Package ‘rmi’

October 14, 2022

Title  Mutual Information Estimators
Version 0.1.1
Author  Isaac Michaud [cre, aut]
Maintainer Isaac Michaud <ijmichau@ncsu.edu>
Description Provides mutual information estimators based on k-nearest neighbor estima-
License GPL-3
Encoding UTF-8
LazyData true
RoxygenNote 6.1.0
LinkingTo Rcpp, RcppArmadillo, BH
Imports Rcpp, stats, graphics
Suggests testthat, parallel, tgp
NeedsCompilation yes
Repository CRAN
Date/Publication 2018-08-02 11:40:06 UTC

R topics documented:

<table>
<thead>
<tr>
<th>estimate_mse</th>
<th>knn_mi</th>
<th>lnn_entropy</th>
<th>lnn_mi</th>
<th>nearest_neighbors</th>
<th>optimize_mse</th>
<th>rmi</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
</tr>
</tbody>
</table>

Index 9
estimate_mse

Estimate MSE of LNC Estimator

Description

Computes the MSE of the Local Non-Uniformity Correct (LNC) KSG estimator for a given value of the tuning parameter alpha, dimension, neighborhood order, and sample size.

Usage

```r
estimate_mse(k = 5, alpha = 0, d = 2, rho = 0, N = 1000,
M = 100, cluster = NULL)
```

Arguments

- `k`: Neighborhood order.
- `alpha`: Non-uniformity threshold (see details).
- `d`: Dimension.
- `rho`: Reference correlation (see details).
- `N`: Sample size.
- `M`: Number of replications.
- `cluster`: A parallel cluster object.

Details

The parameter `alpha` controls the threshold for the application of the non-uniformity correction to a particular point’s neighborhood. Roughly, `alpha` is the ratio of the PCA aligned neighborhood volume to the rectangular aligned neighborhood volume below which indicates non-uniformity and the correction is applied.

If `alpha < 0` then a log scale is assumed; otherwise [0,1] scale is used. `alpha > 1` are unacceptable values. A value of `alpha = 0` forces no correction and LNC reverts to the KSG estimator.

The reference distribution that is assumed is a mean-zero multivariate normal distribution with a compound-symmetric covariance. The covariance matrix has a single correlation parameter supplied by `rho`.

Examples

```r
estimate_mse(N = 100, M = 2)
```
**knn_mi**  

### kNN Mutual Information Estimators

**Description**

Computes mutual information based on the distribution of nearest neighborhood distances. Method available are KSG1 and KSG2 as described by Kraskov, et. al (2004) and the Local Non-Uniformity Corrected (LNC) KSG as described by Gao, et. al (2015). The LNC method is based on KSG2 but with PCA volume corrections to adjust for observed non-uniformity of the local neighborhood of each point in the sample.

**Usage**

```r
knn_mi(data, splits, options)
```

**Arguments**

- **data**: Matrix of sample observations, each row is an observation.
- **splits**: A vector that describes which sets of columns in `data` to compute the mutual information between. For example, to compute mutual information between two variables use `splits = c(1,1)`. To compute redundancy among multiple random variables use `splits = rep(1, ncol(data))`. To compute the mutual information between two random vector list the dimensions of each vector.
- **options**: A list that specifies the estimator and its necessary parameters (see details).

**Details**

Current available methods are LNC, KSG1 and KSG2.

For KSG1 use: `options = list(method = "KSG1", k = 5)`

For KSG2 use: `options = list(method = "KSG2", k = 5)`

For LNC use: `options = list(method = "LNC", k = 10, alpha = 0.65), order needed k > ncol(data)`.

**Author**

Isaac Michaud, North Carolina State University, <ijmichau@ncsu.edu>

**References**


### Examples

```r
set.seed(123)
x <- rnorm(1000)
y <- x + rnorm(1000)
knn_mi(cbind(x, y), c(1, 1), options = list(method = "KSG2", k = 6))
```

```r
set.seed(123)
x <- rnorm(1000)
y <- 100*x + rnorm(1000)
knn_mi(cbind(x, y), c(1, 1), options = list(method = "LNC", alpha = 0.65, k = 10))
# approximate analytic value of mutual information
-0.5*log(1-cor(x, y)^2)
```

```r
z <- rnorm(1000)
# redundancy I(x;y;z) is approximately the same as I(x;y)
knn_mi(cbind(x, y, z), c(1, 1, 1), options = list(method = "LNC", alpha = c(0.5, 0, 0, 0), k = 10))
# mutual information I((x,y);z) is approximately 0
knn_mi(cbind(x, y, z), c(2, 1), options = list(method = "LNC", alpha = c(0.5, 0.65, 0), k = 10))
```

---

**lnn_entropy**  
*Local Nearest Neighbor (LNN) Entropy Estimator*

---

### Description

Local Nearest Neighbor entropy estimator using Gaussian kernel and kNN selected bandwidth. Entropy is estimated by taking a Monte Carlo estimate using local kernel density estimate of the negative-log density.

### Usage

```r
lnn_entropy(data, k = 5, tr = 30, bw = NULL)
```

### Arguments

- **data**: Matrix of sample observations, each row is an observation.
- **k**: Order of the local kNN bandwidth selection.
- **tr**: Order of truncation (number of neighbors to include in entropy).
- **bw**: Bandwidth (optional) manually fix bandwidth instead of using local kNN bandwidth selection.

### References


Examples

```r
set.seed(123)
x <- rnorm(1000)
print(lnn_entropy(x))
# analytic entropy
print(0.5*log(2*pi*exp(1)))
```

```
lxn mi

Local Nearest Neighbor (LNN) MI Estimator

Description

Local Nearest Neighbor (LNN) mutual information estimator by Gao et al. 2017. This estimator uses the LNN entropy (lnn_entropy) estimator into the mutual information identity.

Usage

```
lxn mi(data, splits, k = 5, tr = 30)
```

Arguments

- `data`: Matrix of sample observations, each row is an observation.
- `splits`: A vector that describes which sets of columns in `data` to compute the mutual information between. For example, to compute mutual information between two variables use `splits = c(1,1)`. To compute redundancy among multiple random variables use `splits = rep(1, ncol(data))`. To compute the mutual information between two random vector list the dimensions of each vector.
- `k`: Order of the local kNN bandwidth selection.
- `tr`: Order of truncation (number of neighbors to include in the local density estimation).

References


Examples

```r
set.seed(123)
x <- rnorm(1000)
y <- x + rnorm(1000)
lnn_mi(cbind(x,y),c(1,1))
```
nearest_neighbors  

*Compute Nearest Neighbors*

**Description**

Computes the nearest neighbor distances and indices of a sample using the infinite norm.

**Usage**

```r
nearest_neighbors(data, k)
```

**Arguments**

- `data`  
  Matrix of sample observations, each row is an observation.
- `k`  
  Neighborhood order.

**Details**

Nearest neighbors are computed using the brute-force method.

**Value**

List of distances and indices of the k-nearest neighbors of each point in `data`.

**Examples**

```r
X <- cbind(1:10)
nearest_neighbors(X, 3)
set.seed(123)
X <- cbind(runif(100), runif(100))
plot(X, pch=20)
points(X[3,1], X[3,2], col = 'blue', pch=19, cex=1.5)
n <- nearest_neighbors(X, 5)
a = X[nn$nn_inds[3,-1],1]
b = X[nn$nn_inds[3,-1],2]
points(a, b, col = 'red', pch=19, cex=1.5)
```

**Optimize MSE of LNC Estimator**

**Description**

Gaussian process (GP) optimization is used to minimize the MSE of the LNC estimator with respect to the non-uniformity threshold parameter \( \alpha \). A normal distribution with compound-symmetric covariance is used as a reference distribution to optimize the MSE of LNC with respect to.

**Usage**

```r
optimize_mse(rho, N, M, d, k, lower = -10, upper = -1e-10, num_iter = 10, init_size = 20, cluster = NULL, verbose = TRUE)
```

**Arguments**

- `rho`: Reference correlation.
- `N`: Sample size.
- `M`: Number of replications.
- `d`: Dimension.
- `k`: Neighborhood order.
- `lower`: Lower bound for optimization.
- `upper`: Upper bound for optimization.
- `num_iter`: Number of iterations of GP optimization.
- `init_size`: Number of initial evaluation to estimating GP.
- `cluster`: A parallel cluster object.
- `verbose`: If TRUE then print runtime diagnostic output.

**Details**

The package `tgp` is used to fit a treed-GP to the MSE estimates of LNC. A treed-GP is used because the MSE of LNC with respect to \( \alpha \) exhibits clear non-stationarity. A treed-GP is able to identify the function’s different correlation lengths which improves optimization.
Description

The `rmi` package offers a collection of mutual information estimators based on k-Nearest Neighbor and local density estimators. Currently, `rmi` provides the Kraskov et al. algorithm (KSG) 1 and 2, Local Non-uniformity Corrected (LNC) KSG, and the Local Nearest Neighbor (LNN) estimator. More estimators and examples will be incorporated in the future.

References


Author(s)

Isaac Michaud
Index

estimate_mse, 2
knn_mi, 3
1nn_entropy, 4
1nn_mi, 5
nearest_neighbors, 6
optimize_mse, 7
rmi, 8
rmi-package (rmi), 8