Package ‘mlr3viz’

March 5, 2024

Title  Visualizations for 'mlr3'

Version  0.8.0

Description  Visualization package of the 'mlr3' ecosystem. It features plots for mlr3 objects such as tasks, learners, predictions, benchmark results, tuning instances and filters via the 'autoplot()' generic of 'ggplot2'. The package draws plots with the 'viridis' color palette and applies the minimal theme. 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 (>= 0.7.3), cluster, GGally, ggdendro, ggfortify (>= 0.4.11), ggparty, 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  7.3.1

Collate  'BenchmarkResult.R' 'Filter.R' 'LearnerClassif.R'
          'LearnerClassifCVGlmnet.R' 'LearnerClassifGlmnet.R'
          'LearnerClassifRpart.R' 'LearnerClustHierarchical.R'
          'LearnerRegr.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'
mlr3viz-package

mlr3viz: Visualizations for 'mlr3'

Description

Visualization package of the 'mlr3' ecosystem. It features plots for mlr3 objects such as tasks, learners, predictions, benchmark results, tuning instances and filters via the 'autoplot()' generic of 'ggplot2'. The package draws plots with the 'viridis' color palette and applies the minimal theme. Visualizations include barplots, boxplots, histograms, ROC curves, and Precision-Recall curves.
as_precrec

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://mlr3viz.mlr-org.com)
- [https://github.com/mlr-org/mlr3viz](https://github.com/mlr-org/mlr3viz)

---

**Description**

Converts to a format which is understood by `precrec::evalmod()` of package `precrec`.

**Usage**

```r
as_precrec(object)
```

## S3 method for class 'PredictionClassif'

```r
as_precrec(object)
```

## S3 method for class 'ResampleResult'

```r
as_precrec(object)
```

## S3 method for class 'BenchmarkResult'

```r
as_precrec(object)
```

**Arguments**

- **object** (any)
  
  Object to convert.
Value

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

References


### autoplot.BenchmarkResult

*Plots for Benchmark Results*

#### Description

Visualizations for `mlr3::BenchmarkResult`. The argument `type` controls what kind of plot is drawn. Possible choices are:

- "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`.
- "prc": Precision recall curve. See "roc".

#### Usage

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

#### Arguments

- `object` *(mlr3::BenchmarkResult)*.
- `type` *(character(1))*: Type of the plot. See description.
- `measure` *(mlr3::Measure)*: Performance measure to use.
- `theme` *(ggplot2::theme())*: The `ggplot2::theme_minimal()` is applied by default to all plots.
- `...` *(ignored)*.
Value

```r
ggplot2::ggplot()
```

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")
}
```
theme autoplot2::theme()
The autoplot2::theme_minimal() is applied by default to all plots.

... (ignored).

Value

autoplot2::ggplot().

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)
}
```

---

**autoplot.LearnerClassif**

*Plot for Classification Learners*

**Description**

Visualizations for mlr3::LearnerClassif. The argument type controls what kind of plot is drawn. Possible choices are:

- "prediction" (default): Decision boundary of the learner and the true class labels.

**Usage**

```r
## S3 method for class 'LearnerClassif'
autoplot(
  object,
  type = "prediction",
  task,
  grid_points = 100L,
  expand_range = 0,
  theme = theme_minimal(),
  ...)
```
Arguments

- object (mlr3::LearnerClassif): 
- type (character(1)): Type of the plot. See description.
- task (mlr3::Task): Train task.
- grid_points (integer(1)): Number of grid points per feature dimension.
- expand_range (numeric(1)): Expand the range of the grid.
- theme (ggplot2::theme()): The ggplot2::theme_minimal() is applied by default to all plots.
- ... (ignored).

Value

ggplot2::ggplot().

Examples

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

  task = tsk("pima")$select(c("age", "pedigree"))
  learner = lrn("classif.rpart", predict_type = "prob")
  learner$train(task)

  autoplot(learner, type = "prediction", task)
}
```

autplot.LearnerClassifCVGlmnet

Plots for GLMNet Learners

Description

Visualizations for mlr3learners::LearnerClassifGlmnet. The argument type controls what kind of plot is drawn. Possible choices are:

- "prediction" (default): Decision boundary of the learner and the true class labels.
- "ggfortify": Visualizes the model using the package ggfortify.
Usage

```r
## S3 method for class 'LearnerClassifCVGlmnet'
autoplot(
  object,
  type = "prediction",
  task = NULL,
  grid_points = 100L,
  expand_range = 0,
  theme = theme_minimal(),
  ...
)

## S3 method for class 'LearnerClassifGlmnet'
autoplot(
  object,
  type = "prediction",
  task = NULL,
  grid_points = 100L,
  expand_range = 0,
  theme = theme_minimal(),
  ...
)

## S3 method for class 'LearnerRegrCVGlmnet'
autoplot(
  object,
  type = "prediction",
  task = NULL,
  grid_points = 100L,
  expand_range = 0,
  theme = theme_minimal(),
  ...
)

## S3 method for class 'LearnerRegrGlmnet'
autoplot(
  object,
  type = "prediction",
  task = NULL,
  grid_points = 100L,
  expand_range = 0,
  theme = theme_minimal(),
  ...
)
```
Arguments

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

type (character(1)):
Type of the plot. See description.

task (mlr3::Task)
Train task.

grid_points (integer(1))
Number of grid points per feature dimension.

expand_range (numeric(1))
Expand the range of the grid.

theme (ggplot2::theme())
The ggplot2::theme_minimal() is applied by default to all plots.

Value

ggplot2::ggplot().

References


Examples

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

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

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

## End(Not run)
**Description**

Visualizations for `mlr3::LearnerClassifRpart`. The argument `type` controls what kind of plot is drawn. Possible choices are:

- "prediction" (default): Decision boundary of the learner and the true class labels.
- "ggparty": Visualizes the tree using the package `ggparty`.

**Usage**

```r
## S3 method for class 'LearnerClassifRpart'
autoplot(
  object,
  type = "prediction",
  task = NULL,
  grid_points = 100L,
  expand_range = 0,
  theme = theme_minimal(),
  ...
)
```

```r
## S3 method for class 'LearnerRegrRpart'
autoplot(
  object,
  type = "prediction",
  task = NULL,
  grid_points = 100L,
  expand_range = 0,
  theme = theme_minimal(),
  ...
)
```

**Arguments**

- `object` (mlr3::LearnerClassifRpart | mlr3::LearnerRegrRpart):
- `type` (character(1)):
  Type of the plot. See description.
- `task` (mlr3::Task):
  Train task.
- `grid_points` (integer(1)):
  Number of grid points per feature dimension.
autoplot.LearnerClustHierarchical

Plots for Hierarchical Clustering Learners

Description

Visualizations for hierarchical clusters. The argument type controls what kind of plot is drawn. Possible choices are:

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

Usage

```r
## S3 method for class 'LearnerClustHierarchical'
autoplot(
  object,
  type = "dend",
)```
task = NULL,
theme = theme_minimal(),
theme_dendro = TRUE,
...
)

Arguments

object (mlr3cluster::LearnerClustAgnes | mlr3cluster::LearnerClustDiana | mlr3cluster::LearnerClustHclust).
type (character(1)):
  Type of the plot. See description.
task (mlr3::Task)
  Optionally, pass the task to add labels of observations to a hclust dendrogram.
  Labels are set via the row names of the task.
theme (ggplot2::theme())
  The ggplot2::theme_minimal() is applied by default to all plots.
theme_dendro (logical(1))
  If TRUE (default), the special dendrogram theme from ggdendro package is used
  in plot "dend". Set to FALSE to use the theme passed in theme.
...

Value

ggplot2::ggplot().

Examples

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

  task = tsk("usarrests")

  # agnes clustering
  learner = lrn("clust.agnes")
  learner$train(task)
  autoplot(learner)

  # diana clustering
  learner = lrn("clust.diana")
  learner$train(task)
  autoplot(learner)

  # hclust clustering
  learner = lrn("clust.hclust")
  learner$train(task)
  autoplot(learner, type = "scree")
}
autoplot.LearnerRegr  Plot for Regression Learners

Description
Visualizations for mlr3::LearnerRegr. The argument type controls what kind of plot is drawn. Possible choices are:

- "prediction" (default): Decision boundary of the learner and the true class labels.

Usage
```r
## S3 method for class 'LearnerRegr'
autoplot(
  object,
  type = "prediction",
  task,
  grid_points = 100L,
  expand_range = 0,
  theme = theme_minimal(),
  ...
)
```

Arguments
- object: (mlr3::LearnerRegr).
- type: (character(1)): Type of the plot. See description.
- task: (mlr3::Task) Train task.
- grid_points: (integer(1)) Number of grid points per feature dimension.
- expand_range: (numeric(1)) Expand the range of the grid.
- theme: (ggplot2::theme()) The ggplot2::theme_minimal() is applied by default to all plots.
- ... (ignored).

Value
- ggplot2::ggplot().
Examples

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

  task = tsk("mtcars")$select(c("am", "carb"))
  learner = lrn("regr.rpart")
  learner$train(task)

  autoplot(learner, type = "prediction", task)
}
```

---

**autoplot.OptimInstanceSingleCrit**

*Plots for Optimization Instances*

**Description**

Visualizations for `bbotk::OptimInstanceSingleCrit`. The argument `type` controls what kind of plot is drawn. Possible choices are:

- "marginal" (default): Scatter plots of x versus y. The color of the points shows the batch number.
- "performance": Scatter plots of batch number versus y
- "parameter": Scatter plots of batch number versus input. The color 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 color 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 `mlr3::Learner`.
- "pairs": Plots all x and y values against each other.
- "incumbent": Plots the incumbent versus the number of configurations.

**Usage**

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

object (bbotk::OptimInstanceSingleCrit).

**type** (character(1)):
Type of the plot. See description.

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 `x_domain_`.

**trafo** (logical(1))
If FALSE (default), the untransformed x values are plotted. If TRUE, the transformed x values are plotted.

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

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

**batch** (integer())
The batch number(s) to limit the plot to. The default is all batches.

**theme** (ggplot2::theme())
The `ggplot2::theme_minimal()` is applied by default to all plots.

... (ignored).

Value

`ggplot2::ggplot()`.

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
print(autoplot(instance, type = "performance"))

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

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

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

# plot incumbent
print(autoplot(instance, type = "incumbent"))

---

**autplot.PredictionClassif**

*Plots for Classification Predictions*

**Description**

Visualizations for *mlr3::PredictionClassif*. The argument type controls what kind of plot is drawn. Possible choices are:

- "stacked" (default): Stacked barplot of true and estimated class labels.
- "roc": ROC curve (1 - specificity on x, sensitivity on y). Requires package `precrec`.
- "prc": Precision recall curve. Requires package `precrec`.
- "threshold": Systematically varies the threshold of the *mlr3::PredictionClassif* object and plots the resulting performance as returned by measure.

**Usage**

```r
## S3 method for class 'PredictionClassif'
autplot(
  object,
  type = "stacked",
  measure = NULL,
```
```r
theme = theme_minimal(),
...
)

Arguments

object (mlr3::PredictionClassif).
type (character(1)):
Type of the plot. See description.
measure (mlr3::Measure)
Performance measure to use.
theme (ggplot2::theme())
The ggplot2::theme_minimal() is applied by default to all plots.
...(ignored).

Value

ggplot2::ggplot().

References


Examples

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

task = tsk("spam")
learner = lrn("classif.rpart", predict_type = "prob")
object = learner$train(task)$predict(task)

head(fortify(object))
autoplot(object)
autoplot(object, type = "roc")
autoplot(object, type = "prc")
}
```
Description

Visualizations for mlr3cluster::PredictionClust. The argument type controls what kind of plot is drawn. Possible choices are:

- "scatter" (default): scatterplot with correlation values and colored cluster assignments.
- "sil": Silhouette plot with mean silhouette value as the 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,            # mlr3cluster::PredictionClust.
  task,              # mlr3cluster::TaskClust.
  row_ids = NULL,    # Row ids to subset task data to ensure that only the data used to make predictions are shown in plots.
  type = "scatter", # (character(1)): Type of the plot. See description.
  theme = theme_minimal(), # The ggplot2::theme_minimal() is applied by default to all plots.
  ...                # (ignored).
)
```

Arguments

- `object`: (mlr3cluster::PredictionClust).
- `task`: (mlr3cluster::TaskClust).
- `row_ids`: (integer()) 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.
- `theme`: (ggplot2::theme())
- `...`: (ignored).

Value

`ggplot2::ggplot()`.

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

Visualizations for `mlr3::PredictionRegr`. The argument `type` controls what kind of plot is drawn. Possible choices are:

- "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).
- "confidence": Scatterplot of "true" response vs. "predicted" response with confidence intervals. Error bars calculated as `object$reponse + quantile * object$se` and so only possible with `predict_type = "se"`. `geom_abline()` with `slope = 1` is added to the plot.

Usage

```r
## S3 method for class 'PredictionRegr'
autoplot(
  object,
  type = "xy",
  binwidth = NULL,
  theme = theme_minimal(),
  quantile = 1.96,
  ...
)
```
Arguments

object (mlr3::PredictionRegr).

type (character(1)):
Type of the plot. See description.

binwidth (integer(1))
Width of the bins for the histogram.

theme (ggplot2::theme())
The ggplot2::theme_minimal() is applied by default to all plots.

quantile (numeric(1))
Quantile multiplier for standard errors for type="confidence". Default 1.96.

... (ignored).

Value

ggplot2::ggplot().

Examples

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")

  if (requireNamespace("mlr3learners")) {
    library(mlr3learners)
    learner = lrn("regr.ranger", predict_type = "se")
    object = learner$train(task)$predict(task)
    autoplot(object, type = "confidence")
  }
}

autoplot.ResampleResult

Plots for Resample Results
Description

Visualizations for `mlr3::ResampleResult`. The argument `type` controls what kind of plot is drawn. Possible choices are:

- "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`.
- "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",
  binwidth = NULL,
  theme = theme_minimal(),
  ...
)
```

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.
- **binwidth** (`integer(1)`) Width of the bins for the histogram.
- **theme** (`ggplot2::theme()`) The `ggplot2::theme_minimal()` is applied by default to all plots.
- **...** (ignored).
autoplot.TaskClassif

Plots for Classification Tasks

Value

ggplot2::ggplot().

References


Examples

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

  task = tsk("sonar")
  learner = lrn("classif.rpart", predict_type = "prob")
  resampling = rsmp("cv", folds = 3)
  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")
}
```
Description

Visualizations for mlr3::TaskClassif. The argument type controls what kind of plot is drawn. Possible choices are:

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

Usage

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

Arguments

- `object` (mlr3::TaskClassif).
- `type` (character(1)):
  Type of the plot. See description.
- `theme` (ggplot2::theme())
  The ggplot2::theme_minimal() is applied by default to all plots.
- `...` (ignored).

Value

`ggplot2::ggplot()`.

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  Plots for Clustering Tasks

Description

Visualizations for mlr3cluster::TaskClust. The argument type controls what kind of plot is drawn. Possible choices are:

- "pairs" (default): Passes data GGally::ggpairs().

Usage

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

Arguments

- **object** (mlr3cluster::TaskClust).
- **type** (character(1)): Type of the plot. See description.
- **theme** (ggplot2::theme())
  The ggplot2::theme_minimal() is applied by default to all plots.
- **...** (ignored).

Value

ggplot2::ggplot().

Examples

```r
if (requireNamespace("mlr3")) {
  library(mlr3)
  library(mlr3cluster)
  library(mlr3viz)
  
  task = mlr_tasks$get("usarrests")
  
  head(fortify(task))
  autoplot(task)
}
```
Description

Visualizations for mlr3::TaskRegr. The argument type controls what kind of plot is drawn. Possible choices are:

- "target" (default): Box plot of the target variable.
- "pairs": Passes data to GGally::ggpairs(). Color is set to target column.

Usage

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

Arguments

- `object` (mlr3::TaskRegr).
- `type` (character(1)): Type of the plot. See description.
- `theme` (ggplot2::theme()): The ggplot2::theme_minimal() is applied by default to all plots.
- `...` (ignored).

Value

ggplot2::ggplot().

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

Visualizations for `mlr3tuning::TuningInstanceSingleCrit`. The argument `type` controls what kind of plot is drawn. Possible choices are:

- "marginal" (default): Scatter plots of x versus y. The color of the points shows the batch number.
- "performance": Scatter plots of batch number versus y
- "parameter": Scatter plots of batch number versus input. The color of the points shows the y values.
- "parallel": Parallel coordinates plot. Hyperparameters are rescaled by \( \frac{x - \text{mean}(x)}{\text{sd}(x)} \).
- "points": Scatter plot of two x dimensions versus. The color 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 `mlr3::Learner`.
- "pairs": Plots all x and y values against each other.
- "incumbent": Plots the incumbent versus the number of configurations.

Usage

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

Arguments

- `object` (mlr3tuning::TuningInstanceSingleCrit): The object to be plotted.
- `type` (character(1)): Type of the plot. See description.
- `cols_x` (character()): Column names of hyperparameters. By default, all untransformed hyperparameters are plotted. Transformed hyperparameters are prefixed with `x_domain_`. 
autoplot.TuningInstanceSingleCrit

trafo (logical(1))
If FALSE (default), the untransformed hyperparameters are plotted. If TRUE, the transformed 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.

theme (ggplot2::theme())
The ggplot2::theme_minimal() is applied by default to all plots.

... (ignored).

Value

ggplot2::ggplot().

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

Plots for Learner Predictions

Description

Visualizations 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 the training set).

Usage

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().

Examples

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