Package: densityratio (via r-universe)

August 23, 2024

Type Package

Title Distribution comparison through density ratio estimation

Version 0.1.1

Description This package provides functionality to directly estimate the ratio of two probability distributions from samples from these distributions without estimating the densities separately. Density ratio estimation serves many purposes, for example, prediction, outlier detection, change-point detection in time-series, importance weighting under domain adaptation (i.e., sample selection bias) and evaluation of synthetic data utility. The rationale behind these use-cases is that differences between two data distributions can be captured in the ratio their density ratio, which is estimated over the entire multivariate space of the data. Computationally intensive code is executed in `C++` using `Rcpp` and `RcppArmadillo`. The package provides good default hyperparameters that can be optimized in cross-validation (we do recommend understanding those parameters before using `densityratio` in practice). Multiple density ratio estimation methods are implemented, such as unconstrained least-squares importance fitting (`ulsif()`), Kullback-Leibler importance estimation procedure (`kliep()`), ratio of estimated densities (`naive()`), ratio of estimated densities after dimension reduction (`naivesubspace()`), and least-squares heterodistributional subspace search (`lhss()`; experimental).

License GPL (>= 3)

Encoding UTF-8

LazyData true

Imports quadprog, Rcpp, pbapply, ggplot2

LinkingTo Rcpp, RcppArmadillo, RcppProgress

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

Config/testthat/parallel true

Roxygen list(markdown = TRUE)

Depends R (>= 2.10)

RoxygenNote 7.2.3

VignetteBuilder knitr

Repository https://thomvolker.r-universe.dev

RemoteUrl https://github.com/thomvolker/densityratio

RemoteRef HEAD

RemoteSha 796368446c5cf2f87b1beeb2967f9a2f0ae9879a

Contents

`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` *Indivual 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)

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 '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,
  ...
)
```

```
## S3 method for class 'naivesubspacedensityratio'
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.

`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

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.

Examples

```
set.seed(1)
x <- rnorm(100) |> matrix(100)
y <- rnorm(200, 1, 2) |> matrix(200)
kliep(x, y)
kliep(x, y, nsigma = 20, ncenters = 100, nfold = 20, epsilon = 10^{3:-5}, maxit = 1000)
```

`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",
  method = "unconstrained",
  sigma = NULL,
  lambda = 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.
`method`	Character string containing the method used for kernel mean matching. Currently, `method = "unconstrained"` and `method = "constrained"` are supported.
`sigma`	`NULL` or a scalar value to determine the bandwidth of the Gaussian kernel gram matrix. If `NULL`, `sigma` is the median Euclidean interpoint distance.
`lambda`	`NULL` or a scalar value to determine the regularization imposed on the Gaussian kernel gram matrix of the denominator samples. If `NULL`, `lambda` is chosen to be $\sqrt{N}$.

Value

`kmm` returns `rhat_de`, the estimated density ratio for the denominator samples.

Examples

```
x <- rnorm(100) |> matrix(100)
y <- rnorm(200, 1, 2) |> matrix(200)
kmm(x, y)
kmm(x, y, sigma = 2, lambda = 2)
```

`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,
  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`	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.
`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

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

lhss returns rhat, the estimated density ratio.

Examples

```
set.seed(1)
N <- 250
X <- cbind(rnorm(N), rnorm(N, 0, 0.5))
Y <- cbind(rnorm(N), sample(rep(c(-1, 1), times = N/2)) + rnorm(N))
out <- lhss(X, Y, m = 1, ncenters = 100)
```

---

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, this approach assumes the variables are independent (naive Bayes assumption).

Usage

```
naive(df_numerator, df_denominator, 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)
n	integer the number of equally spaced points at which the density is estimated. When n > 512, it is rounded up to a power of 2 during the calculations (as fft 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

```
x <- rnorm(100)
y <- rnorm(200, 1, 2)

naive(x, y)
naive(x, y, bw = 2)
```

`naivesubspace` *Naive subspace density ratio estimation*

Description

The naive subspace estimator first creates an m-dimensional representation of the data using singular value decomposition, and then runs the naive density ratio estimation procedure on the data projected on this subspace. The SVD is computed using the denominator samples.

Usage

```
naivesubspace(df_numerator, df_denominator, m = NULL, 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`	The size (in number of features) of the subspace
`n`	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 fft is used) and the final result is interpolated by stats::approx. So it almost always makes sense to specify n as a power of two.
`...`	further arguments passed to stats::density

Examples

```
set.seed(456)
# create data that differs only on the first variable
N <- 100
P <- 3 # P-1 noise variables
X <- matrix(rnorm(N*P), N)
Y <- cbind(rnorm(N, 1, 2), matrix(rnorm(N*P-N), N))
df_X <- data.frame(X)
df_Y <- data.frame(Y)

# estimate
dr_naive <- naive(df_X, df_Y)
dr_subspace <- naivesubspace(df_X, df_Y, 1)
```

```
# plot: true, naive, naive_subspace
df_new <- data.frame(
  X1 = seq(-4, 4, length.out = 100),
  X2 = seq(-4, 4, length.out = 100),
  X3 = seq(-4, 4, length.out = 100)
)
true_dr <- dnorm(df_new[,1]) / dnorm(df_new[,1], 1, 2)
plot(df_new[,1], true_dr, type = "l", ylab = "Density ratio", ylim = c(0, 3.2))
lines(df_new[,1], predict(dr_naive, df_new), col = "lightblue")
lines(df_new[,1], predict(dr_subspace, df_new), col = "darkorange")
```

`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)

`permute` *Single permutation*

Description

Single permutation

Single permutation statistic of `ulsif` object

Single permutation statistic of `kliep` object

Single permutation statistic of `lhss` object

Single permutation statistic of `spectral` object

Single permutation statistic of `naivedensityratio` object

Single permutation statistic of `naivesubspacedensityratio` 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 '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)

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

Arguments

`object`	`naivesubspacedensityratio` 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 bidimensional 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.

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.

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

```
x <- rnorm(100) |> matrix(100)
y <- rnorm(200, 1, 2) |> matrix(200)
fit1 <- kliep(x, y)
predict(fit1)
predict(fit1, newdata = rbind(x, y))
predict(fit1, newdata = rbind(x, y), sigma = 2)
```

`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

```
x <- rnorm(100) |> matrix(50)
y <- rnorm(200, 1, 2) |> matrix(100)
fit1 <- lhss(x, y, m = 1)
predict(fit1)
predict(fit1, newdata = rbind(x, y))
predict(fit1, newdata = rbind(x, y), sigma = 2)
```

`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, ...)
```

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
`...`	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

```
x <- rnorm(100) |> matrix(100)
y <- rnorm(200, 1, 2) |> matrix(200)
fit1 <- naive(x, y)
predict(fit1)
predict(fit1, newdata = rbind(x, y))
predict(fit1, newdata = rbind(x, y), log = TRUE)
```

predict.naivesubspacedensityratio

Obtain predicted density ratio values from a `naivesubspace` *object*

Description

Obtain predicted density ratio values from a `naivesubspace` object

Usage

```
## S3 method for class 'naivesubspacedensityratio'
predict(object, newdata = NULL, log = FALSE, ...)
```

Arguments

`object`	A `naivesubspace` object
`newdata`	Optional `matrix` new data set to compute the density
`log`	A logical indicating whether to return the log of the density ratio
`...`	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`, `naivesubspace`

Examples

```
x <- rnorm(100) |> matrix(100)
y <- rnorm(200, 1, 2) |> matrix(200)
fit1 <- naivesubspace(x, y)
predict(fit1)
predict(fit1, newdata = rbind(x, y))
predict(fit1, newdata = rbind(x, y), log = TRUE)
```

`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"),
  J = c("Jopt", "all"),
  ...
)
```

Arguments

`object`	A `spectral` object
`newdata`	Optional `matrix` new data set to compute the density
`sigma`	A scalar with the Gaussian kernel width
`J`	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`

Examples

```
x <- rnorm(100) |> matrix(100)
y <- rnorm(200, 1, 2) |> matrix(200)
fit1 <- spectral(x, y)
predict(fit1)
predict(fit1, newdata = rbind(x, y))
predict(fit1, newdata = rbind(x, y), sigma = 2, J = 10)
```

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

```
x <- rnorm(100) |> matrix(100)
y <- rnorm(200, 1, 2) |> matrix(200)
fit1 <- ulsif(x, y)
predict(fit1)
predict(fit1, newdata = rbind(x, y))
predict(fit1, newdata = rbind(x, y), sigma = 2, lambda = 3)
```

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

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`

`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`

print.naivesubspacedensityratio

Print a `naivedensityratio` *object*

Description

Print a `naivedensityratio` object

Usage

```
## S3 method for class 'naivesubspacedensityratio'
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 `naivesubspacedensityratio` object.

See Also

`print`, `naivesubspace`

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`

`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`

`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`

`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`

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

Description

Print a `summary.naivesubspacedensityratio` object

Usage

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

Arguments

`x`	Object of class `summary.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 `summary.naivesubspacedensityratio` object.

See Also

`print`, `summary.naivesubspacedensityratio`, `naive`

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

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

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`

`spectral` *Spectral series based density ratio estimation*

Description

Spectral series based density ratio estimation

Usage

```
spectral(
  df_numerator,
  df_denominator,
  J = NULL,
  scale = "numerator",
  nsigma = 10,
  sigma_quantile = NULL,
  sigma = NULL,
  ncenters = nrow(df_denominator),
  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`)
`J`	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.
`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(1)
x <- rnorm(100) |> matrix(100)
y <- rnorm(200, 1, 2) |> matrix(200)
spectral(x, y)
spectral(x, y, 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,
  cl = 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
`cl`	`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

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, cl = 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
`cl`	`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

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, cl = 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
`cl`	`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

`summary.naivesubspacedensityratio`

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

Description

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

Usage

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

Arguments

`object`	Object of class `naivesubspacedensityratio`
`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
`cl`	`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

`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, cl = 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
`cl`	`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

`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, cl = 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
`cl`	`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

`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.

Examples

```
set.seed(1)
x <- rnorm(100) |> matrix(100)
y <- rnorm(200, 1, 2) |> matrix(200)
ulsif(x, y)
ulsif(x, y, sigma = 2, lambda = 2)
```

Index