Package: ASML (via r-universe)

February 20, 2025

Type Package

Title Algorithm Portfolio Selection with Machine Learning

Version 1.0.0

Description A wrapper for machine learning (ML) methods to select among a portfolio of algorithms based on the value of a key performance indicator (KPI). A number of features is used to adjust a model to predict the value of the KPI for each algorithm, then, for a new value of the features the KPI is estimated and the algorithm with the best one is chosen. To learn it can use the regression methods in 'caret' package or a custom function defined by the user. Several graphics available to analyze the results obtained. This library has been used in Ghaddar et al. (2023) <doi:10.1287/ijoc.2022.0090>).

License GPL-3

Language en-US

Encoding UTF-8

LazyData true

Imports caret, ggplot2, dplyr, purrr, tibble, tidyr, reshape2, Polychrome, scales, rlang

Suggests snow

Depends R (>= 3.5.0)

RoxygenNote 7.3.2

NeedsCompilation no

Author Brais González-Rodríguez [aut, cre] (<https://orcid.org/0000-0001-5276-2320>), Ignacio Gómez-Casares [aut] (<https://orcid.org/0000-0003-0420-7319>), Beatriz Pateiro-López [aut] (<https://orcid.org/0000-0002-7714-1835>), Julio González-Díaz [aut] (<https://orcid.org/0000-0002-4667-4348>), María Caseiro-Arias [ctb], Antonio Fariña-Elorza [ctb], Manuel Timiraos-López [ctb]

Maintainer Brais González-Rodríguez

<brais.gonzalez.rodriguez@uvigo.gal>

Date/Publication 2025-02-19 14:40:05 UTC

Config/pak/sysreqs libicu-dev

Repository https://brais-gonzalez.r-universe.dev

RemoteUrl https://github.com/cran/ASML

RemoteRef HEAD

RemoteSha 443982853fa36ec87cfa7bffe3950d3bef30c7ba

Contents

`ASpredict` *Predicting the KPI value for the algorithms*

Description

For each algorithm, the output (KPI) is predicted using the models trained with `AStrain()`.

Usage

```
ASpredict(training_object, ...)
```

Arguments

training_object	list of class as_train.
...	other parameters.

Value

A data frame, result of the respective ASpredict method.

ASpredict.as_train *Predicting the KPI value for the algorithms*

Description

For each algorithm, the output (KPI) is predicted using the models traing with AStrain().

Usage

```
## S3 method for class 'as_train'
ASpredict(training_object, newdata = NULL, f = NULL, ...)
```

Arguments

training_object	list of class as_train.
newdata	dataframe with the new data to predict. If not present, predictions are computed using the training data.
f	function to use for the predictions. If NULL, caret's function will be used.
...	arguments passed to the predict function f when f is not NULL.

Details

The ASpredict() uses the prediction function from caret to compute (for each of the models trained) the predictions for the new data provided by the user. If the user used a custom function in AStrain() (given by parameter f), caret's default prediction function might not work, and the user might have to provide a custom function for ASpredict() as well. Additionally, this custom prediction function allows to pass additional arguments, something that caret's default prediction function does not. The object return by the train function used in AStrain() (caret's or a custom one) is the one passed to the custom f function defined by the user. This f function must return a vector with the predictions.

Value

A data frame with the predictions for each instance (rows), corresponding to each algorithm (columns). In case f is specified, some actions might be needed to get the predictions from the returned value.

Examples

```
data(branchingsmall)
data_object <- partition_and_normalize(branchingsmall$x, branchingsmall$y, test_size = 0.3,
family_column = 1, split_by_family = TRUE)
training <- AStrain(data_object, method = "glm")
predictions <- ASpredict(training, newdata = data_object$x.test)
qrf_q_predict <- function(modelFit, newdata, what = 0.5, submodels = NULL) {
  out <- predict(modelFit, newdata, what = what)
  if (is.matrix(out))
    out <- out[, 1]
  out
}
custom_predictions <- ASpredict(training, newdata = data_object$x.test, f = "qrf_q_predict",
what = 0.25)
```

AStrain *Training models for posterior selection of algorithms*

Description

For each algorithm (column) in the data, a model is trained to later predict the output (KPI) for that algorithm (using function `ASpredict()`).

Usage

```
AStrain(data_object, ...)
```

Arguments

`data_object`	an object.
`...`	other parameters.

Value

A list, result of the respective AStrain method.

AStrain.as_data *Training models for posterior selection of algorithms*

Description

For each algorithm (column) in the data, a model is trained to later predict the output (KPI) for that algorithm (using function `ASpredict()`).

Usage

```
## S3 method for class 'as_data'
AStrain(data_object, method = NULL, parallel = FALSE, f = NULL, ...)
```

Arguments

data_object	object of class as_data.
method	name of the model to be used. The user can choose from any of the models provided by caret. See http://topepo.github.io/caret/train-models-by-tag.html for more information about the models supported.
parallel	boolean to control whether to parallelise the training or not (paralellization is handled by library snow).
f	function we want to use to train the models. If NULL, caret's function will be used.
...	arguments passed to the caret train function.

Value

A list is returned of class as_train containing the trained models, one for each of the algorithms.

Examples

```
data(branchingsmall)
data_object <- partition_and_normalize(branchingsmall$x, branchingsmall$y, test_size = 0.3,
family_column = 1, split_by_family = TRUE)
training <- AStrain(data_object, method = "glm")
custom_function <- function(x, y) {
  glm.fit(x, y)
}
custom_training <- AStrain(data_object, f = "custom_function")
```

---

boxplots *Boxplots*

Description

Generates boxplots for an object.

Usage

```
boxplots(data_object, ...)
```

Arguments

data_object	an object.
...	other parameters.

Value

A ggplot object, result of the respective boxplots method.

`boxplots.as_data` *Boxplots*

Description

Represents a boxplot for each of the algorithms to compare their performance according to the response variable (KPI). When available, it also includes a box plot for the "ML" algorithm generated from the predictions.

Usage

```
## S3 method for class 'as_data'
boxplots(
  data_object,
  main = "Boxplot Comparison",
  labels = NULL,
  test = TRUE,
  predictions = NULL,
  by_families = FALSE,
  color_list = NULL,
  ml_color = NULL,
  ordered_option_names = NULL,
  xlab = "Strategy",
  ylab = "KPI",
  ...
)
```

Arguments

`data_object`	object of class `as_data`.
`main`	an overall title for the plot.
`labels`	character vector with the labels for each of the algorithms. If NULL, the y names of the `data_object` names will be used.
`test`	flag that indicates whether the function should use test data or training data.
`predictions`	a data frame with the predicted KPI for each algorithm (columns) and for each instance (rows). If NULL, the plot won't include a ML column.
`by_families`	boolean indicating whether the function should represent data by families or not. The family information must be included in the `data_object` parameter.
`color_list`	list with the colors for the plots. If NULL, or insufficient number of colors, the colors will be generated automatically.
`ml_color`	color por the ML boxplot. If NULL, it will be generated automatically.

ordered_option_names
: vector with the name of the columns of data_object y variable in the correct order.

xlab
: a label for the x axis.

ylab
: a label for the y axis.

...
: other parameters.

Value

A ggplot object representing the boxplots of instance-normalized KPI for each algorithm across instances.

Examples

```
data(branchingsmall)
data <- partition_and_normalize(branchingsmall$x, branchingsmall$y)
training <- AStrain(data, method = "glm")
predict_test <- ASpredict(training, newdata = data$x.test)
boxplots(data, predictions = predict_test)
```

---

branching *Branching point selection in Polynomial Optimization*

Description

Data from Ghaddar et al. (2023) used to select among several branching criteria for an RLT-based algorithm. Includes features for the instances and KPI values for the different branching criteria for executions lasting 1 hour.

Usage

```
branching
```

Format

A list with x (features) and y (KPIs) data.frames.

Source

Ghaddar, B., Gómez-Casares, I., González-Díaz, J., González-Rodríguez, B., Pateiro-López, B., & Rodríguez-Ballesteros, S. (2023). Learning for Spatial Branching: An Algorithm Selection Approach. INFORMS Journal on Computing.

branchingsmall *Branching point selection in Polynomial Optimization*

Description

Data from Ghaddar et al. (2023) used to select among several branching criteria for an RLT-based algorithm. Includes features for the instances and KPI values for the different branching criteria for executions lasting 10 minutes.

Usage

```
branchingsmall
```

Format

A list with x (features) and y (KPIs) data.frames.

Source

Ghaddar, B., Gómez-Casares, I., González-Díaz, J., González-Rodríguez, B., Pateiro-López, B., & Rodríguez-Ballesteros, S. (2023). Learning for Spatial Branching: An Algorithm Selection Approach. INFORMS Journal on Computing.

figure_comparison *Figure comparison*

Description

Generates figure comparison plot for an object.

Usage

```
figure_comparison(data_object, ...)
```

Arguments

data_object	an object
...	other parameters

Value

A ggplot object, result of the respective figure_comparison method.

`figure_comparison.as_data` *Figure Comparison*

Description

Represents a bar plot with the percentage of times each algorithm is selected by ML compared with the optimal selection (according to the response variable or KPI).

Usage

```
## S3 method for class 'as_data'
figure_comparison(
  data_object,
  ties = "different_data_points",
  main = "Option Comparison",
  labels = NULL,
  mllabel = NULL,
  test = TRUE,
  predictions,
  by_families = FALSE,
  stacked = TRUE,
  color_list = NULL,
  legend = TRUE,
  ordered_option_names = NULL,
  xlab = "Criteria",
  ylab = "Instances (%)",
  ...
)
```

Arguments

`data_object`	object of class `as_data`.
`ties`	How to deal with ties. Must be one of: • "different_data_points": Tied algorithms in the optimal selection are all counted as different data points (increasing the total number of x values and therefore giving all of the tied algorithms the same weight). • "ml_if_optimal": For tied algorithms, the one selected by ML is chosen if it corresponds to the optimal one. Otherwise, the same as in option `different_data_points` is done. • "ml_selection": For tied algorithms, the one prefered by the ML is chosen.
`main`	an overall title for the plot.
`labels`	character vector with the labels for each of the algorithms. If NULL, the y names of the `data_object` names will be used.

mllabel	character vector with the labels for the Optimal and ML bars. If NULL, default names will be used.
test	flag that indicates whether the function should use test data or training data.
predictions	a data frame with the predicted KPI for each algorithm (columns) and for each instance (rows).
by_families	boolean indicating whether the function should represent data by families or not. The family information must be included in the data_object parameter.
stacked	boolean to choose between bar plot and stacked bar plot.
color_list	list with the colors for the plots. If NULL, or insufficient number of colors, the colors will be generated automatically.
legend	boolean to activate or deactivate the legend in the plot.
ordered_option_names	vector with the name of the columns of data_object y variable in the correct order.
xlab	a label for the x axis.
ylab	a label for the y axis.
...	other parameters.

Value

A ggplot object representing the bar plot with the percentage of times each algorithm is selected by ML compared with the optimal selection (according to the response variable or KPI).

Examples

```
data(branchingsmall)
data <- partition_and_normalize(branchingsmall$x, branchingsmall$y)
training <- AStrain(data, method = "glm")
predict_test <- ASpredict(training, newdata = data$x.test)
figure_comparison(data, predictions = predict_test)
```

KPI_summary_table *KPI summary table*

Description

Generates a summary table with the values of the KPI.

Function that generates a summary table of the KPI values. Optimal is the value of the KPI when choosing the best option for each instance. It's the best that we could do with respect to that KPI. Best is the value of the KPI for the best option overall according to the KPI. ML is the value of the KPI choosing for each instance the option selected by the learning.

Usage

```
KPI_summary_table(data_object, ...)

## S3 method for class 'as_data'
KPI_summary_table(
  data_object,
  predictions = NULL,
  test = TRUE,
  normalized = FALSE,
  ...
)
```

Arguments

data_object	an object of class as_data.
...	other parameters.
predictions	a data frame with the predicted KPI for each algorithm (columns) and for each instance (rows). If NULL, the table won't include a ML column.
test	flag that indicates whether the function should use test data or training data.
normalized	whether to use the original values of the KPI or the normalized ones used for the learning.

Value

A table, result of the respective KPI_summary_table method.

A table with the statistics of the pace.

Examples

```
data(branchingsmall)
data_object <- partition_and_normalize(branchingsmall$x, branchingsmall$y, test_size = 0.3,
family_column = 1, split_by_family = TRUE)
training <- AStrain(data_object, method = "glm")
predictions <- ASpredict(training, newdata = data_object$x.test)
KPI_summary_table(data_object, predictions = predictions)
```

KPI_table *KPI table*

Description

Generates a table with the values of the KPI.

Function that generates a table with the values of the KPI.

Usage

```
KPI_table(data_object, ...)

## S3 method for class 'as_data'
KPI_table(data_object, predictions = NULL, test = TRUE, ...)
```

Arguments

data_object	an object of class as_data.
...	other parameters.
predictions	a data frame with the predicted KPI for each algorithm (columns) and for each instance (rows). If NULL, the table won't include a ML column.
test	flag that indicates whether the function should use test data or training data.

Value

A table, result of the respective KPI_table method.

A table with the statistics of the pace.

Examples

```
data(branchingsmall)
data_object <- partition_and_normalize(branchingsmall$x, branchingsmall$y, test_size = 0.3,
family_column = 1, split_by_family = TRUE)
training <- AStrain(data_object, method = "glm")
predictions <- ASpredict(training, newdata = data_object$x.test)
KPI_table(data_object, predictions = predictions)
```

---

ml *Machine learning process*

Description

Function that proceses input data, trains the machine learning models, makes a prediction and plots the results.

Usage

```
ml(
  x,
  y,
  x.test = NULL,
  y.test = NULL,
  family_column = NULL,
  split_by_family = FALSE,
  predict = TRUE,
```

```
  test_size = 0.25,
  better_smaller = TRUE,
  method = "ranger",
  test = TRUE,
  color_list = NULL
)
```

Arguments

x	dataframe with the instances (rows) and its features (columns). It may also include a column with the family data.
y	dataframe with the instances (rows) and the corresponding output (KPI) for each algorithm (columns).
x.test	dataframe with the test features. It may also include a column with the family data. If NULL, the algorithm will split x into training and test sets.
y.test	dataframe with the test outputs. If NULL, the algorithm will split y into training and test sets.
family_column	column number of x where each instance family is indicated. If given, aditional options for the training and set test splitting and the graphics are enabled.
split_by_family	boolean indicating if we want to split sets keeping family proportions in case x.test and y.test are NULL. This option requires that option family_column is different from NULL
predict	boolean indicating if predictions will be made or not. If FALSE plots will use training data only and no ML column will be displayed.
test_size	float with the segmentation proportion for the test dataframe. It must be a value between 0 and 1.
better_smaller	boolean that indicates wether the output (KPI) is better if smaller (TRUE) or larger (FALSE).
method	name of the model to be used. The user can choose from any of the models provided by caret. See http://topepo.github.io/caret/train-models-by-tag.html for more information about the models supported.
test	boolean indicating whether the predictions will be made with the test set or the training set.
color_list	list with the colors for the plots. If NULL or insufficient number of colors, the colors will be generated automatically.

Value

A list with the data and plots generated, including:

- data_obj An as_data object with the processed data from partition_and_normalize() function.
- training An as_train object with the trainings from the AStrain() function.
- predictions A data frame with the predictions from the ASpredict() function, if the predict param is TRUE.

- table A table with the summary of the output data.
- boxplot, ranking_plot, figure_comparison, optml_figure_comparison and optmlall_figure_comparison with the corresponding plots.

Examples

```
data(branchingsmall)
machine_learning <- ml(branchingsmall$x, branchingsmall$y, test_size = 0.3,
family_column = 1, split_by_family = TRUE, method = "glm")
```

---

partition_and_normalize

Partition and Normalize

Description

Function that processes the input data splitting it into training and test sets and normalizes the outputs depending on the best instance performance. The user can bypass the partition into training and test set by passing the parameters x.test and y.test.

Usage

```
partition_and_normalize(
  x,
  y,
  x.test = NULL,
  y.test = NULL,
  family_column = NULL,
  split_by_family = FALSE,
  test_size = 0.3,
  better_smaller = TRUE
)
```

Arguments

x	dataframe with the instances (rows) and its features (columns). It may also include a column with the family data.
y	dataframe with the instances (rows) and the corresponding output (KPI) for each algorithm (columns).
x.test	dataframe with the test features. It may also include a column with the family data. If NULL the algorithm will split x into training and test sets.
y.test	dataframe with the test outputs. If NULL the algorithm will y into training and test sets.
family_column	column number of x where each instance family is indicated. If given, aditional options for the training and set test splitting and the graphics are enabled.

`split_by_family`
: boolean indicating if we want to split sets keeping family proportions in case x.test and y.test are NULL. This option requires that option `family_column` is different from NULL.

`test_size`
: float with the segmentation proportion for the test dataframe. It must be a value between 0 and 1. Only needed when `x.test` and `y.test` are NULL.

`better_smaller`
: boolean that indicates wether the output (KPI) is better if smaller (TRUE) or larger (FALSE).

Value

A list is returned of class `as_data` containing:

- `x.train` A data frame with the training features.
- `y.train` A data frame with the training output.
- `x.test` A data frame with the test features.
- `y.test` A data frame with the test output.
- `y.train.original` A vector with the original training output (without normalizing).
- `y.test.original` A vector with the original test output (without normalizing).
- `families.train` A data frame with the families of the training data.
- `families.test` A data frame with the families of the test data.

Examples

```
data(branching)
data_obj <- partition_and_normalize(branching$x, branching$y, test_size = 0.3,
family_column = 1, split_by_family = TRUE)
```

---

plot.as_data *Plot*

Description

For an object of class `as_data`, function that makes several plots, including the following: a boxplot, a ranking plot and comparisons between the different options.

Usage

```
## S3 method for class 'as_data'
plot(
  x,
  labels = NULL,
  test = TRUE,
  predictions = NULL,
```

```
  by_families = FALSE,
  stacked = TRUE,
  legend = TRUE,
  color_list = NULL,
  ml_color = NULL,
  path = NULL,
  ...
)
```

Arguments

x	object of class as_data.
labels	character vector with the labels for each of the algorithms. If NULL, the y names of the data_object names will be used.
test	flag that indicates whether the function should use test data or training data.
predictions	a data frame with the predicted KPI for each algorithm (columns) and for each instance (rows). If NULL, the plot won't include a ML column.
by_families	boolean indicating whether the function should represent data by families or not. The family information must be included in the data_object parameter.
stacked	boolean to choose between bar plot and stacked bar plot.
legend	boolean to activate or deactivate the legend in the plot.
color_list	list with the colors for the plots. If NULL, or insufficient number of colors, the colors will be generated automatically.
ml_color	color por the ML boxplot. If NULL, it will be generated automatically.
path	path where plots will be saved. If NULL they won't be saved.
...	other parameters.

Value

A list with boxplot, ranking, fig_comp, optml_fig_comp and optmlall_fig_comp plots.

Examples

```
data(branchingsmall)
data <- partition_and_normalize(branchingsmall$x, branchingsmall$y)
training <- AStrain(data, method = "glm")
predict_test <- ASpredict(training, newdata = data$x.test)
plot(data, predictions = predict_test)
```

ranking *Ranking*

Description

Generates ranking plot for an object.

Usage

```
ranking(data_object, ...)
```

Arguments

data_object	an object
...	other parameters

Value

A ggplot object, result of the respective ranking method.

ranking.as_data *Ranking Plot*

Description

After ranking the algorithms for each instance, represents for each of the algorithms, a bar with the percentage of times it was in each of the ranking positions. The number inside is the mean value of the normalized response variable (KPI) for the problems for which the algorithm was in that ranking position. The option predictions allows to control if the "ML" algorithm is added to the plot.

Usage

```
## S3 method for class 'as_data'
ranking(
  data_object,
  main = "Ranking",
  labels = NULL,
  test = TRUE,
  predictions = NULL,
  by_families = FALSE,
  ordered_option_names = NULL,
  xlab = "",
  ylab = "",
  ...
)
```

Arguments

data_object	object of class as_data.
main	an overall title for the plot.
labels	character vector with the labels for each of the algorithms. If NULL, the y names of the data_object names will be used.
test	flag that indicates whether the function should use test data or training data.
predictions	a data frame with the predicted KPI for each algorithm (columns) and for each instance (rows). If NULL, the plot won't include a ML column.
by_families	boolean indicating whether the function should represent data by families or not. The family information must be included in the data_object parameter.
ordered_option_names	vector with the name of the columns of data_object y variable in the correct order.
xlab	a label for the x axis.
ylab	a label for the y axis.
...	other parameters.

Value

A ggplot object representing the ranking of algorithms based on the instance-normalized KPI.

Examples

```
data(branchingsmall)
data <- partition_and_normalize(branchingsmall$x, branchingsmall$y)
training <- AStrain(data, method = "glm")
predict_test <- ASpredict(training, newdata = data$x.test)
ranking(data, predictions = predict_test)
```

Index