

Package ‘theftdlc’

March 14, 2024

Type Package

Title Analyse and Interpret Time Series Features

Version 0.1.0

Date 2024-03-14

Maintainer Trent Henderson <then6675@uni.sydney.edu.au>

Description Provides a suite of functions for analysing, interpreting, and visualising time-series features calculated from different feature sets from the 'theft' package. Implements statistical learning methodologies described in Henderson, T., Bryant, A., and Fulcher, B. (2023) <[arXiv:2303.17809](https://arxiv.org/abs/2303.17809)>.

BugReports <https://github.com/henderson trent/theftdlc/issues>

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 3.5.0), theft (>= 0.6.1)

Imports rlang, stats, tibble, dplyr, ggplot2, tidyr, purrr, reshape2, scales, broom, Rtsne, e1071, janitor, umap, MASS, mclust, normaliseR, correctR

Suggests lifecycle, cachem, bslib, knitr, markdown, rmarkdown, pkgdown, testthat

RoxygenNote 7.2.2

VignetteBuilder knitr

URL <https://henderson trent.github.io/theftdlc/>

NeedsCompilation no

Author Trent Henderson [cre, aut]

Repository CRAN

Date/Publication 2024-03-14 20:40:02 UTC

R topics documented:

classify	2
cluster	4
compare_features	5
filter_duplicates	7
filter_good_features	7
find_good_features	8
fit_models	8
get_rescale_vals	9
interval	10
make_title	11
plot.feature_calculations	12
plot.feature_projection	13
project	13
resample_data	15
rescale_zscore	16
select_stat_cols	16
stat_test	17
theftdlc	18

Index **19**

classify	<i>Fit classifiers using time-series features using a resample-based approach and get a fast understanding of performance</i>
----------	---

Description

Fit classifiers using time-series features using a resample-based approach and get a fast understanding of performance

Usage

```

classify(
  data,
  classifier = NULL,
  train_size = 0.75,
  n_resamples = 30,
  by_set = TRUE,
  use_null = FALSE,
  seed = 123
)

tsfeature_classifier(
  data,
  classifier = NULL,
  train_size = 0.75,

```

```

    n_resamples = 30,
    by_set = TRUE,
    use_null = FALSE,
    seed = 123
  )

```

Arguments

<code>data</code>	feature_calculations object containing the raw feature matrix produced by <code>theft::calculate_features</code>
<code>classifier</code>	function specifying the classifier to fit. Should be a function with 2 arguments: <code>formula</code> and <code>data</code> containing a classifier compatible with R's predict functionality. Please note that <code>classify</code> z-scores data prior to modelling using the train set's information so disabling default scaling if your function uses it is recommended. Defaults to NULL which means the following linear SVM is fit: <code>classifier = function(formula, data){mod <- e1071::svm(formula, data = data, kernel = "linear", scale = FALSE, probability = TRUE)}</code>
<code>train_size</code>	numeric denoting the proportion of samples to use in the training set. Defaults to 0.75
<code>n_resamples</code>	integer denoting the number of resamples to calculate. Defaults to 30
<code>by_set</code>	Boolean specifying whether to compute classifiers for each feature set. Defaults to TRUE. If FALSE, the function will instead find the best individually-performing features
<code>use_null</code>	Boolean whether to fit null models where class labels are shuffled in order to generate a null distribution that can be compared to performance on correct class labels. Defaults to FALSE
<code>seed</code>	integer to fix R's random number generator to ensure reproducibility. Defaults to 123

Value

list containing a named vector of train-test set sizes, and a data.frame of classification performance results

Author(s)

Trent Henderson

Examples

```

library(theft)

features <- theft::calculate_features(theft::simData,
  group_var = "process",
  feature_set = "catch22")

classifiers <- classify(features,

```

```
by_set = FALSE,
n_resamples = 3)
```

cluster

Perform cluster analysis of time series using their feature vectors

Description

Perform cluster analysis of time series using their feature vectors

Usage

```
cluster(
  data,
  norm_method = c("zScore", "Sigmoid", "RobustSigmoid", "MinMax"),
  unit_int = FALSE,
  clust_method = c("kmeans", "hclust", "mclust"),
  k = 2,
  features = NULL,
  na_removal = c("feature", "sample"),
  seed = 123,
  ...
)
```

Arguments

data	feature_calculations object containing the raw feature matrix produced by <code>theft::calculate_features</code>
norm_method	character denoting the rescaling/normalising method to apply. Can be one of "zScore", "Sigmoid", "RobustSigmoid", "MinMax", or "MaxAbs". Defaults to "zScore"
unit_int	Boolean whether to rescale into unit interval $[0, 1]$ after applying normalisation method. Defaults to FALSE
clust_method	character specifying the clustering algorithm to use. Can be one of "kmeans" for k-means clustering, "hclust" for hierarchical clustering, or "mclust" for Gaussian mixture model clustering. Defaults to "kMeans"
k	integer denoting the number of clusters to extract. Defaults to 2
features	character vector denoting the names of time-series features to use in the clustering algorithm. Defaults to NULL for no feature filtering and usage of the entire feature matrix
na_removal	character defining the way to deal with NAs produced during feature calculation. Can be one of "feature" or "sample". "feature" removes all features that produced any NAs in any sample, keeping the number of samples the same. "sample" omits all samples that produced at least one NA. Defaults to "feature"

seed integer to fix R's random number generator to ensure reproducibility. Defaults to 123

... arguments to be passed to stats::kmeans or stats::hclust, or mclust::Mclust depending on selection in clust_method

Value

object of class feature_cluster containing the clustering algorithm and a tidy version of clusters joined to the input dataset ready for further analysis

Author(s)

Trent Henderson

Examples

```
library(theft)

features <- theft::calculate_features(theft::simData,
  group_var = "process",
  feature_set = "catch22")

clusts <- cluster(features,
  k = 6)
```

compare_features *Conduct statistical testing on time-series feature classification performance to identify top features or compare entire sets*

Description

Conduct statistical testing on time-series feature classification performance to identify top features or compare entire sets

Usage

```
compare_features(
  data,
  metric = c("accuracy", "precision", "recall", "f1"),
  by_set = TRUE,
  hypothesis = c("null", "pairwise"),
  p_adj = c("none", "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr")
)
```

Arguments

data	list object containing the classification outputs produce by <code>tsfeature_classifier</code>
metric	character denoting the classification performance metric to use in statistical testing. Can be one of "accuracy", "precision", "recall", "f1". Defaults to "accuracy"
by_set	Boolean specifying whether you want to compare feature sets (if TRUE) or individual features (if FALSE). Defaults to TRUE but this is contingent on whether you computed by set or not in <code>tsfeature_classifier</code>
hypothesis	character denoting whether p-values should be calculated for each feature set or feature (depending on <code>by_set</code> argument) individually relative to the null if <code>use_null = TRUE</code> in <code>tsfeature_classifier</code> through "null", or whether pairwise comparisons between each set or feature should be conducted on main model fits only through "pairwise". Defaults to "null"
p_adj	character denoting the adjustment made to p-values for multiple comparisons. Should be a valid argument to <code>stats::p.adjust</code> . Defaults to "none" for no adjustment. "holm" is recommended as a starting point for adjustments

Value

data.frame containing the results

Author(s)

Trent Henderson

References

Henderson, T., Bryant, A. G., and Fulcher, B. D. Never a Dull Moment: Distributional Properties as a Baseline for Time-Series Classification. 27th Pacific-Asia Conference on Knowledge Discovery and Data Mining, (2023).

Examples

```
library(theft)

features <- theft::calculate_features(theft::simData,
  group_var = "process",
  feature_set = NULL,
  features = list("mean" = mean, "sd" = sd))

classifiers <- classify(features,
  by_set = FALSE,
  n_resamples = 3)

compare_features(classifiers,
  by_set = FALSE,
  hypothesis = "pairwise")
```

filter_duplicates	<i>Remove duplicate features that exist in multiple feature sets and retain a reproducible random selection of one of them</i>
-------------------	--

Description

Remove duplicate features that exist in multiple feature sets and retain a reproducible random selection of one of them

Usage

```
filter_duplicates(data, preference = NULL, seed = 123)
```

Arguments

data	feature_calculations object containing the raw feature matrix produced by calculate_features
preference	deprecated. Do not use
seed	integer denoting a fix for R's pseudo-random number generator to ensure selections are reproducible. Defaults to 123

Value

feature_calculations object containing filtered feature data

Author(s)

Trent Henderson

filter_good_features	<i>Filter resample data sets according to good feature list</i>
----------------------	---

Description

Filter resample data sets according to good feature list

Usage

```
filter_good_features(data, x, good_features)
```

Arguments

data	list of "Train" and "Test" data
x	integer denoting the resample index to operate on
good_features	character vector of good features to keep

Value

list of filtered train and test data

Author(s)

Trent Henderson

find_good_features	<i>Helper function to find features in both train and test set that are "good"</i>
--------------------	--

Description

Helper function to find features in both train and test set that are "good"

Usage

```
find_good_features(data, x)
```

Arguments

data	list of "Train" and "Test" data
x	integer denoting the resample index to operate on

Value

character vector of "good" feature names

Author(s)

Trent Henderson

fit_models	<i>Fit classification model and compute key metrics</i>
------------	---

Description

Fit classification model and compute key metrics

Usage

```
fit_models(data, iter_data, row_id, is_null_run = FALSE, classifier)
```


Arguments

data	list containing train and test sets
iter_data	data.frame containing the values to iterate over for seed and either feature name or set name
row_id	integer denoting the row ID for iter_data to filter to
is_null_run	Boolean whether the calculation is for a null model. Defaults to FALSE
classifier	function specifying the classifier to fit. Should be a function with 2 arguments: formula and data. Please note that tsfeature_classifier z-scores data prior to modelling using the train set's information so disabling default scaling if your function uses it is recommended.

Value

data.frame of classification results

Author(s)

Trent Henderson

get_rescale_vals	<i>Calculate central tendency and spread values for all numeric columns in a dataset</i>
------------------	--

Description

Calculate central tendency and spread values for all numeric columns in a dataset

Usage

```
get_rescale_vals(data)
```

Arguments

data	data.frame containing data to normalise
------	---

Value

list of central tendency and spread values

Author(s)

Trent Henderson

interval	<i>Calculate interval summaries with a measure of central tendency of classification results</i>
----------	--

Description

Calculate interval summaries with a measure of central tendency of classification results

Usage

```
interval(
  data,
  metric = c("accuracy", "precision", "recall", "f1"),
  by_set = TRUE,
  type = c("sd", "qt", "quantile"),
  interval = NULL,
  model_type = c("main", "null")
)

calculate_interval(
  data,
  metric = c("accuracy", "precision", "recall", "f1"),
  by_set = TRUE,
  type = c("sd", "qt", "quantile"),
  interval = NULL,
  model_type = c("main", "null")
)
```

Arguments

data	list object containing the classification outputs produce by <code>tsfeature_classifier</code>
metric	character denoting the classification performance metric to calculate intervals for. Can be one of "accuracy", "precision", "recall", "f1". Defaults to "accuracy"
by_set	Boolean specifying whether to compute intervals for each feature set. Defaults to TRUE. If FALSE, the function will instead calculate intervals for each feature
type	character denoting whether to calculate a +/- SD interval with "sd", confidence interval based off the t-distribution with "qt", or based on a quantile with "quantile". Defaults to "sd"
interval	numeric scalar denoting the width of the interval to calculate. Defaults to 1 if type = "sd" to produce a +/- 1 SD interval. Defaults to 0.95 if type = "qt" or type = "quantile" for a 95 per cent interval
model_type	character denoting whether to calculate intervals for main models with "main" or null models with "null" if the <code>use_null</code> argument when using <code>tsfeature_classifier</code> was <code>use_null = TRUE</code> . Defaults to "main"

Value

data.frame containing the results

Author(s)

Trent Henderson

Examples

```
library(theft)

features <- theft::calculate_features(theft::simData,
  group_var = "process",
  feature_set = NULL,
  features = list("mean" = mean, "sd" = sd))

classifiers <- classify(features,
  by_set = FALSE,
  n_resamples = 3)

interval(classifiers,
  by_set = FALSE,
  type = "sd",
  interval = 1)
```

make_title

Helper function for converting to title case

Description

Helper function for converting to title case

Usage

```
make_title(x)
```

Arguments

x character vector

Value

character vector

Author(s)

Trent Henderson

```
plot.feature_calculations
```

Produce a plot for a feature_calculations object

Description

Produce a plot for a feature_calculations object

Usage

```
## S3 method for class 'feature_calculations'
plot(
  x,
  type = c("quality", "matrix", "cor", "violin"),
  norm_method = c("z-score", "Sigmoid", "RobustSigmoid", "MinMax"),
  unit_int = FALSE,
  clust_method = c("average", "ward.D", "ward.D2", "single", "complete", "mcquitty",
    "median", "centroid"),
  cor_method = c("pearson", "spearman"),
  feature_names = NULL,
  ...
)
```

Arguments

x	feature_calculations object containing the raw feature matrix produced by <code>theft::calculate_features</code>
type	character specifying the type of plot to draw. Defaults to "quality"
norm_method	character specifying a rescaling/normalising method to apply if type = "matrix" or if type = "cor". Can be one of "z-score", "Sigmoid", "RobustSigmoid", or "MinMax". Defaults to "z-score"
unit_int	Boolean whether to rescale into unit interval $[0, 1]$ after applying normalisation method. Defaults to FALSE
clust_method	character specifying the hierarchical clustering method to use if type = "matrix" or if type = "cor". Defaults to "average"
cor_method	character specifying the correlation method to use if type = "cor". Defaults to "pearson"
feature_names	character vector denoting the name of the features to plot if type = "violin". Defaults to NULL
...	Arguments to be passed to <code>ggplot2::geom_bar</code> if type = "quality", <code>ggplot2::geom_raster</code> if type = "matrix", <code>ggplot2::geom_raster</code> if type = "cor", or <code>ggplot2::geom_point</code> if type = "violin"

Value

object of class `ggplot` that contains the graphic

Author(s)

Trent Henderson

plot.feature_projection

Produce a plot for a feature_projection object

Description

Produce a plot for a feature_projection object

Usage

```
## S3 method for class 'feature_projection'  
plot(x, show_covariance = TRUE, ...)
```

Arguments

x	feature_projection object containing the two-dimensional embedding calculated by project
show_covariance	Boolean specifying whether covariance ellipses should be shown on the plot. Defaults to TRUE
...	Arguments to be passed to methods

Value

object of class ggplot that contains the graphic

Author(s)

Trent Henderson

project

Project a feature matrix into a two-dimensional representation using PCA, MDS, t-SNE, or UMAP ready for plotting

Description

Project a feature matrix into a two-dimensional representation using PCA, MDS, t-SNE, or UMAP ready for plotting

Usage

```

project(
  data,
  norm_method = c("zScore", "Sigmoid", "RobustSigmoid", "MinMax"),
  unit_int = FALSE,
  low_dim_method = c("PCA", "tSNE", "ClassicalMDS", "KruskalMDS", "SammonMDS", "UMAP"),
  na_removal = c("feature", "sample"),
  seed = 123,
  ...
)

reduce_dims(
  data,
  norm_method = c("zScore", "Sigmoid", "RobustSigmoid", "MinMax"),
  unit_int = FALSE,
  low_dim_method = c("PCA", "tSNE", "ClassicalMDS", "KruskalMDS", "SammonMDS", "UMAP"),
  na_removal = c("feature", "sample"),
  seed = 123,
  ...
)

```

Arguments

<code>data</code>	feature_calculations object containing the raw feature matrix produced by <code>theft::calculate_features</code>
<code>norm_method</code>	character denoting the rescaling/normalising method to apply. Can be one of "zScore", "Sigmoid", "RobustSigmoid", "MinMax", or "MaxAbs". Defaults to "zScore"
<code>unit_int</code>	Boolean whether to rescale into unit interval $[0, 1]$ after applying normalisation method. Defaults to FALSE
<code>low_dim_method</code>	character specifying the low dimensional embedding method to use. Can be one of "PCA", "tSNE", "ClassicalMDS", "KruskalMDS", "SammonMDS", or "UMAP". Defaults to "PCA"
<code>na_removal</code>	character defining the way to deal with NAs produced during feature calculation. Can be one of "feature" or "sample". "feature" removes all features that produced any NAs in any sample, keeping the number of samples the same. "sample" omits all samples that produced at least one NA. Defaults to "feature"
<code>seed</code>	integer to fix R's random number generator to ensure reproducibility. Defaults to 123
<code>...</code>	arguments to be passed to <code>stats::prcomp</code> or <code>Rtsne::Rtsne</code> , <code>stats::cmdscale</code> , <code>MASS::isoMDS</code> , <code>MASS::sammon</code> , or <code>umap::umap</code> depending on selection in <code>low_dim_method</code>

Value

object of class `feature_project` which is a named list containing the `feature_calculations` data supplied to the function, the wide matrix of filtered data, a tidy `data.frame` of the projected

2-D data, and the model fit object

Author(s)

Trent Henderson

Examples

```
library(theft)

features <- theft::calculate_features(theft::simData,
  group_var = "process",
  feature_set = "catch22")

pca <- project(features,
  norm_method = "zScore",
  low_dim_method = "PCA")
```

resample_data

Helper function to create a resampled dataset

Description

Helper function to create a resampled dataset

Usage

```
resample_data(data, train_rows, test_rows, train_groups, test_groups, seed)
```

Arguments

data	data.frame containing time-series features
train_rows	integer denoting the number of cases in the train set
test_rows	integer denoting the number of cases in the test set
train_groups	data.frame containing proportions of each class in original train split
test_groups	data.frame containing proportions of each class in original test split
seed	integer denoting fixed value for R's pseudorandom number generator

Value

list containing new train and test data

Author(s)

Trent Henderson

rescale_zscore	<i>Calculate z-score for all columns in a dataset using train set central tendency and spread</i>
----------------	---

Description

Calculate z-score for all columns in a dataset using train set central tendency and spread

Usage

```
rescale_zscore(data, rescalers)
```

Arguments

data	data.frame containing data to normalise
rescalers	list containing central tendency and spread values for the train set

Value

data.frame of rescaled data

Author(s)

Trent Henderson

select_stat_cols	<i>Helper function to select only the relevant columns for statistical testing</i>
------------------	--

Description

Helper function to select only the relevant columns for statistical testing

Usage

```
select_stat_cols(data, by_set, metric, hypothesis)
```

Arguments

data	data.frame of classification accuracy results
by_set	Boolean specifying whether you want to compare feature sets (if TRUE) or individual features (if FALSE).
metric	character denoting the classification performance metric to use in statistical testing. Can be one of "accuracy", "precision", "recall", "f1". Defaults to "accuracy"

`hypothesis` character denoting whether p-values should be calculated for each feature set or feature (depending on `by_set` argument) individually relative to the null if `use_null = TRUE` in `tsfeature_classifier` through "null", or whether pairwise comparisons between each set or feature should be conducted on main model fits only through "pairwise".

Value

object of class `data.frame`

Author(s)

Trent Henderson

<code>stat_test</code>	<i>Calculate p-values for feature sets or features relative to an empirical null or each other using resampled t-tests</i>
------------------------	--

Description

Calculate p-values for feature sets or features relative to an empirical null or each other using resampled t-tests

Usage

```
stat_test(
  data,
  iter_data,
  row_id,
  by_set = FALSE,
  hypothesis,
  metric,
  train_test_sizes,
  n_resamples
)
```

Arguments

<code>data</code>	<code>data.frame</code> of raw classification accuracy results
<code>iter_data</code>	<code>data.frame</code> containing the values to iterate over for seed and either feature name or set name
<code>row_id</code>	integer denoting the row ID for <code>iter_data</code> to filter to
<code>by_set</code>	Boolean specifying whether you want to compare feature sets (if <code>TRUE</code>) or individual features (if <code>FALSE</code>).

hypothesis	character denoting whether p-values should be calculated for each feature set or feature (depending on <code>by_set</code> argument) individually relative to the null if <code>use_null = TRUE</code> in <code>tsfeature_classifier</code> through "null", or whether pairwise comparisons between each set or feature should be conducted on main model fits only through "pairwise".
metric	character denoting the classification performance metric to use in statistical testing. Can be one of "accuracy", "precision", "recall", "f1". Defaults to "accuracy"
train_test_sizes	integer vector containing the train and test set sample sizes
n_resamples	integer denoting the number of resamples that were calculated

Value

object of class `data.frame`

Author(s)

Trent Henderson

theftdlc

Analyse and Interpret Time Series Features

Description

Analyse and Interpret Time Series Features

Index

`calculate_interval (interval)`, 10
`classify`, 2
`cluster`, 4
`compare_features`, 5

`filter_duplicates`, 7
`filter_good_features`, 7
`find_good_features`, 8
`fit_models`, 8

`get_rescale_vals`, 9

`interval`, 10

`make_title`, 11

`plot.feature_calculations`, 12
`plot.feature_projection`, 13
`project`, 13

`reduce_dims (project)`, 13
`resample_data`, 15
`rescale_zscore`, 16

`select_stat_cols`, 16
`stat_test`, 17

`theftdlc`, 18
`theftdlc-package (theftdlc)`, 18
`tsfeature_classifier (classify)`, 2