Package ‘llrRcpp’

May 3, 2021

Type Package

Title Local linear regression

Version 1.0

Date 2020-10-26

Author chengwei

Maintainer chengwei <e0002969@u.nus.edu>

Description This package provides an implementation of multivariate local linear regression using kd-tree implementation. Both an exact and approxiate algorithm for using the kd-tree are present. The package also provides a binned method for the implementation of up to 2d.

License GPL (>= 2)

Encoding UTF-8

Imports Rcpp, methods, metaheuristicOpt

LinkingTo Rcpp, RcppEigen

RoxygenNote 7.1.1

Suggests rmarkdown,
knitr,
testthat

VignetteBuilder knitr

R topics documented:

autoloocv.llr 2
bin 3
cv.llr 4
get_num_procs 5
llr 5
loocv.llr 7
plot.llr 8
set_num_threads 9

Index **10**

autoloocv.llr *Auto selection of bandwidth using metaheuristic algorithms*

Description

Auto selection of bandwidth using metaheuristic algorithms

Usage

```
autoloocv.llr(
  x,
  y,
  weight,
  kernel = "epanechnikov",
  bw_range = c(0.01, 0.5),
  approx = FALSE,
  epsilon = 0.05,
  control = list(numPopulation = 15, maxIter = 75, Vmax = 2, ci = 1.49445, cg =
    1.49445, w = 0.729),
  algorithm = "PSO",
  seed = 1
)
```

Arguments

`x`	a numeric vector or matrix of x data
`y`	a numeric vector of y data corresponding to `x` data
`weight`	a numeric vector of `length(x)` for weight of each data point
`kernel`	kernel type used; supported are ’epanechnikov’, "rectangular", "triangular", "quartic", "triweight", "tricube", "cosine", "gauss", "logistic". "sigmoid" and "silverman".
`bw_range`	range of bandwidth for the metaheuristic algorithm to search
`approx`	boolean flag: if `true` kdtree approximation is used.
`epsilon`	margin of error allowed for llr approximation using kdtree. Only used when `approx` = `TRUE`.
`control`	control parameter from R package metaheuristicOpt
`algorithm`	algorithm parameter from R pacakage metaheuristicOpt
`seed`	seed to use for algorithm

Examples

```
## Not run:
n <- 1000
x <- runif(n, 0 , 1)
y <- sin(x) + rnorm(n, 0, 0.2)
w <- rep(1/n , n)
## Optimization using Grasshopper Optimisation Algorithm
algorithm <- 'GOA'
control <- list(numPopulation = 10, maxIter = 100)
```

```
h <- autoloocv.llr(x, y, w, control = control , algorithm = algorithm)

## End(Not run)
```

---

bin *Splitting the data into bins*

Description

Splitting the data into bins

Usage

```
bin(x, y, bins = 400, weight)
```

Arguments

`x`	numeric vector or matrix of 2 dimensons of x data
`y`	numeric vector of y data with length corresponding to x
`bins`	number of bins to split the data
`weight`	numeric vector corresponding to weight of each observation

Value

returns an S3 object of the class ’bin’ containing

x a numeric vector of x data of size `bins`

y a numeric vector of y data corresponding to ’bin’ class `x`

weight weight corresponding to ’bin’ class `x`

Examples

```
n <- 1000
x <- seq(0,10,length.out = n)
x1 <- rnorm(n,0,0.2)
y <- sin(x) + x1
w <- rep(1/n, n)
binned <- bin(x,y,bins=400, w)
```

cv.llr *Bandwidth selection using k-fold cross validation*

Description

Bandwidth selection using k-fold cross validation

Usage

```
cv.llr(
  x,
  y,
  weight,
  kernel = "epanechnikov",
  bandwidth,
  kdtree = FALSE,
  approx = FALSE,
  epsilon = 0.05,
  N_min = 1,
  k = 5
)
```

Arguments

`x` a numeric vector or matrix of x data

`y` a numeric vector of y data corresponding to x data

`weight` a numeric vector of `length(x)` for weight of each data point.

`kernel` kernel type used; supported are 'epanechnikov', "rectangular", "triangular", "quartic", "triweight", "tricube", "cosine", "gauss", "logistic". "sigmoid" and "silverman".

`bandwidth` a numeric vector or matrix of bandwidth considered for selection.

`kdtree` boolean flag: If `TRUE`, a kdtree is used for computation of local linear regression.

`approx` boolean flag: if `TRUE` kdtree approximation is used.

`epsilon` margin of error allowed for llr approximation using kdtree. Only used when `kdtree` = `TRUE` and `approx` = `TRUE`.

`N_min` minimum number of points stored in the kd-tree. Only used when `kdtree` = `TRUE` and `approx` = TRUE.

`k` number of folds used for cross validation

Value

returns a single numeric value or vector of `bandwidth` that gives the smallest mean square error from cross validation.

Examples

```
n <- 1000
x <- seq(0, 10, length.out = n)
x1 <- rnorm(n, 0, 0.2)
y <- sin(x) + x1
w <- rep(1 / n, n)
binned <- bin(x, y, bins = 400, w)
bandwidth <- seq(0.02, 0.3, by = 0.02)
## Bandwidth selection of binned data
h_bin <- cv.llr(binned$x, binned$y, binned$weight, bandwidth = bandwidth)
## Bandwidth selection of exact local linear regression
h_exact <- cv.llr(x, y, w, bandwidth = bandwidth)
## Bandwidth selection of exact local linear regression with kdtree
h_kdexact <- cv.llr(x, y, w, kdtree = TRUE, approx = FALSE, bandwidth = bandwidth)
## Bandwidth selection of approx local linear regression with kdtree
h_kdapprox <- cv.llr(x, y, w, kdtree = TRUE, approx = TRUE, bandwidth = bandwidth)
```

get_num_procs *Get number of processors*

Description

Get number of processors

Usage

```
get_num_procs()
```

Value

number of processors

llr *Local linear regression (Regular and binned version)*

Description

Local linear regression (Regular and binned version)

Usage

```
llr(x, ...)

## Default S3 method:
llr(
  x,
  y,
  xpred,
  kernel = "epanechnikov",
  bandwidth,
  weight,
  kdtree = FALSE,
  approx = FALSE,
  epsilon = 0.05,
  N_min = 1,
  ...
)

## S3 method for class 'bin'
llr(x, xpred, kernel = "epanechnikov", bandwidth, ...)
```

Arguments

x bin object

... further arguments to be passed

y a numeric vector of y data corresponding to x.

xpred a numeric vector or matrix of same dimension as x. x values to be predicted.

kernel kernel type used; supported are 'epanechnikov', "rectangular", "triangular", "quartic", "triweight", "tricube", "cosine", "gauss", "logistic".

bandwidth a numeric vector or single number of same dimension as x.

weight a numeric vector of length(x) for weight of each data point.

kdtree boolean flag: If TRUE, kdtree is used for computation of local linear regression.

approx boolean flag: If TRUE, kdtree approximation is used . Only used when kdtree = TRUE.

epsilon margin of error allowed for llr approximation using kdtree. Only used when both kdtree = TRUE and approx = TRUE.

N_min minimum number of points stored in the kd-tree. Only used when both kdtree = TRUE and approx = TRUE. Currently not in use.

Value

returns a S3 object of class "llr" containing

x sorted numeric vector or matrix of xpred

y estimated values corresponding to 'llr' class x

Examples

```
## Not run:
n <- 1000
x <- seq(0, 10, length.out = n)
x1 <- rnorm(n, 0, 0.2)
y <- sin(x) + x1
w <- rep(1 / n, n)
binned <- bin(x, y, bins = 400, w)
## local linear regression for exact without kdtree
llr_exact <- llr(x, y, x, bandwidth = 0.2, weight = w)
## local linear regression for kdtree exact
llr_kdexact <- llr(x, y, x, bandwidth = 0.2, weight = w, kdtree = TRUE)
## local linear regression for kdtree approximation
llr_kdapprox <- llr(x, y, x, bandwidth = 0.2, weight = w, kdtree = TRUE, approx = TRUE)
```

```
## local linear regression for data after binning.
llr_bin <- llr(binned, x, bandwidth = 0.2)

## End(Not run)
```

loocv.llr *Bandwidth selection using leave one out cross validation*

Description

Bandwidth selection using leave one out cross validation

Usage

```
loocv.llr(
  x,
  y,
  weight,
  kernel = "epanechnikov",
  approx = FALSE,
  epsilon = 0.05,
  N_min = 1,
  bandwidth
)
```

Arguments

`x` a numeric vector or matrix of x data

`y` a numeric vector of y data corresponding to `x` data

`weight` a numeric vector of `length(x)` for weight of each data point.

`kernel` kernel type used; supported are 'epanechnikov', "rectangular", "triangular", "quartic", "triweight", "tricube", "cosine", "gauss", "logistic". "sigmoid" and "silverman".

`approx` boolean flag: if `true` kdtree approximation is used.

`epsilon` margin of error allowed for llr approximation using kdtree. Only used when `approx` = `TRUE`.

`N_min` minimum number of points stored in the kd-tree. Only used when `approx` = TRUE.

`bandwidth` a numeric vector or matrix of bandwidth considered for selection.

Value

returns a single or numeric vector of `bandwidth` that gives the smallest mean square error.

Examples

```
## Not run:
n <- 1000
x <- seq(0, 10, length.out = n)
x1 <- rnorm(n, 0, 0.2)
y <- sin(x) + x1
w <- rep(1 / n, n)
bandwidth <- seq(0.02, 0.4, by = 0.01)
binned <- bin(x, y, bins = 400, w)
## Bandwidth selection of binned data
h_bin <- gcv.llr(binned$x, binned$y, binned$weight, bandwidth = bandwidth)
## Bandwidth selection of exact local linear regression
h_exact <- gcv.llr(x, y, w, bandwidth = bandwidth)
## Bandwidth selection of approx local linear regression with kdtree
h_kdapprox <- gcv.llr(x, y, w, approx = TRUE, bandwidth = bandwidth)

## End(Not run)
```

---

plot.llr *Plot class 'llr' object*

Description

Plot class 'llr' object

Plot 'bin' object

Usage

```
## S3 method for class 'llr'
plot(x, xorig, yorig, ...)

## S3 method for class 'bin'
plot(x, ...)
```

Arguments

x	a object of class 'bin'
xorig	initial numeric vector x
yorig	initial numeric vector y
...	for consistency

`set_num_threads` *Set number of threads to use*

Description

Set number of threads to use

Usage

```
set_num_threads(threads)
```

Arguments

`threads`	number of threads to use

Value

number of threads set

Index

autoloocv.llr, 2

bin, 3

cv.llr, 4

get_num_procs, 5

llr, 5
llr-class *(llr)*, 5
llr.bin *(llr)*, 5
llr.default *(llr)*, 5
loocv.llr, 7

plot.bin *(plot.llr)*, 8
plot.llr, 8

set_num_threads, 9