Package 'densityratio'

May 19, 2025

Type Package

Title Distribution Comparison Through Density Ratio Estimation

Version 0.2.0

Description Fast, flexible and user-friendly tools for distribution comparison through direct density ratio estimation. The estimated density ratio can be used for covariate shift adjustment, outlier-detection, change-point detection, classification and evaluation of synthetic data quality. The package implements multiple non-parametric estimation techniques (unconstrained least-squares importance fitting, ulsif(), Kullback-Leibler importance estimation procedure, kliep(), spectral density ratio estimation, spectral(), kernel mean matching, kmm(), and least-squares hetero-distributional subspace search, lhss()). with automatic tuning of hyperparameters. Helper functions are available for two-sample testing and visualizing the density ratios. For an overview on density ratio estimation, see Sugiyama et al. (2012) <doi:10.1017/CBO9781139035613> for a general overview, and the help files for references on the specific estimation techniques.

License GPL (>= 3)

Encoding UTF-8

LazyData true

Imports osqp, Rcpp, pbapply, ggplot2

LinkingTo Rcpp, RcppArmadillo, RcppProgress

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

Config/testthat/parallel true

Depends R (>= 2.10)

RoxygenNote 7.3.2

VignetteBuilder knitr

URL `https://thomvolker.github.io/densityratio/`

BugReports `https://github.com/thomvolker/densityratio/issues`

NeedsCompilation yes

Author Thom Volker [aut, cre] (ORCID: <https://orcid.org/0000-0002-2408-7820>),
Carlos Gonzalez Poses [ctb],
Erik-Jan van Kesteren [ctb]

Maintainer Thom Volker <thombenjaminvolker@gmail.com>

Repository CRAN

Date/Publication 2025-05-19 13:30:05 UTC

Contents

`colon` *colon*

Description

Colon cancer data set from princeton, containing 2000 gene expressions from 22 colon tumor tissues and 40 non-tumor tissues. The data is collected by Alon et al. (1999) and can be obtained from here.

Format

A data.frame with 62 rows and 2001 columns (class variable and 2000 gene expressions).

`create_bivariate_plot` *Bivariate plot*

Description

Bivariate plot

Usage

```
create_bivariate_plot(data, ext, vars, logscale, show.sample)
```

Arguments

`data`	Data frame with the individual values and density ratio estimates
`ext`	Data frame with the density ratio estimates and sample indicator
`vars`	Character vector of variable names to be plotted.
`logscale`	Logical indicating whether the density ratio should be plotted in log scale. Defaults to TRUE.
`show.sample`	Logical indicating whether to give different shapes to observations, depending on the sample they come from (numerator or denominator). Defaults to FALSE.

Value

Bivariate plot

`create_univariate_plot` *Univariate plot*

Description

Scatterplot of individual values and density ratio estimates. Used internally in `create_univariate_plot()`

Usage

```
create_univariate_plot(data, ext, var, y_lab, sample.facet = TRUE)
```

Arguments

`data`	Data frame with the individual values and density ratio estimates
`ext`	Data frame with the density ratio estimates and sample indicator
`var`	Name of the variable to be plotted on the x-axis
`y_lab`	Name of the y-axis label, typically ("Density Ratio" or "Log Density Ratio")
`sample.facet`	Logical indicating whether to facet the plot by sample. Default is TRUE.

Value

A scatterplot of variable values and density ratio estimates.

`denominator_data` *denominator_data*

Description

Simulated data set (see data-raw/generate-data-densityratio.R) with five variables that are used in the examples.

Format

A data frame with 1000 rows and 5 columns:

x1 Categorical variable with three categories, 'A', 'B' and 'C'

x2 Categorical variable with two categories, 'G1' and 'G2'

x3 Continuous variable (normally distributed given x1 and x2)

x4 Continuous variable (normally distributed)

x5 Continuous variable (normally distributed)

`denominator_small` *denominator_small*

Description

Subset of the denominator_data with three variables and 50 observations

Format

A data frame with 100 rows and 3 columns:

x1 Continuous variable (normally distributed given x1 and x2)

x2 Continuous variable (normally distributed)

x3 Continuous variable (normally distributed)

`distance` *Create a Gram matrix with squared Euclidean distances between observations in the input matrix `X` and the input matrix `Y`*

Description

Create a Gram matrix with squared Euclidean distances between observations in the input matrix `X` and the input matrix `Y`

Arguments

`X`	A numeric input matrix
`Y`	A numeric input matrix with the same variables as `X`
`intercept`	Logical indicating whether an intercept should be added to the estimation procedure. In this case, the first column is an all-zero column (which will be transformed into an all-ones column in the kernel).

`dr.histogram` *A histogram of density ratio estimates*

Description

Creates a histogram of the density ratio estimates. Useful to understand the distribution of estimated density ratios in each sample, or compare it among samples. It is the default plotting method for density ratio objects.

Usage

```
dr.histogram(
  x,
  samples = "both",
  logscale = TRUE,
  binwidth = NULL,
  bins = NULL,
  tol = 0.01,
  ...
)

## S3 method for class 'ulsif'
plot(
  x,
  samples = "both",
  logscale = TRUE,
  binwidth = NULL,
  bins = NULL,
  tol = 0.01,
  ...
)

## S3 method for class 'kliep'
plot(
  x,
  samples = "both",
  logscale = TRUE,
  binwidth = NULL,
  bins = NULL,
  tol = 0.01,
  ...
)

## S3 method for class 'kmm'
plot(
  x,
  samples = "both",
```

```
  logscale = TRUE,
  binwidth = NULL,
  bins = NULL,
  tol = 0.01,
  ...
)

## S3 method for class 'spectral'
plot(
  x,
  samples = "both",
  logscale = TRUE,
  binwidth = NULL,
  bins = NULL,
  tol = 0.01,
  ...
)

## S3 method for class 'lhss'
plot(
  x,
  samples = "both",
  logscale = TRUE,
  binwidth = NULL,
  bins = NULL,
  tol = 0.01,
  ...
)

## S3 method for class 'naivedensityratio'
plot(
  x,
  samples = "both",
  logscale = TRUE,
  binwidth = NULL,
  bins = NULL,
  tol = 0.01,
  ...
)
```

Arguments

`x`	Density ratio object created with e.g., `kliep()`, `ulsif()`, or `naive()`
`samples`	Character string indicating whether to plot the 'numerator', 'denominator', or 'both' samples. Default is 'both'.
`logscale`	Logical indicating whether to plot the density ratio estimates on a log scale. Default is TRUE.
`binwidth`	Numeric indicating the width of the bins, passed on to `ggplot2`.

`bins`	Numeric indicating the number of bins. Overriden by binwidth, and passed on to `ggplot2`.
`tol`	Numeric indicating the tolerance: values below this value will be set to the tolerance value, for legibility of the plots
`...`	Additional arguments passed on to `predict()`.

Value

A histogram of density ratio estimates.

A histogram of density ratio estimates.

A histogram of density ratio estimates.

A histogram of density ratio estimates.

A histogram of density ratio estimates.

A histogram of density ratio estimates.

A histogram of density ratio estimates.

See Also

`ulsif` for example usage

`kliep` for example usage

`kmm` for example usage

`spectral` for example usage

`lhss` for example usage

`naive` for example usage

`insurance` *insurance*

Description

Insurance data that is openly available (e.g., on Kaggle).

Format

A data.frame with 1338 rows and 7 columns:

age Age of the insured (continuous)

sex Sex of the insured (binary)

bmi Body mass index of the insured (continuous)

children Number of children/dependents covered by the insurance (integer)

smoker Whether the insured is a smoker (binary)

region The region in which the insured lives (categorical)

charges The medical costs billed by the insurance (continuous)

`kernel_gaussian` *Create gaussian kernel gram matrix from distance matrix*

Description

Create gaussian kernel gram matrix from distance matrix

Arguments

`dist`	A numeric distance matrix
`sigma`	A scalar with the length-scale parameter

`kidiq` *kidiq*

Description

The kidiq data stems from the National Longitudinal Survey of Youth and is used in Gelman and Hill (2007). The data set contains 434 observations measured on five variables, and is obtained from https://github.com/jknowles/BDAexampleR.

Format

A data.frame with 434 rows and 5 columns

kid_score Child’s IQ score (continuous)

mom_hs Whether the mother obtained a high school degree (binary)

mom_iq Mother’s IQ score (continuous)

mom_work Whether the mother worked in the first three years of the child’s life (1: not in the first three years; 2: in the second or third year; 3: parttime in the first year; 4: fulltime in the first year)

mom_age Mother’s age (continuous)

`kliep` *Kullback-Leibler importance estimation procedure*

Description

Kullback-Leibler importance estimation procedure

Usage

```
kliep(
  df_numerator,
  df_denominator,
  scale = "numerator",
  nsigma = 10,
  sigma_quantile = NULL,
  sigma = NULL,
  ncenters = 200,
  centers = NULL,
  cv = TRUE,
  nfold = 5,
  epsilon = NULL,
  maxit = 5000,
  progressbar = TRUE
)
```

Arguments

`df_numerator`	`data.frame` with exclusively numeric variables with the numerator samples
`df_denominator`	`data.frame` with exclusively numeric variables with the denominator samples (must have the same variables as `df_denominator`)
`scale`	`"numerator"`, `"denominator"`, or `NULL`, indicating whether to standardize each numeric variable according to the numerator means and standard deviations, the denominator means and standard deviations, or apply no standardization at all.
`nsigma`	Integer indicating the number of sigma values (bandwidth parameter of the Gaussian kernel gram matrix) to use in cross-validation.
`sigma_quantile`	`NULL` or numeric vector with probabilities to calculate the quantiles of the distance matrix to obtain sigma values. If `NULL`, `nsigma` values between `0.25` and `0.75` are used.
`sigma`	`NULL` or a scalar value to determine the bandwidth of the Gaussian kernel gram matrix. If `NULL`, `nsigma` values between `0.25` and `0.75` are used.
`ncenters`	Maximum number of Gaussian centers in the kernel gram matrix. Defaults to all numerator samples.
`centers`	Option to specify the Gaussian samples manually.
`cv`	Logical indicating whether or not to do cross-validation

`nfold`	Number of cross-validation folds used in order to calculate the optimal `sigma` value (default is 5-fold cv).
`epsilon`	Numeric scalar or vector with the learning rate for the gradient-ascent procedure. If a vector, all values are used as the learning rate. By default, `10^{1:-5}` is used.
`maxit`	Maximum number of iterations for the optimization scheme.
`progressbar`	Logical indicating whether or not to display a progressbar.

Value

`kliep`-object, containing all information to calculate the density ratio using optimal sigma and optimal weights.

References

Sugiyama, M., Suzuki, T., Nakajima, S., Kashima, H., Von Bünau, P., & Kawanabe, M. (2008). Direct importance estimation for covariate shift adaptation. Annals of the Institute of Statistical Mathematics 60, 699-746. Doi: https://doi.org/10.1007/s10463-008-0197-x.

Examples

```
set.seed(123)
# Fit model
dr <- kliep(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
kliep(numerator_small, denominator_small,
      nsigma = 1, ncenters = 100, nfold = 10,
      epsilon = 10^{2:-5}, maxit = 500)
```

`kmm` *Kernel mean matching approach to density ratio estimation*

Description

Kernel mean matching approach to density ratio estimation

Usage

```
kmm(
  df_numerator,
  df_denominator,
  scale = "numerator",
  constrained = FALSE,
  nsigma = 10,
  sigma_quantile = NULL,
  sigma = NULL,
  ncenters = 200,
  centers = NULL,
  cv = TRUE,
  nfold = 5,
  parallel = FALSE,
  nthreads = NULL,
  progressbar = TRUE,
  osqp_settings = NULL,
  cluster = NULL
)
```

Arguments

`df_numerator`	`data.frame` with exclusively numeric variables with the numerator samples
`df_denominator`	`data.frame` with exclusively numeric variables with the denominator samples (must have the same variables as `df_denominator`)
`scale`	`"numerator"`, `"denominator"`, or `NULL`, indicating whether to standardize each numeric variable according to the numerator means and standard deviations, the denominator means and standard deviations, or apply no standardization at all.
`constrained`	`logical` equals `FALSE` to use unconstrained optimization, `TRUE` to use constrained optimization. Defaults to `FALSE`.
`nsigma`	Integer indicating the number of sigma values (bandwidth parameter of the Gaussian kernel gram matrix) to use in cross-validation.
`sigma_quantile`	`NULL` or numeric vector with probabilities to calculate the quantiles of the distance matrix to obtain sigma values. If `NULL`, `nsigma` values between `0.25` and `0.75` are used.
`sigma`	`NULL` or a scalar value to determine the bandwidth of the Gaussian kernel gram matrix. If `NULL`, `nsigma` values between `0.25` and `0.75` are used.
`ncenters`	Maximum number of Gaussian centers in the kernel gram matrix. Defaults to all numerator samples.
`centers`	Option to specify the Gaussian samples manually.
`cv`	Logical indicating whether or not to do cross-validation
`nfold`	Number of cross-validation folds used in order to calculate the optimal `sigma` value (default is 5-fold cv).
`parallel`	logical indicating whether to use parallel processing in the cross-validation scheme.

`nthreads`	`NULL` or integer indicating the number of threads to use for parallel processing. If parallel processing is enabled, it defaults to the number of available threads minus one.
`progressbar`	Logical indicating whether or not to display a progressbar.
`osqp_settings`	Optional: settings to pass to the `osqp` solver for constrained optimization.
`cluster`	Optional: a cluster object to use for parallel processing, see `parallel::makeCluster`.

Value

`kmm`-object, containing all information to calculate the density ratio using optimal sigma and optimal weights.

References

Huang, J., Smola, A. J., Gretton, A., Borgwardt, K. M., & Schölkopf, B. (2006). Correcting sample selection bias by unlabeled data. In Advances in Neural Information Processing Systems, edited by B. Schölkopf, J. Platt and T. Hoffman. Available from `https://proceedings.neurips.cc/paper/2006/hash/a2186aa7c086b46ad4e8bf81e2a3a19b-Abstract.html`.

Examples

```
set.seed(123)
# Fit model
dr <- kmm(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
kmm(numerator_small, denominator_small,
    nsigma = 5, ncenters = 100, nfold = 10,
    constrained = TRUE)
```

`lhss` *Least-squares heterodistributional subspace search*

Description

Least-squares heterodistributional subspace search

Usage

```
lhss(
  df_numerator,
  df_denominator,
  m = NULL,
  intercept = TRUE,
  scale = "numerator",
  nsigma = 10,
  sigma_quantile = NULL,
  sigma = NULL,
  nlambda = 10,
  lambda = NULL,
  ncenters = 200,
  centers = NULL,
  maxit = 200,
  progressbar = TRUE
)
```

Arguments

`df_numerator`	`data.frame` with exclusively numeric variables with the numerator samples
`df_denominator`	`data.frame` with exclusively numeric variables with the denominator samples (must have the same variables as `df_denominator`)
`m`	Scalar indicating the dimensionality of the reduced subspace
`intercept`	`logical` Indicating whether to include an intercept term in the model. Defaults to `TRUE`.
`scale`	`"numerator"`, `"denominator"`, or `NULL`, indicating whether to standardize each numeric variable according to the numerator means and standard deviations, the denominator means and standard deviations, or apply no standardization at all.
`nsigma`	Integer indicating the number of sigma values (bandwidth parameter of the Gaussian kernel gram matrix) to use in cross-validation.
`sigma_quantile`	`NULL` or numeric vector with probabilities to calculate the quantiles of the distance matrix to obtain sigma values. If `NULL`, `nsigma` values between `0.05` and `0.95` are used.
`sigma`	`NULL` or a scalar value to determine the bandwidth of the Gaussian kernel gram matrix. If `NULL`, `nsigma` values between `0.05` and `0.95` are used.
`nlambda`	Integer indicating the number of `lambda` values (regularization parameter), by default, `lambda` is set to `10^seq(3, -3, length.out = nlambda)`.
`lambda`	`NULL` or numeric vector indicating the lambda values to use in cross-validation
`ncenters`	Maximum number of Gaussian centers in the kernel gram matrix. Defaults to all numerator samples.
`centers`	Numeric matrix with the same variables as `nu` and `de` that are used as Gaussian centers in the kernel Gram matrix. By default, the matrix `nu` is used as the matrix with Gaussian centers.
`maxit`	Maximum number of iterations in the updating scheme.
`progressbar`	Logical indicating whether or not to display a progressbar.

Value

`lhss`-object, containing all information to calculate the density ratio using optimal sigma, optimal lambda and optimal weights.

References

Sugiyama, M., Yamada, M., Von Bünau, P., Suzuki, T., Kanamori, T. & Kawanabe, M. (2011). Direct density-ratio estimation with dimensionality reduction via least-squares hetero-distributional subspace search. Neural Networks, 24, 183-198. doi:10.1016/j.neunet.2010.10.005.

Examples

```
set.seed(123)
# Fit model
dr <- naive(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
naive(numerator_small, denominator_small, m=2, kernel="epanechnikov")
```

`naive` *Naive density ratio estimation*

Description

The naive approach creates separate kernel density estimates for the numerator and the denominator samples, and then evaluates their ratio for the denominator samples. For multivariate data, the density ratio is computed after a orthogonal linear transformation, such that the new variables can be treated as independent. To reduce the dimensionality of the PCA solution, one can set the number of components by setting the `m` parameter to an integer value smaller than the number of variables.

Usage

```
naive(
  df_numerator,
  df_denominator,
  m = NULL,
  bw = "SJ",
```

```
  kernel = "gaussian",
  n = 2L^11,
  ...
)
```

Arguments

`df_numerator`	`data.frame` with exclusively numeric variables with the numerator samples
`df_denominator`	`data.frame` with exclusively numeric variables with the denominator samples (must have the same variables as `df_denominator`)
`m`	`integer` Optional parameter to reduce the dimensionality of the data in multivariate density ratio estimation problems. If missing, the number of variables in the data is used. If set to an integer value smaller than the number of variables, the first `m` principal components are used to estimate the density ratio. If set to `NULL`, the square root of the number of variables is used (for consistency with other methods).
`bw`	the smoothing bandwidth to be used. See stats::density for more information.
`kernel`	the kernel to be used. See stats::density for more information.
`n`	`integer` the number of equally spaced points at which the density is to be estimated. When n > 512, it is rounded up to a power of 2 during the calculations (as fast Fourier transform is used) and the final result is interpolated by stats::approx. So it makes sense to specify n as a power' of two.
`...`	further arguments passed to stats::density

Value

`naivedensityratio` object

See Also

`stats::density()`

Examples

```
set.seed(123)
# Fit model
dr <- naive(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
```

```
# Fit model with custom parameters
naive(numerator_small, denominator_small, m=2, kernel="epanechnikov")
```

`numerator_data` *numerator_data*

Description

Simulated data set (see data-raw/generate-data-densityratio.R) with five variables that are used in the examples.

Format

A data frame with 1000 rows and 5 columns:

x1 Categorical variable with three categories, 'A', 'B' and 'C'

x2 Categorical variable with two categories, 'G1' and 'G2'

x3 Continuous variable (normally distributed given x1 and x2)

x4 Continuous variable (normally distributed given x3)

x5 Continuous variable (mixture of two normally distributed variables)

`numerator_small` *numerator_small*

Description

Subset of the numerator_data with three variables and 50 observations

Format

A data frame with 50 rows and 3 columns:

x1 Continuous variable (normally distributed given x1 and x2)

x2 Continuous variable (normally distributed given x3)

x3 Continuous variable (mixture of two normally distributed variables)

`permute` *Single permutation*

Description

Single permutation

Single permutation statistic of `ulsif` object

Single permutation statistic of `kliep` object

Single permutation statistic of `kmm` object

Single permutation statistic of `lhss` object

Single permutation statistic of `spectral` object

Single permutation statistic of `naivedensityratio` object

Usage

```
permute(object, ...)

## S3 method for class 'ulsif'
permute(object, stacked, nnu, nde, ...)

## S3 method for class 'kliep'
permute(object, stacked, nnu, nde, min_pred = sqrt(.Machine$double.eps), ...)

## S3 method for class 'kmm'
permute(object, stacked, nnu, nde, ...)

## S3 method for class 'lhss'
permute(object, stacked, nnu, nde, ...)

## S3 method for class 'spectral'
permute(object, stacked, nnu, nde, ...)

## S3 method for class 'naivedensityratio'
permute(object, stacked, nnu, nde, min_pred, max_pred, ...)
```

Arguments

`object`	`naivedensityratio` object
`...`	Additional arguments to pass through to specific permute functions.
`stacked`	`matrix` with stacked numerator and denominator samples
`nnu`	Scalar with numerator sample size
`nde`	Scalar with denominator sample size
`min_pred`	Minimum value of the predicted density ratio
`max_pred`	Maximum value of the predicted density ratio

Value

permutation statistic for a single permutation of the data

permutation statistic for a single permutation of the data

permutation statistic for a single permutation of the data

permutation statistic for a single permutation of the data

permutation statistic for a single permutation of the data

permutation statistic for a single permutation of the data

permutation statistic for a single permutation of the data

`plot_bivariate` *Densityratio in two-dimensional plot*

Description

Plots a scatterplot of two variables, with densityratio mapped to the colour scale.

Usage

```
plot_bivariate(
  x,
  vars = NULL,
  samples = "both",
  grid = FALSE,
  logscale = TRUE,
  show.sample = FALSE,
  tol = 0.01,
  ...
)
```

Arguments

`x`	Density ratio object created with e.g., `kliep()`, `ulsif()`, or `naive()`
`vars`	Character vector of variable names for which all pairwise bivariate plots are created
`samples`	Character string indicating whether to plot the 'numerator', 'denominator', or 'both' samples. Default is 'both'.
`grid`	Logical indicating whether output should be a list of individual plots ("individual"), or one facetted plot with all variables ("assembled"). Defaults to "individual".
`logscale`	Logical indicating whether to plot the density ratio estimates on a log scale. Default is `TRUE`.
`show.sample`	Logical indicating whether to give different shapes to observations, depending on the sample they come from (numerator or denominator). Defaults to `FALSE`.

`tol`	Numeric indicating the tolerance: values below this value will be set to the tolerance value, for legibility of the plots
`...`	Additional arguments passed to the predict() function.

Value

Bivariate scatter plots of all combinations of variables in vars.

Examples

```
set.seed(123)
# Fit model
dr <- ulsif(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
ulsif(numerator_small, denominator_small, sigma = 2, lambda = 2)
```

`plot_univariate` *Scatter plot of density ratios and individual variables*

Description

A scatter plot showing the relationship between estimated density ratios and individual variables.

Usage

```
plot_univariate(
  x,
  vars = NULL,
  samples = "both",
  logscale = TRUE,
  grid = FALSE,
  sample.facet = FALSE,
  nrow.panel = NULL,
  tol = 0.01,
  ...
)
```

Arguments

`x`	Density ratio object created with e.g., `kliep()`, `ulsif()`, or `naive()`
`vars`	Character vector of variable names to be plotted.
`samples`	Character string indicating whether to plot the 'numerator', 'denominator', or 'both' samples. Default is 'both'.
`logscale`	Logical indicating whether to plot the density ratio estimates on a log scale. Default is TRUE.
`grid`	Logical indicating whether output should be a list of individual plots ("individual"), or one facetted plot with all variables ("assembled"). Defaults to "individual".
`sample.facet`	Logical indicating whether to facet the plot by sample, i.e, showing plots separate for each sample, and side to side. Defaults to FALSE.
`nrow.panel`	Integer indicating the number of rows in the assembled plot. If NULL, the number of rows is automatically calculated.
`tol`	Numeric indicating the tolerance: values below this value will be set to the tolerance value, for legibility of the plots
`...`	Additional arguments passed to the predict() function.

Value

Scatter plot of density ratios and individual variables.

Examples

```
set.seed(123)
# Fit model
dr <- ulsif(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
ulsif(numerator_small, denominator_small, sigma = 2, lambda = 2)
```

`predict.kliep` *Obtain predicted density ratio values from a* `kliep` *object*

Description

Obtain predicted density ratio values from a `kliep` object

Usage

```
## S3 method for class 'kliep'
predict(object, newdata = NULL, sigma = c("sigmaopt", "all"), ...)
```

Arguments

`object`	A `kliep` object
`newdata`	Optional `matrix` new data set to compute the density
`sigma`	A scalar with the Gaussian kernel width
`...`	Additional arguments to be passed to the function

Value

An array with predicted density ratio values from possibly new data, but otherwise the numerator samples.

See Also

`predict`, `kliep`

Examples

```
set.seed(123)
# Fit model
dr <- kliep(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
kliep(numerator_small, denominator_small,
      nsigma = 1, ncenters = 100, nfold = 10,
      epsilon = 10^{2:-5}, maxit = 500)
```

`predict.kmm` *Obtain predicted density ratio values from a `kmm` object*

Description

Obtain predicted density ratio values from a `kmm` object

Usage

```
## S3 method for class 'kmm'
predict(object, newdata = NULL, sigma = c("sigmaopt", "all"), ...)
```

Arguments

`object`	A `kmm` object
`newdata`	Optional `matrix` new data set to compute the density
`sigma`	A scalar with the Gaussian kernel width
`...`	Additional arguments to be passed to the function

Value

An array with predicted density ratio values from possibly new data, but otherwise the numerator samples.

See Also

`predict`, `kmm`

Examples

```
set.seed(123)
# Fit model
dr <- kmm(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
kmm(numerator_small, denominator_small,
    nsigma = 5, ncenters = 100, nfold = 10,
```

```
    constrained = TRUE)
```

`predict.lhss` *Obtain predicted density ratio values from a* `lhss` *object*

Description

Obtain predicted density ratio values from a `lhss` object

Usage

```
## S3 method for class 'lhss'
predict(
  object,
  newdata = NULL,
  sigma = c("sigmaopt", "all"),
  lambda = c("lambdaopt", "all"),
  ...
)
```

Arguments

`object`	A `lhss` object
`newdata`	Optional `matrix` new data set to compute the density
`sigma`	A scalar with the Gaussian kernel width
`lambda`	A scalar with the regularization parameter
`...`	Additional arguments to be passed to the function

Value

An array with predicted density ratio values from possibly new data, but otherwise the numerator samples.

See Also

`predict`, `lhss`

Examples

```
set.seed(123)
# Fit model (minimal example to limit computation time)
dr <- lhss(numerator_small, denominator_small,
           nsigma = 5, nlambda = 3, ncenters = 50, maxit = 100)
# Inspect model object
dr
# Obtain summary of model object
```

```
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
```

`predict.naivedensityratio` *Obtain predicted density ratio values from a* `naivedensityratio` *object*

Description

Obtain predicted density ratio values from a `naivedensityratio` object

Usage

```
## S3 method for class 'naivedensityratio'
predict(object, newdata = NULL, log = FALSE, tol = 1e-06, ...)
```

Arguments

`object`	A `naive` object
`newdata`	Optional `matrix` new data set to compute the density
`log`	A logical indicating whether to return the log of the density ratio
`tol`	Minimal density value to avoid numerical issues
`...`	Additional arguments to be passed to the function

Value

An array with predicted density ratio values from possibly new data, but otherwise the numerator samples.

See Also

`predict`, `naive`

Examples

```
set.seed(123)
# Fit model
dr <- naive(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
naive(numerator_small, denominator_small, m=2, kernel="epanechnikov")
```

`predict.spectral` *Obtain predicted density ratio values from a* `spectral` *object*

Description

Obtain predicted density ratio values from a `spectral` object

Usage

```
## S3 method for class 'spectral'
predict(
  object,
  newdata = NULL,
  sigma = c("sigmaopt", "all"),
  m = c("opt", "all"),
  ...
)
```

Arguments

`object`	A `spectral` object
`newdata`	Optional `matrix` new data set to compute the density
`sigma`	A scalar with the Gaussian kernel width
`m`	integer indicating the dimension of the eigenvector expansion
`...`	Additional arguments to be passed to the function

Value

An array with predicted density ratio values from possibly new data, but otherwise the numerator samples.

See Also

`predict`, `spectral`

`predict.ulsif` *Obtain predicted density ratio values from a* `ulsif` *object*

Description

Obtain predicted density ratio values from a `ulsif` object

Usage

```
## S3 method for class 'ulsif'
predict(
  object,
  newdata = NULL,
  sigma = c("sigmaopt", "all"),
  lambda = c("lambdaopt", "all"),
  ...
)
```

Arguments

`object`	A `ulsif` object
`newdata`	Optional `matrix` new data set to compute the density
`sigma`	A scalar with the Gaussian kernel width
`lambda`	A scalar with the regularization parameter
`...`	Additional arguments to be passed to the function

Value

An array with predicted density ratio values from possibly new data, but otherwise the numerator samples.

See Also

`predict`, `ulsif`

Examples

```
set.seed(123)
# Fit model
dr <- ulsif(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
ulsif(numerator_small, denominator_small, sigma = 2, lambda = 2)
```

`print.kliep` *Print a* `kliep` *object*

Description

Print a `kliep` object

Usage

```
## S3 method for class 'kliep'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

`x`	Object of class `kliep`.
`digits`	Number of digits to use when printing the output.
`...`	further arguments on how to format the number of digits.

Value

`invisble` The inputted `kliep` object.

See Also

`print`, `kliep`

Examples

```
set.seed(123)
# Fit model
dr <- kliep(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
kliep(numerator_small, denominator_small,
      nsigma = 1, ncenters = 100, nfold = 10,
      epsilon = 10^{2:-5}, maxit = 500)
```

`print.kmm` *Print a* `kmm` *object*

Description

Print a `kmm` object

Usage

```
## S3 method for class 'kmm'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

`x`	Object of class `kmm`.
`digits`	Number of digits to use when printing the output.
`...`	further arguments on how to format the number of digits.

Value

`invisble` The inputted `kmm` object.

See Also

`print`, `kmm`

Examples

```
set.seed(123)
# Fit model
dr <- kmm(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
kmm(numerator_small, denominator_small,
    nsigma = 5, ncenters = 100, nfold = 10,
    constrained = TRUE)
```

`print.lhss` *Print a* `lhss` *object*

Description

Print a `lhss` object

Usage

```
## S3 method for class 'lhss'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

`x`	Object of class `lhss`.
`digits`	Number of digits to use when printing the output.
`...`	further arguments on how to format the number of digits.

Value

`invisble` The inputted `lhss` object.

See Also

`print`, `lhss`

Examples

```
set.seed(123)
# Fit model (minimal example to limit computation time)
dr <- lhss(numerator_small, denominator_small,
           nsigma = 5, nlambda = 3, ncenters = 50, maxit = 100)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
```

`print.naivedensityratio`

Print a `naivedensityratio` *object*

Description

Print a `naivedensityratio` object

Usage

```
## S3 method for class 'naivedensityratio'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

`x`	Object of class `naivesubspacedensityratio`.
`digits`	Number of digits to use when printing the output.
`...`	further arguments on how to format the number of digits.

Value

`invisble` The inputted `naivedensityratio` object.

See Also

`print`, `naive`

Examples

```
set.seed(123)
# Fit model
dr <- naive(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
naive(numerator_small, denominator_small, m=2, kernel="epanechnikov")
```

`print.spectral` *Print a* `spectral` *object*

Description

Print a `spectral` object

Usage

```
## S3 method for class 'spectral'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

`x`	Object of class `spectral`.
`digits`	Number of digits to use when printing the output.
`...`	further arguments on how to format the number of digits.

Value

`invisble` The inputted `spectral` object.

See Also

`print`, `spectral`

Examples

```
set.seed(123)
# Fit model
dr <- spectral(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
spectral(numerator_small, denominator_small, sigma = 2)
```

`print.summary.kliep` *Print a* `summary.kliep` *object*

Description

Print a `summary.kliep` object

Usage

```
## S3 method for class 'summary.kliep'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

`x`	Object of class `summary.kliep`.
`digits`	Number of digits to use when printing the output.
`...`	further arguments on how to format the number of digits.

Value

`invisble` The inputted `summary.kliep` object.

See Also

`print`, `summary.kliep`, `kliep`

Examples

```
set.seed(123)
# Fit model
dr <- kliep(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
kliep(numerator_small, denominator_small,
      nsigma = 1, ncenters = 100, nfold = 10,
      epsilon = 10^{2:-5}, maxit = 500)
```

`print.summary.kmm` *Print a* `summary.kmm` *object*

Description

Print a `summary.kmm` object

Usage

```
## S3 method for class 'summary.kmm'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

`x`	Object of class `summary.kmm`.
`digits`	Number of digits to use when printing the output.
`...`	further arguments on how to format the number of digits.

Value

`invisble` The inputted `summary.kmm` object.

See Also

`print`, `summary.kmm`, `kmm`

Examples

```
set.seed(123)
# Fit model
dr <- kmm(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
kmm(numerator_small, denominator_small,
    nsigma = 5, ncenters = 100, nfold = 10,
    constrained = TRUE)
```

`print.summary.lhss` *Print a* `summary.lhss` *object*

Description

Print a `summary.lhss` object

Usage

```
## S3 method for class 'summary.lhss'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

`x`	Object of class `summary.lhss`.
`digits`	Number of digits to use when printing the output.
`...`	further arguments on how to format the number of digits.

Value

`invisble` The inputted `summary.lhss` object.

See Also

`print`, `summary.lhss`, `lhss`

Examples

```
set.seed(123)
# Fit model (minimal example to limit computation time)
dr <- lhss(numerator_small, denominator_small,
           nsigma = 5, nlambda = 3, ncenters = 50, maxit = 100)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
```

`print.summary.naivedensityratio`

Print a `summary.naivedensityratio` *object*

Description

Print a `summary.naivedensityratio` object

Usage

```
## S3 method for class 'summary.naivedensityratio'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

`x`	Object of class `summary.naivedensityratio`.
`digits`	Number of digits to use when printing the output.
`...`	further arguments on how to format the number of digits.

Value

`invisble` The inputted `summary.naivedensityratio` object.

See Also

`print`, `summary.naivedensityratio`, `naive`

Examples

```
set.seed(123)
# Fit model
dr <- naive(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
naive(numerator_small, denominator_small, m=2, kernel="epanechnikov")
```

`print.summary.spectral`

Print a `summary.spectral` *object*

Description

Print a `summary.spectral` object

Usage

```
## S3 method for class 'summary.spectral'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

`x`	Object of class `summary.spectral`.
`digits`	Number of digits to use when printing the output.
`...`	further arguments on how to format the number of digits.

Value

`invisble` The inputted `summary.spectral` object.

See Also

`print`, `summary.spectral`, `spectral`

Examples

```
set.seed(123)
# Fit model
dr <- spectral(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
spectral(numerator_small, denominator_small, sigma = 2)
```

`print.summary.ulsif` *Print a* `summary.ulsif` *object*

Description

Print a `summary.ulsif` object

Usage

```
## S3 method for class 'summary.ulsif'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

`x`	Object of class `summary.ulsif`.
`digits`	Number of digits to use when printing the output.
`...`	further arguments on how to format the number of digits.

Value

`invisble` The inputted `summary.ulsif` object.

See Also

`print`, `summary.ulsif`, `ulsif`

Examples

```
set.seed(123)
# Fit model
dr <- ulsif(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
ulsif(numerator_small, denominator_small, sigma = 2, lambda = 2)
```

`print.ulsif` *Print a* `ulsif` *object*

Description

Print a `ulsif` object

Usage

```
## S3 method for class 'ulsif'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

`x`	Object of class `ulsif`.
`digits`	Number of digits to use when printing the output.
`...`	further arguments on how to format the number of digits.

Value

`invisble` The inputted `ulsif` object.

See Also

`print`, `ulsif`

Examples

```
set.seed(123)
# Fit model
dr <- ulsif(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
ulsif(numerator_small, denominator_small, sigma = 2, lambda = 2)
```

`spectral` *Spectral series based density ratio estimation*

Description

Spectral series based density ratio estimation

Usage

```
spectral(
  df_numerator,
  df_denominator,
  m = NULL,
  scale = "numerator",
  nsigma = 10,
  sigma_quantile = NULL,
  sigma = NULL,
  ncenters = NULL,
  cv = TRUE,
  nfold = 10,
  parallel = FALSE,
  nthreads = NULL,
  progressbar = TRUE
)
```

Arguments

`df_numerator`	`data.frame` with exclusively numeric variables with the numerator samples

`df_denominator`	`data.frame` with exclusively numeric variables with the denominator samples (must have the same variables as `df_denominator`)
`m`	Integer vector indicating the number of eigenvectors to use in the spectral series expansion. Defaults to 50 evenly spaced values between 1 and the number of denominator samples (or the largest number of samples that can be used as centers in the cross-validation scheme).
`scale`	`"numerator"`, `"denominator"`, or `NULL`, indicating whether to standardize each numeric variable according to the numerator means and standard deviations, the denominator means and standard deviations, or apply no standardization at all.
`nsigma`	Integer indicating the number of sigma values (bandwidth parameter of the Gaussian kernel gram matrix) to use in cross-validation.
`sigma_quantile`	`NULL` or numeric vector with probabilities to calculate the quantiles of the distance matrix to obtain sigma values. If `NULL`, `nsigma` values between `0.05` and `0.95` are used.
`sigma`	`NULL` or a scalar value to determine the bandwidth of the Gaussian kernel gram matrix. If `NULL`, `nsigma` values between `0.05` and `0.95` are used.
`ncenters`	integer If smaller than the number of denominator observations, an approximation to the eigenvector expansion based on only ncenters samples is performed, instead of the full expansion. This can be useful for large datasets. Defaults to `NULL`, such that all denominator samples are used.
`cv`	logical indicating whether to use cross-validation to determine the optimal sigma value and the optimal number of eigenvectors.
`nfold`	Integer indicating the number of folds to use in the cross-validation scheme. If `cv` is `FALSE`, this parameter is ignored.
`parallel`	logical indicating whether to use parallel processing in the cross-validation scheme.
`nthreads`	`NULL` or integer indicating the number of threads to use for parallel processing. If parallel processing is enabled, it defaults to the number of available threads minus one.
`progressbar`	Logical indicating whether or not to display a progressbar.

Value

`spectral`-object, containing all information to calculate the density ratio using optimal sigma and optimal spectral series expansion.

References

Izbicki, R., Lee, A. & Schafer, C. (2014). High-Dimensional Density Ratio Estimation with Extensions to Approximate Likelihood Computation. Proceedings of Machine Learning Research 33, 420-429. Available from `https://proceedings.mlr.press/v33/izbicki14.html`.

Examples

```
set.seed(123)
# Fit model
dr <- spectral(numerator_small, denominator_small)
```

```
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
spectral(numerator_small, denominator_small, sigma = 2)
```

`summary.kliep` *Extract summary from* `kliep` *object, including two-sample significance test for homogeneity of the numerator and denominator samples*

Description

Extract summary from `kliep` object, including two-sample significance test for homogeneity of the numerator and denominator samples

Usage

```
## S3 method for class 'kliep'
summary(
  object,
  test = FALSE,
  n_perm = 100,
  parallel = FALSE,
  cluster = NULL,
  min_pred = 1e-06,
  ...
)
```

Arguments

`object`	Object of class `kliep`
`test`	logical indicating whether to statistically test for homogeneity of the numerator and denominator samples.
`n_perm`	Scalar indicating number of permutation samples
`parallel`	`logical` indicating to run the permutation test in parallel
`cluster`	`NULL` or a cluster object created by `makeCluster`. If `NULL` and `parallel = TRUE`, it uses the number of available cores minus 1.
`min_pred`	Scalar indicating the minimum value for the predicted density ratio values (used in the divergence statistic) to avoid negative density ratio values.
`...`	further arguments passed to or from other methods.

Value

Summary of the fitted density ratio model

Examples

```
set.seed(123)
# Fit model
dr <- kliep(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
kliep(numerator_small, denominator_small,
      nsigma = 1, ncenters = 100, nfold = 10,
      epsilon = 10^{2:-5}, maxit = 500)
```

`summary.kmm` *Extract summary from* `kmm` *object, including two-sample significance test for homogeneity of the numerator and denominator samples*

Description

Extract summary from `kmm` object, including two-sample significance test for homogeneity of the numerator and denominator samples

Usage

```
## S3 method for class 'kmm'
summary(
  object,
  test = FALSE,
  n_perm = 100,
  parallel = FALSE,
  cluster = NULL,
  min_pred = 1e-06,
  ...
)
```

Arguments

`object`	Object of class `kmm`
`test`	logical indicating whether to statistically test for homogeneity of the numerator and denominator samples.
`n_perm`	Scalar indicating number of permutation samples
`parallel`	`logical` indicating to run the permutation test in parallel
`cluster`	`NULL` or a cluster object created by `makeCluster`. If `NULL` and `parallel = TRUE`, it uses the number of available cores minus 1.
`min_pred`	Scalar indicating the minimum value for the predicted density ratio values (used in the divergence statistic) to avoid negative density ratio values.
`...`	further arguments passed to or from other methods.

Value

Summary of the fitted density ratio model

Examples

```
set.seed(123)
# Fit model
dr <- kmm(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
kmm(numerator_small, denominator_small,
    nsigma = 5, ncenters = 100, nfold = 10,
    constrained = TRUE)
```

`summary.lhss` *Extract summary from* `lhss` *object, including two-sample significance test for homogeneity of the numerator and denominator samples*

Description

Extract summary from `lhss` object, including two-sample significance test for homogeneity of the numerator and denominator samples

Usage

```
## S3 method for class 'lhss'
summary(
  object,
  test = FALSE,
  n_perm = 100,
  parallel = FALSE,
  cluster = NULL,
  ...
)
```

Arguments

`object`	Object of class `lhss`
`test`	logical indicating whether to statistically test for homogeneity of the numerator and denominator samples.
`n_perm`	Scalar indicating number of permutation samples
`parallel`	`logical` indicating to run the permutation test in parallel
`cluster`	`NULL` or a cluster object created by `makeCluster`. If `NULL` and `parallel` = `TRUE`, it uses the number of available cores minus 1.
`...`	further arguments passed to or from other methods.

Value

Summary of the fitted density ratio model

Examples

```
set.seed(123)
# Fit model (minimal example to limit computation time)
dr <- lhss(numerator_small, denominator_small,
           nsigma = 5, nlambda = 3, ncenters = 50, maxit = 100)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
```

`summary.naivedensityratio` *Extract summary from* `naivedensityraito` *object, including two-sample significance test for homogeneity of the numerator and denominator samples*

Description

Extract summary from `naivedensityraito` object, including two-sample significance test for homogeneity of the numerator and denominator samples

Usage

```
## S3 method for class 'naivedensityratio'
summary(
  object,
  test = FALSE,
  n_perm = 100,
  parallel = FALSE,
  cluster = NULL,
  ...
)
```

Arguments

`object`	Object of class `naivedensityratio`
`test`	logical indicating whether to statistically test for homogeneity of the numerator and denominator samples.
`n_perm`	Scalar indicating number of permutation samples
`parallel`	`logical` indicating to run the permutation test in parallel
`cluster`	`NULL` or a cluster object created by `makeCluster`. If `NULL` and `parallel = TRUE`, it uses the number of available cores minus 1.
`...`	further arguments passed to or from other methods.

Value

Summary of the fitted density ratio model

Examples

```
set.seed(123)
# Fit model
dr <- naive(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
```

```
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
naive(numerator_small, denominator_small, m=2, kernel="epanechnikov")
```

`summary.spectral` *Extract summary from* `spectral` *object, including two-sample significance test for homogeneity of the numerator and denominator samples*

Description

Extract summary from `spectral` object, including two-sample significance test for homogeneity of the numerator and denominator samples

Usage

```
## S3 method for class 'spectral'
summary(
  object,
  test = FALSE,
  n_perm = 100,
  parallel = FALSE,
  cluster = NULL,
  ...
)
```

Arguments

`object`	Object of class `spectral`
`test`	logical indicating whether to statistically test for homogeneity of the numerator and denominator samples.
`n_perm`	Scalar indicating number of permutation samples
`parallel`	`logical` indicating to run the permutation test in parallel
`cluster`	`NULL` or a cluster object created by `makeCluster`. If `NULL` and `parallel = TRUE`, it uses the number of available cores minus 1.
`...`	further arguments passed to or from other methods.

Value

Summary of the fitted density ratio model

Examples

```
set.seed(123)
# Fit model
dr <- spectral(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
spectral(numerator_small, denominator_small, sigma = 2)
```

`summary.ulsif` *Extract summary from* `ulsif` *object, including two-sample significance test for homogeneity of the numerator and denominator samples*

Description

Extract summary from `ulsif` object, including two-sample significance test for homogeneity of the numerator and denominator samples

Usage

```
## S3 method for class 'ulsif'
summary(
  object,
  test = FALSE,
  n_perm = 100,
  parallel = FALSE,
  cluster = NULL,
  ...
)
```

Arguments

`object`	Object of class `ulsif`
`test`	logical indicating whether to statistically test for homogeneity of the numerator and denominator samples.
`n_perm`	Scalar indicating number of permutation samples
`parallel`	`logical` indicating to run the permutation test in parallel

`cluster`	`NULL` or a cluster object created by `makeCluster`. If `NULL` and `parallel = TRUE`, it uses the number of available cores minus 1.
`...`	further arguments passed to or from other methods.

Value

Summary of the fitted density ratio model

Examples

```
set.seed(123)
# Fit model
dr <- ulsif(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
ulsif(numerator_small, denominator_small, sigma = 2, lambda = 2)
```

`ulsif` *Unconstrained least-squares importance fitting*

Description

Unconstrained least-squares importance fitting

Usage

```
ulsif(
  df_numerator,
  df_denominator,
  intercept = TRUE,
  scale = "numerator",
  nsigma = 10,
  sigma_quantile = NULL,
  sigma = NULL,
  nlambda = 20,
  lambda = NULL,
  ncenters = 200,
```

```
  centers = NULL,
  parallel = FALSE,
  nthreads = NULL,
  progressbar = TRUE
)
```

Arguments

`df_numerator`	`data.frame` with exclusively numeric variables with the numerator samples
`df_denominator`	`data.frame` with exclusively numeric variables with the denominator samples (must have the same variables as `df_denominator`)
`intercept`	`logical` Indicating whether to include an intercept term in the model. Defaults to `TRUE`.
`scale`	`"numerator"`, `"denominator"`, or `NULL`, indicating whether to standardize each numeric variable according to the numerator means and standard deviations, the denominator means and standard deviations, or apply no standardization at all.
`nsigma`	Integer indicating the number of sigma values (bandwidth parameter of the Gaussian kernel gram matrix) to use in cross-validation.
`sigma_quantile`	`NULL` or numeric vector with probabilities to calculate the quantiles of the distance matrix to obtain sigma values. If `NULL`, `nsigma` values between `0.05` and `0.95` are used.
`sigma`	`NULL` or a scalar value to determine the bandwidth of the Gaussian kernel gram matrix. If `NULL`, `nsigma` values between `0.05` and `0.95` are used.
`nlambda`	Integer indicating the number of `lambda` values (regularization parameter), by default, `lambda` is set to `10^seq(3, -3, length.out = nlambda)`.
`lambda`	`NULL` or numeric vector indicating the lambda values to use in cross-validation
`ncenters`	Maximum number of Gaussian centers in the kernel gram matrix. Defaults to all numerator samples.
`centers`	`NULL` or numeric matrix with the same dimensions as the data, indicating the centers for the Gaussian kernel gram matrix.
`parallel`	logical indicating whether to use parallel processing in the cross-validation scheme.
`nthreads`	`NULL` or integer indicating the number of threads to use for parallel processing. If parallel processing is enabled, it defaults to the number of available threads minus one.
`progressbar`	Logical indicating whether or not to display a progressbar.

Value

`ulsif`-object, containing all information to calculate the density ratio using optimal sigma and optimal weights.

References

Kanamori, T., Hido, S., & Sugiyama, M. (2009). A least-squares approach to direct importance estimation. Journal of Machine Learning Research, 10, 1391-1445. Available from https://jmlr.org/papers/v10/kanamori09a.html

Examples

```
set.seed(123)
# Fit model
dr <- ulsif(numerator_small, denominator_small)
# Inspect model object
dr
# Obtain summary of model object
summary(dr)
# Plot model object
plot(dr)
# Plot density ratio for each variable individually
plot_univariate(dr)
# Plot density ratio for each pair of variables
plot_bivariate(dr)
# Predict density ratio and inspect first 6 predictions
head(predict(dr))
# Fit model with custom parameters
ulsif(numerator_small, denominator_small, sigma = 2, lambda = 2)
```

Index