmnet), 6
autoplot::LearnerClassifRpart, 7
autoplot::LearnerClustHierarchical, 9
autoplot::LearnerRegrCVGlmnet
  (autoplot::LearnerClassifCVGlmnet), 6
autoplot::LearnerRegrGlmnet
  (autoplot::LearnerClassifCVGlmnet), 6
autoplot::LearnerRegrRpart
  (autoplot::LearnerClassifRpart), 7
autoplot::OptimInstanceSingleCrit, 10
autoplot::PredictionClassif, 12
autoplot::PredictionClust, 13
autoplot::PredictionRegr, 15
autoplot::ResampleResult, 16
autoplot::ResampleResult(), 23
autoplot::TaskClassif, 18
autoplot::TaskClust, 19
autoplot::TaskRegr, 20
autoplot::TuningInstanceSingleCrit, 21

bbotk::OptimInstanceSingleCrit, 10
factoextra::fviz_dend(), 9
GGally::ggduo(), 18
GGally::ggpairs(), 18–20
ggfortify::autoplot.kmeans, 13
ggparty::autoplot.party(), 7, 8
ggplot2::ggplot(), 4, 6–9, 11, 13–15, 17–20, 22, 23

mlr3::BenchmarkResult, 4
mlr3::Learner, 4, 11, 22, 23
mlr3::LearnerClassifRpart, 8
mlr3::LearnerRegrRpart, 8
mlr3::Measure, 4, 12, 13, 17
mlr3::mlr_learners_classif.rpart, 8
mlr3::mlr_learners_regr.rpart, 8
mlr3::mlr_resamplings_holdout, 23
mlr3::Prediction, 23
mlr3::PredictionClassif, 12
mlr3::PredictionRegr, 15
mlr3::ResampleResult, 16, 17, 23
mlr3::Resampling, 4, 16
mlr3::Task, 4, 23
mlr3::TaskClassif, 18
mlr3::TaskRegr, 20
mlr3cluster::LearnerClustAgnes, 9
mlr3cluster::LearnerClustDiana, 9
mlr3cluster::LearnerClustHclust, 9
mlr3cluster::PredictionClust, 13, 14
mlr3cluster::TaskClust, 14, 19
mlr3filters::Filter, 5
mlr3learners::LearnerClassifGlmnet, 7
mlr3learners::LearnerRegrCVGlmnet, 7
mlr3learners::LearnerRegrGlmnet, 7
mlr3learners::mlr_learners_classif.cv_glmnet, 6
mlr3learners::mlr_learners_classif.glmnet, 6
mlr3learners::mlr_learners_regr.cv_glmnet, 6
mlr3learners::mlr_learners_regr.glmnet, 6
mlr3tuning::TuningInstanceSingleCrit, 21
mlr3viz (mlr3viz-package), 2
mlr3viz-package, 2

plot_learner_prediction, 23
precrec::evalmod(), 3, 4, 16
INDEX

precrec::mmdata(), 4
predict_grid, 24

theme_mlr3, 24
theme_mlr3(), 5–9, 11, 13–15, 17–20, 22