

Package: ppsr (via r-universe)

September 15, 2024

Type Package

Title Predictive Power Score

Version 0.0.5

Description The Predictive Power Score (PPS) is an asymmetric, data-type-agnostic score that can detect linear or non-linear relationships between two variables. The score ranges from 0 (no predictive power) to 1 (perfect predictive power). PPS can be useful for data exploration purposes, in the same way correlation analysis is. For more information on PPS, see <https://github.com/paulvanderlaken/pps>.

License GPL (>= 3)

Encoding UTF-8

LazyData true

Suggests testthat (>= 2.0.0)

Config/testthat/edition 3

Config/testthat/parallel true

RoxygenNote 7.2.3

Imports ggplot2 (>= 3.3.3), parsnip (>= 0.1.5), rpart (>= 4.1.15), withr (>= 2.4.1), gridExtra (>= 2.3)

Repository <https://paulvanderlaken.r-universe.dev>

RemoteUrl <https://github.com/paulvanderlaken/pps>

RemoteRef HEAD

RemoteSha 8245f33568ff8854545a0e290b7d59558a9a413f

Contents

available_algorithms	2
available_evaluation_metrics	2
normalize_score	3
pps	3
score	4

score_correlations	5
score_df	6
score_matrix	7
score_model	7
score_naive	8
score_predictors	9
visualize_both	10
visualize_correlations	11
visualize_pps	12

Index	13
--------------	-----------

available_algorithms *Lists all algorithms currently supported*

Description

Lists all algorithms currently supported

Usage

```
available_algorithms()
```

Value

a list of all available parsnip engines

Examples

```
available_algorithms()
```

available_evaluation_metrics
Lists all evaluation metrics currently supported

Description

Lists all evaluation metrics currently supported

Usage

```
available_evaluation_metrics()
```

Value

a list of all available evaluation metrics and their implementation in functional form

Examples

```
available_evaluation_metrics()
```

normalize_score	<i>Normalizes the original score compared to a naive baseline score The calculation that's being performed depends on the type of model</i>
-----------------	---

Description

Normalizes the original score compared to a naive baseline score The calculation that's being performed depends on the type of model

Usage

```
normalize_score(baseline_score, model_score, type)
```

Arguments

baseline_score	float, the evaluation metric score for a naive baseline (model)
model_score	float, the evaluation metric score for a statistical model
type	character, type of model

Value

numeric vector of length one, normalized score

ppsr	<i>ppsr: An R implementation of the Predictive Power Score (PPS)</i>
------	--

Description

The PPS is an asymmetric, data-type-agnostic score that can detect linear or non-linear relationships between two columns. The score ranges from 0 (no predictive power) to 1 (perfect predictive power). It can be used as an alternative to the correlation (matrix).

score	<i>Calculate predictive power score for x on y</i>
-------	--

Description

Calculate predictive power score for x on y

Usage

```
score(
  df,
  x,
  y,
  algorithm = "tree",
  metrics = list(regression = "MAE", classification = "F1_weighted"),
  cv_folds = 5,
  seed = 1,
  verbose = TRUE
)
```

Arguments

df	data.frame containing columns for x and y
x	string, column name of predictor variable
y	string, column name of target variable
algorithm	string, see available_algorithms()
metrics	named list of eval_* functions used for regression and classification problems, see available_evaluation_metrics()
cv_folds	float, number of cross-validation folds
seed	float, seed to ensure reproducibility/stability
verbose	boolean, whether to print notifications

Value

a named list, potentially containing

- x** the name of the predictor variable
- y** the name of the target variable
- result_type** text showing how to interpret the resulting score
- pps** the predictive power score
- metric** the evaluation metric used to compute the PPS
- baseline_score** the score of a naive model on the evaluation metric
- model_score** the score of the predictive model on the evaluation metric

- cv_folds** how many cross-validation folds were used
- seed** the seed that was set
- algorithm** text showing what algorithm was used
- model_type** text showing whether classification or regression was used

Examples

```
score(iris, x = 'Petal.Length', y = 'Species')
```

score_correlations	<i>Calculate correlation coefficients for whole dataframe</i>
--------------------	---

Description

Calculate correlation coefficients for whole dataframe

Usage

```
score_correlations(df, ...)
```

Arguments

df	data.frame containing columns for x and y
...	arguments to pass to <code>stats::cor()</code>

Value

a data.frame with x-y correlation coefficients

Examples

```
score_correlations(iris)
```

score_df	<i>Calculate predictive power scores for whole dataframe Iterates through the columns of the dataframe, calculating the predictive power score for every possible combination of x and y.</i>
----------	---

Description

Calculate predictive power scores for whole dataframe Iterates through the columns of the dataframe, calculating the predictive power score for every possible combination of x and y.

Usage

```
score_df(df, ..., do_parallel = FALSE, n_cores = -1)
```

Arguments

df	data.frame containing columns for x and y
...	any arguments passed to <code>score</code>
do_parallel	bool, whether to perform <code>score</code> calls in parallel
n_cores	numeric, number of cores to use, defaults to maximum minus 1

Value

a data.frame containing

- x** the name of the predictor variable
- y** the name of the target variable
- result_type** text showing how to interpret the resulting score
- pps** the predictive power score
- metric** the evaluation metric used to compute the PPS
- baseline_score** the score of a naive model on the evaluation metric
- model_score** the score of the predictive model on the evaluation metric
- cv_folds** how many cross-validation folds were used
- seed** the seed that was set
- algorithm** text showing what algorithm was used
- model_type** text showing whether classification or regression was used

Examples

```
score_df(iris)
score_df(mtcars, do_parallel = TRUE, n_cores = 2)
```

score_matrix	<i>Calculate predictive power score matrix Iterates through the columns of the dataset, calculating the predictive power score for every possible combination of x and y.</i>
--------------	---

Description

Note that the targets are on the rows, and the features on the columns.

Usage

```
score_matrix(df, ...)
```

Arguments

df	data.frame containing columns for x and y
...	any arguments passed to <code>score_df</code> , some of which will be passed on to <code>score</code>

Value

a matrix of numeric values, representing predictive power scores

Examples

```
score_matrix(iris)
score_matrix(mtcars, do_parallel = TRUE, n_cores=2)
```

score_model	<i>Calculates out-of-sample model performance of a statistical model</i>
-------------	--

Description

Calculates out-of-sample model performance of a statistical model

Usage

```
score_model(train, test, model, x, y, metric)
```

Arguments

train	df, training data, containing variable y
test	df, test data, containing variable y
model	parsnip model object, with mode preset
x	character, column name of predictor variable
y	character, column name of target variable
metric	character, name of evaluation metric being used, see <code>available_evaluation_metrics()</code>

Value

numeric vector of length one, evaluation score for predictions using naive model

score_naive	<i>Calculate out-of-sample model performance of naive baseline model The calculation that's being performed depends on the type of model For regression models, the mean is used as prediction For classification, a model predicting random values and a model predicting modal values are used and the best model is taken as baseline score</i>
-------------	--

Description

Calculate out-of-sample model performance of naive baseline model The calculation that's being performed depends on the type of model For regression models, the mean is used as prediction For classification, a model predicting random values and a model predicting modal values are used and the best model is taken as baseline score

Usage

```
score_naive(train, test, x, y, type, metric)
```

Arguments

train	df, training data, containing variable y
test	df, test data, containing variable y
x	character, column name of predictor variable
y	character, column name of target variable
type	character, type of model
metric	character, evaluation metric being used

Value

numeric vector of length one, evaluation score for predictions using naive model

score_predictors	<i>Calculate predictive power scores for y</i> <i>Calculates the predictive power scores for the specified y variable using every column in the dataset as x, including itself.</i>
------------------	---

Description

Calculate predictive power scores for y *Calculates the predictive power scores for the specified y variable using every column in the dataset as x, including itself.*

Usage

```
score_predictors(df, y, ..., do_parallel = FALSE, n_cores = -1)
```

Arguments

df	data.frame containing columns for x and y
y	string, column name of target variable
...	any arguments passed to score
do_parallel	bool, whether to perform score calls in parallel
n_cores	numeric, number of cores to use, defaults to maximum minus 1

Value

a data.frame containing

- x** the name of the predictor variable
- y** the name of the target variable
- result_type** text showing how to interpret the resulting score
- pps** the predictive power score
- metric** the evaluation metric used to compute the PPS
- baseline_score** the score of a naive model on the evaluation metric
- model_score** the score of the predictive model on the evaluation metric
- cv_folds** how many cross-validation folds were used
- seed** the seed that was set
- algorithm** text showing what algorithm was used
- model_type** text showing whether classification or regression was used

Examples

```
score_predictors(df = iris, y = 'Species')
score_predictors(df = mtcars, y = 'mpg', do_parallel = TRUE, n_cores = 2)
```

`visualize_both`*Visualize the PPS & correlation matrices*

Description

Visualize the PPS & correlation matrices

Usage

```
visualize_both(  
  df,  
  color_value_positive = "#08306B",  
  color_value_negative = "#8b0000",  
  color_text = "#FFFFFF",  
  include_missings = TRUE,  
  nrow = 1,  
  ...  
)
```

Arguments

<code>df</code>	data.frame containing columns for x and y
<code>color_value_positive</code>	color used for upper limit of gradient (high positive correlation)
<code>color_value_negative</code>	color used for lower limit of gradient (high negative correlation)
<code>color_text</code>	string, hex value or color name used for text, best to pick high contrast with <code>color_value_high</code>
<code>include_missings</code>	bool, whether to include the variables without correlation values in the plot
<code>nrow</code>	numeric, number of rows, either 1 or 2
<code>...</code>	any arguments passed to <code>score</code>

Value

a grob object, a grid with two ggplot2 heatmap visualizations

Examples

```
visualize_both(iris)
```

```
visualize_both(mtcars, do_parallel = TRUE, n_cores = 2)
```

`visualize_correlations`*Visualize the correlation matrix*

Description

Visualize the correlation matrix

Usage

```
visualize_correlations(  
  df,  
  color_value_positive = "#08306B",  
  color_value_negative = "#8b0000",  
  color_text = "#FFFFFF",  
  include_missings = FALSE,  
  ...  
)
```

Arguments

<code>df</code>	data.frame containing columns for x and y
<code>color_value_positive</code>	color used for upper limit of gradient (high positive correlation)
<code>color_value_negative</code>	color used for lower limit of gradient (high negative correlation)
<code>color_text</code>	color used for text, best to pick high contrast with <code>color_value_high</code>
<code>include_missings</code>	bool, whether to include the variables without correlation values in the plot
<code>...</code>	arguments to pass to <code>stats::cor()</code>

Value

a ggplot object, a heatmap visualization

Examples

```
visualize_correlations(iris)
```

visualize_pps	<i>Visualize the Predictive Power scores of the entire dataframe, or given a target</i>
---------------	---

Description

If `y` is specified, `visualize_pps` returns a barplot of the PPS of every predictor on the specified target variable. If `y` is not specified, `visualize_pps` returns a heatmap visualization of the PPS for all X-Y combinations in a dataframe.

Usage

```
visualize_pps(
  df,
  y = NULL,
  color_value_high = "#08306B",
  color_value_low = "#FFFFFF",
  color_text = "#FFFFFF",
  include_target = TRUE,
  ...
)
```

Arguments

<code>df</code>	data.frame containing columns for x and y
<code>y</code>	string, column name of target variable, can be left NULL to visualize all X-Y PPS
<code>color_value_high</code>	string, hex value or color name used for upper limit of PPS gradient (high PPS)
<code>color_value_low</code>	string, hex value or color name used for lower limit of PPS gradient (low PPS)
<code>color_text</code>	string, hex value or color name used for text, best to pick high contrast with <code>color_value_high</code>
<code>include_target</code>	boolean, whether to include the target variable in the barplot
<code>...</code>	any arguments passed to <code>score</code>

Value

a ggplot object, a vertical barplot or heatmap visualization

Examples

```
visualize_pps(iris, y = 'Species')

visualize_pps(iris)

visualize_pps(mtcars, do_parallel = TRUE, n_cores = 2)
```

Index

[available_algorithms](#), [2](#)
[available_evaluation_metrics](#), [2](#)

[normalize_score](#), [3](#)

[ppsr](#), [3](#)
[ppsr-package \(ppsr\)](#), [3](#)

[score](#), [4](#), [6](#), [7](#), [9](#), [10](#), [12](#)
[score_correlations](#), [5](#)
[score_df](#), [6](#), [7](#)
[score_matrix](#), [7](#)
[score_model](#), [7](#)
[score_naive](#), [8](#)
[score_predictors](#), [9](#)

[visualize_both](#), [10](#)
[visualize_correlations](#), [11](#)
[visualize_pps](#), [12](#)