Package ‘mlr3summary’

April 24, 2024

Title Model and Learner Summaries for 'mlr3'

Version 0.1.0

Description Concise and interpretable summaries for machine learning models and learners of the 'mlr3' ecosystem.
The package takes inspiration from the summary function for (generalized) linear models but extends it to non-parametric machine learning models, based on generalization performance, model complexity, feature importances and effects, and fairness metrics.

License LGPL-3

Depends R (>= 3.5.0)

Imports backports, checkmate (>= 2.0.0), data.table, mlr3 (>= 0.12.0), mlr3misc, cli, future.apply (>= 1.5.0),

Suggests testthat (>= 3.1.0), iml, mlr3pipelines, mlr3fairness, mlr3learners, fastshap, ranger, rpart

Config/testthat/edition 3

Language en-US

NeedsCompilation no

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Author Susanne Dandl [aut, cre] (<https://orcid.org/0000-0003-4324-4163>),
Marc Becker [aut] (<https://orcid.org/0000-0002-8115-0400>),
Bernd Bischl [aut] (<https://orcid.org/0000-0001-6002-6980>),
Giuseppe Casalicchio [aut] (<https://orcid.org/0000-0001-5324-5966>),
Ludwig Bothmann [aut] (<https://orcid.org/0000-0002-1471-6582>)

Maintainer Susanne Dandl <dandls.datascience@gmail.com>

Repository CRAN

Date/Publication 2024-04-24 14:20:02 UTC

R topics documented:

`mlr3summary-package` *mlr3summary: Model and Learner Summaries for 'mlr3'*

Description

Concise and interpretable summaries for machine learning models and learners of the 'mlr3' ecosystem. The package takes inspiration from the summary function for (generalized) linear models but extends it to non-parametric machine learning models, based on generalization performance, model complexity, feature importances and effects, and fairness metrics.

Author(s)

Maintainer: Susanne Dandl `<dandls.datascience@gmail.com>` (ORCID)

Authors:

- Marc Becker `<marcbecker@posteo.de>` (ORCID)
- Bernd Bischl `<bernd_bischl@gmx.net>` (ORCID)
- Giuseppe Casalicchio `<giuseppe.casalicchio@stat.uni-muenchen.de>` (ORCID)
- Ludwig Bothmann `<ludwig.bothmann@stat.uni-muenchen.de>` (ORCID)

`credit` *German Credit Dataset (Preprocessed)*

Description

Preprocessed version of the German Credit Risk dataset available on kaggle, based on the Statlog (German credit dataset) of Hofmann (1994) available on UCI.

Format

A data frame with 522 and 6 variables:

age age of the customer [19-75]

sex sex of the customer (female, male)

saving.accounts saving account balance of the customer (little, moderate, rich)

duration payback duration of credit (in month) [6-72]

credit.amount credit amount [276-18424]

risk whether the credit is of low/good or high/bad risk (bad, good)

Details

The dataset was further adapted: rows with missing values were removed, low-cardinal classes were binned, classes of the job feature were renamed, the features on the savings and checking account were defined as ordinal variables, and all feature names were transposed to lower. Only a subset of features was selected: "age", "sex", "saving.accounts", "duration", "credit.amount", "risk".

References

Hofmann, Hans (1994). "Statlog (German Credit Data)." *UCI Machine Learning Repository*. `https://archive.ics.uci.edu/dataset/144/statlog+german+credit+data`.

Ferreira L (2018). "German credit risk." Last accessed 10.04.2024, `https://www.kaggle.com/datasets/kabure/german-credit-data-with-risk`.

`summary.Learner` *Summarizing mlr3 Learners*

Description

summary method for mlr3::Learner. The output can be tailored via the `control` argument, see summary_control.

Usage

```
## S3 method for class 'Learner'
summary(object, resample_result = NULL, control = summary_control(), ...)

## S3 method for class 'GraphLearner'
summary(object, resample_result = NULL, control = summary_control(), ...)

## S3 method for class 'summary.Learner'
print(x, digits = NULL, n_important = NULL, hide = NULL, ...)
```

Arguments

`object`	(mlr3::Learner) trained model of class `Learner`.
`resample_result`	(mlr3::ResampleResult) outcome of `resample`. If NULL (default), no residuals, performances, etc. are derived.
`control`	(`summary_control`) a list with control parameters, see `summary_control`.
`...`	(any) further arguments passed to or from other methods.
`x`	(`summary.Learner`) an object of class "summary.Learner", usually a result of a call to `summary.Learner`.

`digits` (numeric(1))
the number of digits to use when printing.

`n_important` (numeric(1))
number of important variables to be displayed. If NULL, `x$control$n_important` is used.

`hide` (character)
Names of paragraphs which should not be part of the summary. Possible values are "general", "residuals", "performance", "complexity", "fairness", "importance", "effect". If NULL, no paragraph is hided.

Details

This function can be parallelized with the **future** package. One job is one resampling iteration, and all jobs are sent to an apply function from **future.apply** in a single batch. To select a parallel backend, use `future::plan()`.

Value

summary.Learner returns an object of class "summary.Learner", a list with the following entries.

- task_type: The type of task, either `classif` (classification) or `regr` (regression).
- target_name: The name of the target variable.
- feature_names: The names of the features.
- classes: The classes of the target variable. NULL if regression task.
- resample_info: Information on the resample objects, strategy type and hyperparameters.
- residuals: Vector of hold-out residuals over the resampling iterations of `resample_result`. For regression models, residuals are the difference between true and predicted outcome. For classifiers with probabilities, the residuals are the difference between predicted probabilities and a one-hot-encoding of the true class. For hard-label classifier, a `confusion_matrix` is shown instead of `residuals`.
- confusion_matrix: Confusion matrix of predicted vs. true classes. Alternative to `residuals`, in case of hard-label classification.
- performance: Vector of aggregated performance measures over the iterations of `resample_result`. The arrows display whether lower or higher values are better. (micro/macro) displays whether it is a micro or macro measure. For macro aggregation, measures are computed for each iteration separately before averaging. For micro aggregation, measures are computed across all iterations. See Bischl et al. (2024), for details.
- performance_sd: Vector of standard deviations of performance measures over the iterations of `resample_result`. The arrows display whether lower or higher values are better. (micro/macro) displays whether it is a micro or macro measure.
- fairness: Vector of aggregated fairness measures over the iterations of `resample_result`. The arrows display whether lower or higher values are better. (micro/macro) displays whether it is a micro or macro measure.
- fairness_sd: Vector of standard deviations of fairness measures over the iterations of `resample_result`. The arrows display whether lower or higher values are better. (micro/macro) displays whether it is a micro or macro measure (see details above).

- importances: List of `data.table` that display the feature importances per importance measure. Given are the means and standard deviations over the resampling iterations of `resample_result`. Higher average values display higher importance of a feature.
- effects: List of `data.tables` that display the feature effects per effect method. Given are the mean effects over the resampling iterations of `resample_result` for a maximum of 5 grid points. For binary classifiers, effects are only displayed for the positively-labeled class. For multi-class, effect plots are displayed separately for each class. For categorical features, the factor levels of the feature determine the ordering of the bars.
- complexity: List of vectors that display the complexity values per complexity measure for each resampling iteration.
- control: summary_control used as an input for `summary.Learner`.

For details on the performance measures, complexity measures, feature importance and feature effect methods, see summary_control.

References

Bischl, Bernd, Sonabend, Raphael, Kotthoff, Lars, Lang, Michel (2024). *Applied machine learning using mlr3 in R*. Chapman and Hall/CRC. ISBN 9781003402848, https://mlr3book.mlr-org.com/.

Examples

```
if (require("mlr3")) {
  tsk_iris = tsk("iris")
  lrn_rpart =  lrn("classif.rpart", predict_type = "prob")
  lrn_rpart$train(task = tsk_iris)
  rsmp_cv3 = rsmp("cv", folds = 3L)
  rr = resample(tsk_iris, lrn_rpart, rsmp_cv3, store_model = TRUE)
  summary(lrn_rpart, rr)
}
```

`summary_control` *Control for Learner summaries*

Description

Various parameters that control aspects of `summary.Learner`.

Usage

```
summary_control(
  measures = NULL,
  complexity_measures = c("sparsity", "interaction_strength"),
  importance_measures = NULL,
  n_important = 15L,
  effect_measures = c("pdp", "ale"),
```

```
  fairness_measures = NULL,
  protected_attribute = NULL,
  hide = NULL,
  digits = max(3L, getOption("digits") - 3L)
)
```

Arguments

`measures` (mlr3::Measure | list of mlr3::Measure | NULL)
measure(s) to calculate performance on. If NULL (default), a set of selected measures are calculated (choice depends on Learner type (classif vs. regr)). See details below.

`complexity_measures` (character)
vector of complexity measures. Possible choices are "sparsity" (the number of used features) and "interaction_strength" (see Molnar et al. (2020)). Both are the default. See details below.

`importance_measures` (character()|NULL)
vector of importance measure names. Possible choices are "pfi.<loss>" (iml::FeatureImp), "pdp" (iml::FeatureEffects, see) and "shap" (fastshap::explain). Default of NULL results in "pfi.<loss>" and "pdp", where the <loss> depends on the Learner type (classif vs. regr). See details below.

`n_important` (numeric(1))
number of important variables to be displayed. Default is 15L.

`effect_measures` (character | NULL)
vector of effect method names. Possible choices are "pfi" and "ale" (see iml::FeatureEffects). Both are the default. See details below.

`fairness_measures` (mlr3fairness::MeasureFairness | list of mlr3fairness::MeasureFairness | NULL)
measure(s) to assess fairness. If NULL (default), a set of selected measures are calculated (choice depends on Learner type (classif vs. regr)). See details below.

`protected_attribute` (character(1))
name of the binary feature that is used as a protected attribute. If no `protected_attribute` is specified (and also no `pta` feature is available in the `mlr3::Task` for training the `mlr3::Learner`), no fairness metrics are computed.

`hide` (character)
names of paragraphs which should not be part of the summary. Possible values are "general", "residuals", "performance", "complexity", "fairness", "importance", "effect". If NULL, no paragraph is hided.

`digits` (numeric(1))
number of digits to use when printing.

Details

The following provides some details on the different choices of measures.

Performance The default `measures` depend on the type of task. Therefore, NULL is displayed as default and the measures will be initialized in `summary.Learner` with the help of `mlr3::msr`. The following provides an overview of these defaults:

- Regression: regr.rmse, regr.rsq, regr.mae, regr.medae
- Binary classification with probabilities: classif.auc, classif.fbeta, classif.bbrier, classif.mcc
- Binary classification with hard labels: classif.acc, classif.bacc, classif.fbeta, classif.mcc
- Multi-class classification with probabilities: classif.mauc_aunp, classif.mbrier

Complexity Currently only two `complexity_measures` are available, which are based on Molnar et al. (2020):

- `sparsity`: The number of used features, that have a non-zero effect on the prediction (evaluated by accumulated local effects (ale, Apley and Zhu (2020)). The measure can have values between 0 and the number of features.
- `interaction_strength`: The scaled approximation error between a main effect model (based on ale) and the prediction function. It can have values between 0 and 1, where 0 means no interaction and 1 only interaction, and no main effects. Interaction strength can only be measured for binary classification and regression models.

Importance The `importance_measures` are based on the `iml` and `fastshap` packages. Multiple measures are available:

- pdp: This cornensponds to importances based on the standard deviations in partial dependence plots (Friedmann (2001)), as proposed by Greenwell et al. (2018).
- pfi.<loss>: This corresponds to the permutation feature importance as implemented in iml::FeatureImp. Different loss functions are possible and rely on the task at hand.
- shap: This importance corresponds to the mean absolute Shapley values computed with fastshap::explain. Higher values display higher importance.

NULL is the default, corresponding to importance calculations based on pdp and pfi. Because the loss function for pfi relies on the task at hand, the importance measures are initialized in `summary`."pdp" and "pfi.ce" are the defaults for classification, "pdp" and "pfi.mse" for regression.

Effects The `effect_measures` are based on iml::FeatureEffects. Currently partial dependence plots (pdp) and accumulated local effects are available (ale). Ale has the advantage over pdp that it takes feature correlations into account but has a less natural interpretation than pdp. Therefore, both "pdp" and "ale" are the defaults.

Fairness The default `fairness_measures` depend on the type of task. Therefore, NULL is displayed as default and the measures will be initialized in `summary.Learner` based on mlr3fairness::mlr_measures_fairness. There is currently a mismatch between the naming convention of measures in `mlr3fairness` and the underlying measurements displayed. To avoid confusion, the id of the fairness measures were adapted. The following provides an overview of these defaults and adapted names:

- Binary classification: "fairness.dp" (demographic parity) based on "fairness.cv", "fairness.cuae" (conditional use accuracy equality) based on "fairness.pp", "fairness.eod" (equalized odds) based on "fairness.eod". Smaller values are better.
- Multi-class classification: "fairness.acc", the smallest absolute difference in accuracy between groups of the `protected_attribute`. Smaller values are better.

- Regression: "fairness.rmse" and "fairness.mae", the smallest absolute difference (see mlr3fairness::groupdiff_absdiff) in the either the root mean-squared error (rmse) or the mean absolute error (mae) between groups of the `protected_attribute`. Smaller values are better.

Value

list of class `summary_control`

References

Molnar, Christoph, Casalicchio, Giuseppe, Bischl, Bernd (2020). "Quantifying Model Complexity via Functional Decomposition for Better Post-hoc Interpretability." In *Communications in Computer and Information Science*, chapter 1, 193–204. Springer International Publishing.

Greenwell, M. B, Boehmke, C. B, McCarthy, J. A (2018). "A Simple and Effective Model-Based Variable Importance Measure." arXiv preprint. arXiv:1805.04755, `http://arxiv.org/abs/1805.04755`.

Apley, W. D, Zhu, Jingyu (2020). "Visualizing the Effects of Predictor Variables in Black Box Supervised Learning Models." *Journal of the Royal Statistical Society Series B: Statistical Methodology*, **82**(4), 1059-1086.

Friedman, H. J (2001). "Greedy Function Approximation: A Gradient Boosting Machine." *The Annals of Statistics*, **29**(5).

Index