Package ‘mlr3viz’

November 2, 2020

Title  Visualizations for 'mlr3'

Version 0.5.0

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, utils

Suggests cluster, distr6 (>= 1.4.4), factoextra, GGally, ggfortify (>= 0.4.11), ggreparty, glmnet, lgr, mlr3 (>= 0.6.0), mlr3cluster, mlr3filters, mlr3learners, mlr3proba (>= 0.2.3), partykit, precenc, rpart, stats, survival, testthat

Encoding UTF-8

LazyData true

NeedsCompilation no

RoxygenNote 7.1.1

Author Michel Lang [cre, aut] (<https://orcid.org/0000-0001-9754-0393>), Patrick Schratz [aut] (<https://orcid.org/0000-0003-0748-6624>), Raphael Sonabend [aut] (<https://orcid.org/0000-0001-9225-4654>), Damir Pulatov [ctb]

Maintainer Michel Lang <michellang@gmail.com>

Repository CRAN

Date/Publication 2020-11-02 11:10:02 UTC
**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)

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 `precrec::evalmod()` of package `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**

<table>
<thead>
<tr>
<th>object</th>
<th>(any)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object to convert.</td>
<td></td>
</tr>
</tbody>
</table>

**Value**

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

**References**

aut plot.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 aut plot() function in package precrec. Arguments calc_avg and cb_alpha are passed to precrec::evalmod().
- "prc": Precision recall curve. See "roc".

Usage

## S3 method for class 'BenchmarkResult'
aut plot(object, type = "boxplot", measure = NULL, ...)

Arguments

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

Value

ggplot2::ggplot() object.

References


Examples

library(mlr3)
library(mlr3viz)
tasks = tsk(c("pima", "sonar"))
learner = lrn(c("classif.featureless", "classif.rpart"),
autoplot.Filter

```r
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")
```

---

**Description**

Generates plots for `mlr3filters::Filter`, depending on argument `type`:

- **"barplot"** (default): Bar plot of filter scores.

**Usage**

```r
## S3 method for class SingleVarFilter
autoplot(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.

**Examples**

```r
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 \texttt{mlr3learners::mlr_learners_classif.glmnet}, \texttt{mlr3learners::mlr_learners_regr.glmnet}, \texttt{mlr3learners::mlr_learners_classif.cv_glmnet} and \texttt{mlr3learners::mlr_learners_regr.cv_glmnet} using the package \texttt{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` (\texttt{mlr3learners::LearnerClassifGlmnet | mlr3learners::LearnerRegrGlmnet | mlr3learners::LearnerRegrCVGlmnet | mlr3learners::LearnerRegrCVGlmnet}).
- `...` (any): Additional arguments, passed down to \texttt{ggparty::autoplot.party()}.  

Value

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

References


Examples

```r
## 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)

---

## autoplot.LearnerClassifRpart

*Plot for LearnerClassifRpart / LearnerRegrRpart*

### 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

```
## 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.
Examples

```r
library(mlr3)
library(mlr3cluster)
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

Visualize dendrograms for hierarchical clusterers using the package `factoextra`.

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

Usage

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

Arguments

- `object` (mlr3cluster::LearnerClustAgnes | mlr3cluster::LearnerClustDiana).
- `...` (any): Additional arguments, passed down to function `factoextra::fviz_dend()` in package `factoextra`.

Value

`ggplot2::ggplot()` object.

Examples

```r
library(mlr3)
library(mlr3cluster)
library(mlr3viz)

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

# diana clustering
learner = mlr_learners$get("clust.diana")
learner$train(task)
autoplot(learner,
  k = learner$param_set$values$k, rect_fill = TRUE,
  rect = TRUE, rect_border = "red")

---

autplot.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 `precrec`.
- "prc": Precision recall curve. Requires package `precrec`.

**Usage**

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

**Arguments**

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

**Value**

`ggplot2::ggplot()` object.

**References**

Examples

```r
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")
```

---

**autoplot.PredictionClust**

*Plot for 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.
autoplot.PredictionRegr

References

Examples
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)

autoplot.PredictionRegr

Plot for PredictionRegr

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
## 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.

Examples

```r
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")
```

Description

Generates plots for `mlr3proba::PredictionSurv`, depending on argument `type`:

- "calib" (default): Calibration plot comparing the average predicted survival distribution to a Kaplan-Meier prediction, this is not a comparison of a stratified crank or lp prediction. `object` must have distr prediction. `geom_line()` is used for comparison split between the prediction (Pred) and Kaplan-Meier estimate (KM). In addition labels are added for the x (T) and y (S(T)) axes.

Usage

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

Arguments

- `object` (mlr3proba::PredictionSurv): (mnr3proba::PredictionSurv).
- `type` (character(1)): Type of the plot. See description.
autoplot.ResampleResult

**task**
(mlr3proba::TaskSurv)
If type = "calib" then task is passed to $predict in the Kaplan-Meier learner.

**row_ids**
(integer())
If type = "calib" then row_ids is passed to $predict in the Kaplan-Meier learner.

**times**
(numeric())
If type = "calib" then times is the values on the x-axis to plot over, if NULL uses all times from task.

... (any): Additional arguments, currently unused.

**Examples**

```r
library(mlr)
library(mlr3proba)
library(mlr3viz)

learn = lrn("surv.coxph")
task = tsk("rats")
p = learn$train(task, row_ids = 1:100)$predict(task, row_ids = 101:200)
autoplot(p, type = "calib", task = task)
```

**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'
aplot(object, type = "boxplot", measure = NULL, predict_sets = "test", ...)
```
autoplot.ResampleResult

Arguments

- **object**: (mlr3::ResampleResult).
- **type**: (character(1)). Type of the plot. See description.
- **measure**: (mlr3::Measure).
- **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.

References


Examples

```r
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)
```
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::gguo(). 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.

Examples

```r
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.

Examples
```r
library(mlr3)
library(mlr3cluster)
library(mlr3viz)

task = mlr_tasks$get("usarrests")

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

autoplot.TaskDens  Plot for Density Tasks

Description
Generates plots for mlr3proba::TaskDens.
Usage

### S3 method for class 'TaskDens'

`autoplot(object, type = "dens", ...)`

Arguments

- `object`: (mlr3proba::TaskDens).
- `type`: (character(1)): Type of the plot. Available choices:
  - "dens": histogram density estimator (default) with `ggplot2::geom_histogram()`.
  - "freq": histogram frequency plot with `ggplot2::geom_histogram()`.
  - "overlay": histogram with overlaid density plot with `ggplot2::geom_histogram()` and `ggplot2::geom_density()`.
  - "freqpoly": frequency polygon plot with `ggplot2::geom_freqpoly`.
- `...`: (any): Additional arguments, possibly passed down to the underlying plot functions.

Value

`ggplot2::ggplot()` object.

Examples

```r
library(mlr3)
library(mlr3proba)
task = tsk("precip")
head(fortify(task))
autoplot(task, bins = 15)
autoplot(task, type = "freq", bins = 15)
autoplot(task, type = "overlay", bins = 15)
autoplot(task, type = "freqpoly", bins = 15)
```

---

`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

### S3 method for class 'TaskRegr'

`autoplot(object, type = "target", ...)"
Arguments

object (mlr3::TaskRegr).

Type of the plot. See description.

... (any): Additional argument, passed down to the underlying geom or plot functions.

Value

ggplot2::ggplot() object.

Examples

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 mlr3proba::TaskSurv, depending on argument type:

- "target": Calls GGally::ggsurv() on a survival::survfit() object.
- "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

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

Arguments

object (mlr3proba::TaskSurv).

Type of the plot. Available choices:

... (any): Additional argument, passed down to $formula of mlr3proba::TaskSurv or the underlying plot functions.
Value

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

Examples

```
library(mlr3)
library(mlr3viz)
library(mlr3proba)

task = tsk("lung")

head(fortify(task))
autoplot(task)
autoplot(task, rhs = "sex")
autoplot(task, type = "duo")
```

---

**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**

```
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.

Examples

```r
library(mlr3)
library(mlr3viz)

task = 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
## Index

<table>
<thead>
<tr>
<th>Package</th>
<th>Page(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>as_precrec</td>
<td>3</td>
</tr>
<tr>
<td>autoplot()</td>
<td>4, 13</td>
</tr>
<tr>
<td>autoplot::BenchmarkResult</td>
<td>4</td>
</tr>
<tr>
<td>autoplot::Filter</td>
<td>5</td>
</tr>
<tr>
<td>autoplot::LearnerClassifCVGlmnet</td>
<td>6</td>
</tr>
<tr>
<td>autoplot::LearnerClassifGlmnet</td>
<td>6</td>
</tr>
<tr>
<td>autoplot::LearnerClassifRpart</td>
<td>7</td>
</tr>
<tr>
<td>autoplot::LearnerClustHierarchical</td>
<td>8</td>
</tr>
<tr>
<td>autoplot::LearnerRegrCVGlmnet</td>
<td>6</td>
</tr>
<tr>
<td>autoplot::LearnerRegrGlmnet</td>
<td>6</td>
</tr>
<tr>
<td>autoplot::LearnerRegrRpart</td>
<td>7</td>
</tr>
<tr>
<td>autoplot::PredictionClassif</td>
<td>9</td>
</tr>
<tr>
<td>autoplot::PredictionClust</td>
<td>10</td>
</tr>
<tr>
<td>autoplot::PredictionRegr</td>
<td>11</td>
</tr>
<tr>
<td>autoplot::PredictionSurv</td>
<td>12</td>
</tr>
<tr>
<td>autoplot::ResampleResult</td>
<td>13</td>
</tr>
<tr>
<td>autoplot::ResampleResult::</td>
<td>13, 14, 19</td>
</tr>
<tr>
<td>autoplot::TaskClassif</td>
<td>15</td>
</tr>
<tr>
<td>autoplot::TaskRegr</td>
<td>17, 18</td>
</tr>
<tr>
<td>factoextra::fviz_dend()</td>
<td>8</td>
</tr>
<tr>
<td>GGally::ggduo()</td>
<td>15, 18</td>
</tr>
<tr>
<td>GGally::ggpairs()</td>
<td>15–18</td>
</tr>
<tr>
<td>GGally::ggsurv()</td>
<td>18</td>
</tr>
<tr>
<td>ggfortify::autoplot.kmeans</td>
<td>10</td>
</tr>
<tr>
<td>ggparty::autoplot.party</td>
<td>6, 7</td>
</tr>
<tr>
<td>ggplot2::geom_density</td>
<td>17</td>
</tr>
<tr>
<td>ggplot2::geom_histogram</td>
<td>17</td>
</tr>
<tr>
<td>mlr3::BenchmarkResult</td>
<td>4</td>
</tr>
<tr>
<td>mlr3::Learner</td>
<td>4, 19</td>
</tr>
<tr>
<td>mlr3::LearnerClassifRpart</td>
<td>7</td>
</tr>
<tr>
<td>mlr3::LearnerRegrRpart</td>
<td>7</td>
</tr>
<tr>
<td>mlr3::Measure</td>
<td>4, 14</td>
</tr>
<tr>
<td>mlr3::mlr_learners_classif.rpart</td>
<td>7</td>
</tr>
<tr>
<td>mlr3::mlr_learners_regr.rpart</td>
<td>7</td>
</tr>
<tr>
<td>mlr3::mlr_resamplings_holdout</td>
<td>19</td>
</tr>
<tr>
<td>mlr3::PredictionClassif</td>
<td>9</td>
</tr>
<tr>
<td>mlr3::PredictionRegr</td>
<td>11</td>
</tr>
<tr>
<td>mlr3::PredictionResult</td>
<td>13, 14, 19</td>
</tr>
<tr>
<td>mlr3::Resampling</td>
<td>4, 13</td>
</tr>
<tr>
<td>mlr3::Task</td>
<td>4, 19</td>
</tr>
<tr>
<td>mlr3::mlr_learners_classif.cv_glmnet</td>
<td>6</td>
</tr>
<tr>
<td>mlr3::mlr_learners_regr.cv_glmnet</td>
<td>6</td>
</tr>
<tr>
<td>mlr3::mlr_learners_classif.glmnet</td>
<td>6</td>
</tr>
<tr>
<td>mlr3::mlr_learners_regr.glmnet</td>
<td>6</td>
</tr>
<tr>
<td>mlr3::mlr_learners_regr.cv_rpart</td>
<td>6</td>
</tr>
<tr>
<td>mlr3::mlr_learners_regr.glmnet</td>
<td>6</td>
</tr>
<tr>
<td>mlr3::mlr_learners_regr.rpart</td>
<td>6</td>
</tr>
<tr>
<td>mlr3::PredictionSurv</td>
<td>12</td>
</tr>
<tr>
<td>mlr3::TaskDens</td>
<td>16, 17</td>
</tr>
<tr>
<td>mlr3::TaskSurv</td>
<td>13, 18</td>
</tr>
<tr>
<td>mlr3viz(mlr3viz-package)</td>
<td>2</td>
</tr>
<tr>
<td>mlr3viz-package</td>
<td>2</td>
</tr>
</tbody>
</table>
plot_learner_prediction, 19
precrec::evalmod(), 3, 4, 13
precrec::mmdata(), 3
predict_grid, 20

survival::survfit(), 18