Package 'uHMM'

April 22, 2016

Type Package

Title Construct an Unsupervised Hidden Markov Model

Version 1.0

Date 2016-03-18

Author Emilie POISSON-CAILLAULT [aut], Paul TERNYNCK [aut, cre]

Maintainer Paul TERNYNCK <ternynck@lisic.univ-littoral.fr>

Description Construct a Hidden Markov Model with states learnt by unsupervised classification.

License GPL (>= 2)

RoxygenNote 5.0.1

LazyData TRUE

Depends R (>= 3.0.0), stats, grDevices

Imports tcltk, tcltk2, tkrplot, HMM, clValid, class, cluster, FactoMineR, corrplot, chron

NeedsCompilation no

Repository CRAN

Date/Publication 2016-04-22 16:25:47

R topics documented:

`uHMM-package` *Construct an unsupervised Hidden Markov Model*

Description

This package proposes an interface to detect usual or extreme events in a dataset and to characterize their dynamic, by building an unsupervised Hidden Markov Model (use `uHMMinterface` to launch the interface). Functions can also be used out of the interface to build an uHMM.

Details

Package:	uHMM
Version:	1.0
Date:	2016-04-13
Depends:	R (>= 3.0.0), stats, grDevices
Import:	tcltk, tcltk2, tkrplot, HMM, clValid, class, cluster, FactoMineR, corrplot, chron
License:	GPL (>=2)
LazyLoad:	yes

Author(s)

Emilie Poisson-Caillault and Paul Ternynck

Maintainer: <emilie.caillault@lisic.univ-littoral.fr>

Source

Rousseeuw, Kevin, et al. "Hybrid hidden Markov model for marine environment monitoring." Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of 8.1 (2015): 204-213.

`computeGap` *Compute gap between eigenvalues of a similarity matrix*

Description

Find the highest gap between eigenvalues of a similarity matrix. The 2 first eigenvalues are considered as equal to each other (the gap between the 2 first eigenvalues is set to 0).

Usage

```
computeGap(similarity, Gmax)
```

Arguments

similarity	a similarity matrix.
Gmax	the maximum gap value allowed (only the first Gmax eigenvalues will be taken into account).

Value

The function returns a list containing the following components:

gap	a vector indicating the gap between similarity matrix eigenvalues (the gap between the 2 first eigenvalues is set to 0)
Kmax	an integer indicating the index of the highest gap (the highest gap is between the Kmax-th and the (Kmax+1)-th eigenvalues)

Examples

```
x <- rbind(matrix(rnorm(50, mean = 0, sd = 0.3), ncol = 2),
           matrix(rnorm(50, mean = 2, sd = 0.3), ncol = 2),
           matrix(rnorm(50, mean = 4, sd = 0.3), ncol = 2))

similarity<-ZPGaussianSimilarity(x,7)
Gap<-computeGap(similarity,10)
plot(1:length(Gap$gap),Gap$gap,type="h",
main=paste("Gap criteria =",Gap$K),ylab="gap value",xlab="eigenvalues")



x=(runif(1000)*4)-2;y=(runif(1000)*4)-2
keep<-which((x**2+y**2<0.5)|(x**2+y**2>1.5**2 & x**2+y**2<2**2 ))
data<-data.frame(x,y)[keep,]
plot(data)

similarity<-ZPGaussianSimilarity(data,1)
Gap<-computeGap(similarity,10)
plot(1:length(Gap$gap),Gap$gap,type="h",
main=paste("Gap criteria =",Gap$K),ylab="gap value",xlab="eigenvalues")
```

---

cutCalculation *cutCalculation function*

Description

Compute intra and inter-cluster cuts from the similarity matrix of a dataset.

Usage

```
cutCalculation(similarity, label, K)
```

Arguments

similarity	a similarity matrix.
label	vector of cluster sequencing.
K	number of clusters. (= nbCluster CALCULE DANS LA FONCTION ???)

Details

intra cluster cut :

$$Cut(g_k, g_l) = \sum_{i=1, x(i) \in g_k}^{N_p} \sum_{j=1, x(j) \in g_l}^{N_p} w(x(i), x(j))$$

Value

The function returns a list containing:

mncut	the inter-cluster cut, i.e. K-sum(ratioCutVol).
ratioCutVol	vector of intra-cluster cuts, one component per cluster.

Examples

```
x<-rbind(matrix(runif(100),ncol=2),matrix(runif(100)+2,ncol=2),matrix(runif(20)*3,ncol=2))
similarity<-ZPGaussianSimilarity(x,7)%*%t(ZPGaussianSimilarity(x,7))
km<-kmeans(similarity,2)
label<-km$cluster
plot(x,col=km$cluster)
cutCalculation(similarity,label,length(unique(label)))
```

---

emissionMatrix *Emission matrix estimation*

Description

This function estimates the emission matrix of a Hidden Markov Model from vectors of state and symbol sequencing.

Usage

```
emissionMatrix(states, symbols)
```

Arguments

states	a numeric vector of state sequencing.
symbols	a numeric vector of symbol sequencing.

Value

Estimated emission matrix.

See Also

HMMparams

Examples

```
states<-c(1,1,3,2,1,2,1,3)
symbols<-c(4,1,3,1,4,4,4,2)
B<-emissionMatrix(states,symbols)
B
```

FastSpectralNJW *Jordan Fast Spectral Algorithm*

Description

Perform the Jordan spectral algorithm for large databases. Data are sampled, using K-means with Elbow criteria, before being classified.

Usage

```
FastSpectralNJW(data, nK = NULL, Kech = 2000, StopCriteriaElbow = 0.97,
  neighbours = 7, method = "", nb.iter = 10, uHMMinterface = FALSE,
  console = NULL, tm = NULL)
```

Arguments

data	numeric matrix or dataframe.
nK	number of clusters desired. If NULL, optimal number of clusters will be computed using gap criteria.
Kech	maximum number of representative points in sampled data.
StopCriteriaElbow	maximum (minimum ?) de variance expliquees des points representatifs souhaite.
neighbours	number of neighbours considered for the computation of local scale parameters.
method	string specifying the spectral classification method desired, either "PAM" (for spectral kmedoids) or "" (for "spectral kmeans").
nb.iter	number of iterations.
uHMMinterface	logical indicating whether the function is used via the uHMMinterface.
console	frame of the uHMM interface in which messages should be displayed (only if uHMMinterface=TRUE).
tm	a one row dataframe containing text to display in the uHMMinterface (only if uHMMinterface=TRUE).

Details

Algorithme de Jordan pour un grand jeu de donnees : echantillonage puis spectral

Value

The function returns a list containing:

sim	similarity matrix of representative points, multiplied by its transpose (ZPGaussianSimilarity).
label	vector of cluster sequencing.
gap	number of clusters.
labelElbow	vector of prototype sequencing.
vpK	matrix containing, in columns, the K first normalised eigen vectors of the data similarity matrix.
valp	vector containing the K first eigen values of the data similarity matrix.
echantillons	matrix of prototypes coordinates.
label.echantillons	vector containing the cluster of each prototype.
numSymbole	vector containing the nearest prototype of each data item.

See Also

KmeansAutoElbow ZPGaussianSimilarity knn silhouette dunn connectivity dist

Examples

```
x=(runif(1000)*4)-2;y=(runif(1000)*4)-2
keep<-which((x**2+y**2<0.5)|(x**2+y**2>1.5**2 & x**2+y**2<2**2 ))
data<-data.frame(x,y)[keep,]

cl<-FastSpectralNJW(data,2)
plot(data,col=cl$label)
```

HMMparams *Hidden Markov Model parameter estimation*

Description

This function is used by the uHMMinterface to estimate parameters of a Hidden Markov Model.

Usage

```
HMMparams(stateSeq, symbolSeq)
```

Arguments

stateSeq	a numeric vector of state sequencing.
symbolSeq	a numeric vector of symbol sequencing.

Value

HMMparams returns a list containing :

`trans`	The transition matrix.
`emis`	The emission matrix.
`startProb`	The vector of initial probability distribution (initial states are supposed equiprobable).

See Also

`transitionMatrix` `emissionMatrix`

`KmeansAutoElbow` *KmeansAutoElbow function*

Description

KmeansAutoElbow performs k-means clustering on a dataframe with selection of optimal number of clusters using elbow criteria.

Usage

```
KmeansAutoElbow(features, Kmax, StopCriteria = 0.99, graph = FALSE)
```

Arguments

`features`	dataframe or matrix of raw data.
`Kmax`	maximum number of clusters allowed.
`StopCriteria`	elbow method cumulative explained variance > criteria to stop K-search. (???)
`graph`	boolean, if TRUE figures are plotted.

Details

KmeansAutoElbow returns partition and K number of groups according to kmeans clustering and Elbow method

Value

The function returns a list containing the following components:

`K`	number of clusters in data according to explained variance and kmeans algorithm.
`res.kmeans`	an object of class "kmeans" (see `kmeans`) containing classification results.

See Also

`kmeans`

Examples

```
x <- rbind(matrix(rnorm(300, mean = 0, sd = 0.3), ncol = 2),
           matrix(rnorm(100, mean = 2, sd = 0.3), ncol = 2),
           matrix(rnorm(100, mean = 4, sd = 0.3), ncol = 2))
colnames(x) <- c("x", "y")
km<-KmeansAutoElbow(x,round(dim(x)/25,0)[1],StopCriteria=0.99,graph=TRUE)
plot(x,col=km$res.kmeans$cluster)
points(km$res.kmeans$centers, col = 1:km$K, pch = 16)
```

KpartitionNJW *KpartitionNJW function*

Description

Perform spectral classification on the similarity matrix of a dataset (Ng et al. (2001) algorithm), using kmeans algorithm on data projected in the space of its K first eigen vectors.

Usage

```
KpartitionNJW(similarity, K)
```

Arguments

similarity	matrix of similarity.
K	number of clusters.

Value

The function returns a list containing:

label	vector of cluster sequencing.
centres	matrix of cluster centers in the space of the K first normalised eigen vectors.
vecteursPropresProjK	matrix containing, in columns, the K first normalised eigen vectors of the similarity matrix.
valeursPropresK	vector containing the K first eigen values of the similarity matrix.
vecteursPropres	matrix containing, in columns, eigen vectors of the similarity matrix.
valeursPropres	vector containing eigen values of the similarity matrix.
inertieZ	vector of within-cluster sum of squares, one component per cluster.

References

Ng Andrew, Y., M. I. Jordan, and Y. Weiss. "On spectral clustering: analysis and an algorithm [C]." Advances in Neural Information Processing Systems (2001).

Examples

```
#####
x <- rbind(matrix(rnorm(100, mean = 0, sd = 0.3), ncol = 2),
           matrix(rnorm(100, mean = 2, sd = 0.3), ncol = 2),
           matrix(rnorm(100, mean = 4, sd = 0.3), ncol = 2))

similarity<-ZPGaussianSimilarity(x,7)
similarity=similarity%*%t(similarity)
sp<-KpartitionNJW(similarity,3)
plot(x,col=sp$label)

#####
x <- rbind(data.frame(x=1:100+(runif(100)-0.5)*2,y=runif(100)/5),
           data.frame(x=1:100+(runif(100)-0.5)*2,y=runif(100)/5+1),
           data.frame(x=1:100+(runif(100)-0.5)*2,y=runif(100)/5+2))

similarity<-ZPGaussianSimilarity(x,7)
similarity=similarity%*%t(similarity)
sp<-KpartitionNJW(similarity,3)
plot(x,col=sp$label)

#####
x=(runif(1000)*4)-2;y=(runif(1000)*4)-2
keep<-which((x**2+y**2<0.5)|(x**2+y**2>1.5**2 & x**2+y**2<2**2 ))
data<-data.frame(x,y)[keep,]

similarity=ZPGaussianSimilarity(data, 7)
similarity=similarity%*%t(similarity)
sp<-KpartitionNJW(similarity,2)

plot(data,col=sp$label)
```

`MarelCarnot` *MarelCarnot dataset*

Description

The MarelCarnot data set gives the measurements of 14 physico-chemical and biological parameters performed by the Marel-Carnot station (Boulogne-sur-Mer, France), at high frequency resolution.

Usage

```
MarelCarnot
```

Format

A data frame with 131487 rows and 16 columns.

Details

Dates	date of measurement	(YYYY:MM:DD)
Hours	time of measurement	(HH:MM:SS)
C_NI1	nitrate concentration	(in μmol/L)
C_PO1	phosphate concentration	(in μmol/L)
C_O21	corrected dissolved oxygen	(in mg/L)
C_SI1	silicate concentration	(in μmol/L)
CSAL1	salinity	(in PSU)
CSAT1	oxygen saturation	(in %)
ETCO1	air temperature	(in degrees Celsius)
E_LU1	P.A.R	(in μmol of photons/s/m2)
E_O21	uncorrected dissolved oxygen	(in mg/L)
E_PH1	pH	
E_TU1	turbidity	(in NTU)
ECHL1	fluorescence	(in FFU)
E__TA	water temperature	(in degrees Celsius)
XMAHH	water level	(in m)

Source

Lefebvre Alain (2015). MAREL Carnot data and metadata from Coriolis Data Centre. SEANOE. http://doi.org/10.17882/39754

`selfKNN` *Self KNN*

Description

This function performs the k-Nearest Neighbour algorithm without class estimation, but only computation of distances and neighbours.

Usage

```
selfKNN(train, K = 1)
```

Arguments

`train`	numeric matrix or data frame.
`K`	number of neighbours considered.

Value

The function returns a list with the following components:

D	matrix of squared root of the distances between observations and their nearest neighbours.
idx	Index of K nearest neighbours of each observation.

Examples

```
x<-matrix(runif(10),ncol=2)
plot(x,pch=c("1","2","3","4","5"))
selfKNN(x,K=4)
```

---

spectralPamClusteringNg

spectralPamClusteringNg function

Description

Perform spectral classification on the similarity matrix of a dataset, using pam algorithm (a more robust version of K-means) on projected data.

Usage

```
spectralPamClusteringNg(similarity, K)
```

Arguments

similarity	matrix of similarity
K	number of clusters

Value

The function returns a list containing:

label	vector of cluster sequencing.
centres	matrix of cluster medoids (similar in concept to means, but medoids are members of the dataset) in the space of the K first normalised eigen vectors.
id.med	integer vector of indices giving the medoid observation numbers.
vecteursPropresProjK	matrix containing, in columns, the K first normalised eigen vectors of the similarity matrix.
valeursPropresK	vector containing the K first eigen values of the similarity matrix.
vecteursPropres	matrix containing, in columns, eigen vectors of the similarity matrix.

valeursPropres	vector containing eigen values of the similarity matrix.
cluster.info	matrix, each row gives numerical information for one cluster. These are the cardinality of the cluster (number of observations), the maximal and average dissimilarity between the observations in the cluster and the cluster's medoid, the diameter of the cluster (maximal dissimilarity between two observations of the cluster), and the separation of the cluster (minimal dissimilarity between an observation of the cluster and an observation of another cluster).

References

Ng Andrew, Y., M. I. Jordan, and Y. Weiss. "On spectral clustering: analysis and an algorithm [C]." Advances in Neural Information Processing Systems (2001).

See Also

pam

---

transitionMatrix *Transition matrix estimation*

Description

This function estimates the transition matrix of a (Hidden) Markov Model from a vector of state sequencing.

Usage

```
transitionMatrix(states)
```

Arguments

states	a numeric vector of state sequencing.

Value

Estimated transition matrix.

See Also

HMMparams

Examples

```
states<-c(1,1,3,2,1,2,1,3)
A<-transitionMatrix(states)
A
```

uHMMinterface *Graphical Interface to Build an uHMM*

Description

A user-friendly interface to detect usual or extreme events in a dataset and to characterize their dynamic, by building an unsupervised Hidden Markov Model.

Usage

```
uHMMinterface(uHMMenv = NULL)
```

Arguments

uHMMenv	an environment in which data and results will be stored. If NULL, a local environment will be created.

Value

Results are saved in the directory chosen by the user.

References

Rousseeuw, Kevin, et al. "Hybrid hidden Markov model for marine environment monitoring." Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of 8.1 (2015): 204-213.

ZPGaussianSimilarity *Similarity matrix with local scale parameter*

Description

Compute and return the similarity matrix of a data frame using gaussian kernel with a local scale parameter for each data point, rather than a unique scale parameter.

Usage

```
ZPGaussianSimilarity(data, K)
```

Arguments

data	a matrix or numeric data frame.
K	number of neighbours considered to compute scale parameters.

Value

The matrix of similarity.

References

Zelnik-Manor, Lihi, and Pietro Perona. "Self-tuning spectral clustering." Advances in neural information processing systems. 2004.

Examples

```
x <- rbind(matrix(rnorm(50, mean = 0, sd = 0.3), ncol = 2))
similarity<-ZPGaussianSimilarity(x,7)
```

Index