Package 'kernda'

July 4, 2026

Type Package

Title Kernel Discriminant Analysis

Version 1.0

Date 2026-06-29

Author Michail Tsagris [aut, cre],
Nikolaos Kontemeniotis [aut]

Maintainer Michail Tsagris <mtsagris@uoc.gr>

Depends R (>= 4.0)

Imports Compositional, Rfast, Rcpp, kernreg

LinkingTo Rcpp

Description Functions to perform discriminant analysis using kernel density estimation. For the case of binary classification the package relies on the Nadaraya-Watson estimator. References: Wand M.P. and Jones M.C. (1995). ``Kernel smoothing". CRC Press. <ISBN: 0412552701>.

License GPL (>= 2)

RoxygenNote 7.3.3

NeedsCompilation yes

Repository CRAN

Date/Publication 2026-07-04 08:40:13 UTC

Contents

kernda-package 2
cv.kdeda 2
cv.lrkde 4
kde_da 5
lr.kde 6

Index **8**

kernda-package *Kernel Discriminant Analysis*

Description

Functions to perform discriminant analysis using kernel density estimation. For the case of binary classification the package relies on the Nadaraya-Watson estimator. References: Wand M.P. and Jones M.C. (1994). Kernel smoothing. CRC press.

Details

Package:	kernda
Type:	Package
Version:	1.0
Date:	2026-06-29

Maintainers

Michail Tsagris <mtsagris@uoc.gr>.

Author(s)

Michail Tsagris <mtsagris@uoc.gr> and Nikolaos Kontemeniotis <kontemeniotisn@gmail.com>

cv.kdeda *Cross-validation for the kernel discrimininant analysis*

Description

Cross-validation for the kernel discrimininant analysis.

Usage

```
cv.kdeda(groups, x, h = seq(0.1, 1, length = 10), nfolds = 10,
         folds = NULL, seed = NULL, ncores = 1)
```

Arguments

groups	A variable indicating the groupings.
x	A numerical vector with the data.
h	A vector with the bandwidth parameter values of the Gaussian kernel.
nfolds	How many folds to create?
folds	Do you already have a list with the folds? If not, leave this NULL.
seed	If seed is TRUE, the results will always be the same.
ncores	The number of cores to use.

Details

This is described in Chapter 8.2 of Hansen (2019). The idea is to inimise the sum of squares of the residuals under the constraint $R^\top \boldsymbol{\beta} = c$. As mentioned above, be careful with the input you give in the x matrix and the R vector. The cls() function performs a single regression model, whereas the mcls() function performs a regression for each column of y. Each regression is independent of the others.

Value

A list including:

cv	A numerical matrix with the percentage of correct classification for each fold and bandwidth value.
perf	The average percentage of correct classification for each bandwidth value.
hopt	The optimal bandwidth value.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

Wand M.P. and Jones M.C. (1995). Kernel smoothing. CRC Press.

Hastie T., Tibshirani R. and Friedman J. (2009). The elements of statistical learning.

See Also

kde_da, lr.kde

Examples

```
x <- iris[, 4]
groups <- as.numeric( iris[, 5] )
a <- cv.kdeda(groups, x)
```

`cv.lrkde` *Cross-validation for the kernel regression with a binary response*

Description

Cross-validation for the kernel regression with a binary response.

Usage

```
cv.lrkde(groups, x, h = seq(0.1, 1, length = 10), nfolds = 10,
folds = NULL, seed = NULL, graph = FALSE, ncores = 1)
```

Arguments

`groups`	A vector with the binary response (0s and 1s).
`x`	A matrix with the available predictor variables.
`h`	A vector with the bandwidth value(s) h to consider.
`nfolds`	The number of folds. Set to 10 by default.
`folds`	If you have the list with the folds supply it here. You can also leave it NULL and it will create folds.
`seed`	You can specify your own seed number here or leave it NULL.
`graph`	If graph is TRUE (default value) a plot will appear.
`ncores`	The number of cores to use. Default value is 1.

Details

A k-fold cross-validation for the kernel regression with a binary response is performed and the AUC is computed.

Value

A list including:

`cv`	A numerical matrix with the percentage of correct classification for each fold and bandwidth value.
`perf`	The average percentage of correct classification for each bandwidth value.
`hopt`	The optimal bandwidth value.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

Wand M. P. and Jones M. C. (1994). Kernel smoothing. CRC press.

See Also

lr.kde

Examples

```
groups <- as.numeric( iris[1:100, 5] ) - 1
x <- iris[1:100, 2:4]
cv.lrkde(groups, x)
```

---

kde_da *Kernel discrimininant analysis*

Description

Kernel discrimininant analysis.

Usage

```
kde_da(xnew, groups, x, h, ncores = 1L)
```

Arguments

xnew	The values of a new vector for which the group is to be predicted.
x	A numeric vector with continuous data.
groups	A vector with the groups (labels) of the data.
h	A vector with one or more values for the bandwidth parameter of the Gaussian kernel.
ncores	The number of cores to use.

Details

This is described in Chapter 8.2 of Hansen (2019). The idea is to inimise the sum of squares of the residuals under the constraint $R^\top \boldsymbol{\beta} = c$. As mentioned above, be careful with the input you give in the x matrix and the R vector. The cls() function performs a single regression model, whereas the mcls() function performs a regression for each column of y. Each regression is independent of the others.

Value

A list including:

be	A numerical matrix with the constrained beta coefficients.
mse	A numerical vector with the mean squared error.

Author(s)

Michail Tsagris and Nikolas Kontemeniotis.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Nikolas Kontemeniotis <kontemeniotisn@gmail.com>.

References

Wand M.P. and Jones M.C. (1995). Kernel smoothing. CRC Press.

See Also

cv.kdeda

Examples

```
x <- iris[, 4]
groups <- as.numeric(iris[, 5])
a <- kde_da(x, groups, x, h = c(0.1, 0.2))
```

---

lr.kde *Kernel regression with a binary response*

Description

Kernel regression with a binary response.

Usage

```
lr.kde(xnew, groups, x, h, ncores = 1)
lr.kde2(groups, x, h)
lr.kde3(groups, x, h = seq(0.01, 2, by = 0.01), ncores = 1)
```

Arguments

xnew	A matrix with the new predictor variables whose compositions are to be predicted.
groups	A numerical vector or a matrix with the binary response value (0s and 1s).
x	A matrix with the available predictor variables.
h	The bandwidth value(s) to consider.
ncores	The number of cores to use. If greater than 1, parallel computing will take place. It is advisable to use it if you have many observations and or many variables, otherwise it will slow down the process. The default is 1, meaning that code is executed serially.

Details

The Nadaraya-Watson estimator regression is applied. The function lr.kde2() chooses the bandwidth value that maximizes the AUC, using the command optimize(). The function lr.kde3() does a similar job but for given bandwidth values and plots the results.

Value

For the lr.kde() a list including:

`prob`	The estimated probabilities of belonging to class 1.
`cl`	A vector with the estimated classes.

Author(s)

Michail Tsagris and Christos Adam.

References

Wand M. P. and Jones M. C. (1994). Kernel smoothing. CRC press.

See Also

`cv.lrkde`

Examples

```
groups <- as.numeric( iris[1:100, 5] ) - 1
x <- iris[1:100, 2:4]
est <- lr.kde(x, groups, x, h = c(0.1, 0.2) )
```

Index

cv.kdeda, 2, *6*
cv.lrkde, 4, *7*

kde_da, *3*, 5
kernda-package, 2

lr.kde, *3*, *5*, 6
lr.kde2 *(*lr.kde*)*, 6
lr.kde3 *(*lr.kde*)*, 6