# Package ‘ClickClust’

October 12, 2022

Version 1.1.5

Date 2016-10-22

Title Model-Based Clustering of Categorical Sequences

Depends R (>= 3.0.0)

LazyLoad yes

LazyData no

Description Clustering categorical sequences by means of finite mixtures with Markov model components is the main utility of ClickClust. The package also allows detecting blocks of equivalent states by forward and backward state selection procedures.

License GPL (>= 2)

Author Volodymyr Melnykov [aut, cre], Rouben Rostamian [ctb, cph] (memory allocation in c)

Maintainer Volodymyr Melnykov <vmelnykov@cba.ua.edu>

NeedsCompilation yes

Repository CRAN

Date/Publication 2016-10-23 00:20:21

### R topics documented:

- ClickClust-package ................................................. 2
- B3 .................................................................................. 3
- C ...................................................................................... 4
- click.backward ............................................................... 5
- click.EM ........................................................................ 7
- click.forward ................................................................. 9
- click.plot ...................................................................... 11
- click.predict ................................................................. 13
- click.read ..................................................................... 15
- click.sim ...................................................................... 17
- click.var ..................................................................... 18
- msnbc323 ..................................................................... 20
- print.object ................................................................. 21
- synth ............................................................................. 22
Description

The package runs finite mixture modeling and model-based clustering for categorical sequences.

Details

<table>
<thead>
<tr>
<th>Package:</th>
<th>ClickClust</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type:</td>
<td>Package</td>
</tr>
<tr>
<td>Version:</td>
<td>1.0</td>
</tr>
<tr>
<td>Date:</td>
<td>2014-04-04</td>
</tr>
<tr>
<td>License:</td>
<td>GPL (&gt;= 2)</td>
</tr>
<tr>
<td>LazyLoad:</td>
<td>no</td>
</tr>
</tbody>
</table>

Function 'click.EM' runs the EM algorithm for finite mixture models with Markov model components.

Author(s)

Volodymyr Melnykov
Maintainer: Volodymyr Melnykov <vmelnykov@cba.ua.edu>

References

Melnykov, V. (2016) Model-Based Biclustering of Clickstream Data, Computational Statistics and Data Analysis, 93, 31-45.

Examples

```r
set.seed(123)
n.seq <- 50
p <- 5
K <- 2
mix.prop <- c(0.3, 0.7)
TP1 <- matrix(c(0.20, 0.10, 0.15, 0.15, 0.40,
```
0.20, 0.20, 0.20, 0.20, 0.20, 0.15, 0.10, 0.20, 0.20, 0.35, 0.15, 0.10, 0.20, 0.20, 0.35, 0.30, 0.30, 0.10, 0.10, 0.20), byrow = TRUE, ncol = p)

TP2 <- matrix(c(0.15, 0.15, 0.20, 0.20, 0.30, 0.20, 0.10, 0.30, 0.30, 0.10, 0.25, 0.20, 0.20, 0.15, 0.15, 0.25, 0.25, 0.20, 0.15, 0.15, 0.25, 0.10, 0.30, 0.20, 0.20, 0.20), byrow = TRUE, ncol = p)

TP <- array(rep(NA, p * p * K), c(p, p, K))
TP[,,1] <- TP1
TP[,,2] <- TP2

# DATA SIMULATION
A <- click.sim(n = n.seq, int = c(10, 50), alpha = mix.prop, gamma = TP)
C <- click.read(A$S)

# EM ALGORITHM
click.EM(X = C$X, K = 2)

---

**Dataset: result of backward state selection**

**Description**

These data demonstrate the result of the backward state selection procedure obtained for the dataset "C".

**Usage**

data(utilityB3)

**Details**

Results of the backward state selection procedure assuming three components are provided for the dataset "C".

**References**

See Also

help(C, package = "ClickClust")

Examples

data(utilityB3)

dev.new(width = 11, height = 11)
click.plot(X = C$X, id = B3$id, colors = c("lightyellow", "red", "darkred"), col.levels = 10)

---

C

Dataset: simulated dataset

Description

This dataset is used to run the backward state selection procedure (results in "B3").

Usage

data(utilityB3)

Details

Original dataset used to illustrate the utility of backward selection.

References

Melnykov, V. (2016) Model-Based Biclustering of Clickstream Data, Computational Statistics and Data Analysis, 93, 31-45.


See Also

help(B3)

Examples

data(utilityB3)

dev.new(width = 11, height = 11)
click.plot(X = C$X, id = B3$id, colors = c("lightyellow", "red", "darkred"), col.levels = 10)
click.backward  

**Backward search for equivalent states**

**Description**

Runs backward search to detect blocks of equivalent states.

**Usage**

```r
click.backward(X, K, eps = 1e-10, r = 100, iter = 5, bic = TRUE, 
               min.gamma = 1e-3, scale.const = 1.0, silent = FALSE)
```

**Arguments**

- `X`: dataset array (p x p x n)
- `K`: number of mixture components
- `eps`: tolerance level
- `r`: number of restarts for initialization
- `iter`: number of iterations for each short EM run
- `bic`: flag indicating whether BIC or AIC is used
- `min.gamma`: lower bound for transition probabilities
- `scale.const`: scaling constant for avoiding numerical issues
- `silent`: output control

**Details**

Runs backward search to detect blocks of equivalent states. States i and j are called equivalent if their behavior expressed in terms of transition probabilities is identical, i.e., the probabilities of leaving i and j to visit another state h are the same as well as the probabilities of coming to i and j from another state h are the same; this condition should hold for all mixture components. Notation: p - number of states, n - sample size, K - number of mixture components, d - number of equivalence blocks.

**Value**

- `z`: matrix of posterior probabilities (n x K)
- `alpha`: vector of mixing proportions (length K)
- `gamma`: array of transition probabilities (d x d x K)
- `states`: detected equivalence blocks (length p)
- `logl`: log likelihood value
- `BIC`: Bayesian Information Criterion
- `AIC`: Akaike Information Criterion
- `id`: classification vector (length n)
References

Melnykov, V. (2016) Model-Based Biclustering of Clickstream Data, Computational Statistics and Data Analysis, 93, 31-45.


See Also

forward.search, click.EM

Examples

```r
set.seed(123)

n.seq <- 50

p <- 5
K <- 2
mix.prop <- c(0.3, 0.7)

TP1 <- matrix(c(0.20, 0.10, 0.15, 0.15, 0.40, 0.20, 0.20, 0.20, 0.20, 0.20, 0.15, 0.10, 0.20, 0.20, 0.35, 0.15, 0.10, 0.20, 0.20, 0.35, 0.30, 0.30, 0.10, 0.10, 0.20), byrow = TRUE, ncol = p)

TP2 <- matrix(c(0.15, 0.15, 0.20, 0.20, 0.30, 0.20, 0.10, 0.30, 0.30, 0.10, 0.25, 0.20, 0.15, 0.15, 0.25, 0.25, 0.20, 0.15, 0.15, 0.25, 0.10, 0.30, 0.20, 0.20, 0.20), byrow = TRUE, ncol = p)

TP <- array(rep(NA, p * p * K), c(p, p, K))
TP[,1] <- TP1
TP[,2] <- TP2

# DATA SIMULATION
A <- click.sim(n = n.seq, int = c(10, 50), alpha = mix.prop, gamma = TP)
B <- click.read(A$S)

# BACKWARD SEARCH
click.backward(X = B$X, K = 2)
```
click.EM

EM algorithm for mixtures of Markov models

Description

Runs the EM algorithm for finite mixture models with Markov model components.

Usage

```r
click.EM(X, y = NULL, K, eps = 1e-10, r = 100, iter = 5, min.beta = 1e-3, 
   min.gamma = 1e-3, scale.const = 1)
```

Arguments

- `X`: dataset array (p x p x n)
- `y`: vector of initial states (length n)
- `K`: number of mixture components
- `eps`: tolerance level
- `r`: number of restarts for initialization
- `iter`: number of iterations for each short EM run
- `min.beta`: lower bound for initial state probabilities
- `min.gamma`: lower bound for transition probabilities
- `scale.const`: scaling constant for avoiding numerical issues

Details

Runs the EM algorithm for finite mixture models with first order Markov model components. The function returns estimated mixing proportions \( \alpha \) and transition probability matrices \( \gamma \). If initial states \( y \) are not provided, initial state probabilities \( \beta \) are not estimated and assumed to be equal to \( 1 / p \). In this case, the total number of estimated parameters is given by \( M = K - 1 + K \times p \times (p - 1) \). Otherwise, initial state probabilities \( \beta \) are also estimated and the total number of parameters is \( M = K - 1 + K \times (p - 1) + K \times p \times (p - 1) \). Notation: \( p \) - number of states, \( n \) - sample size, \( K \) - number of mixture components, \( d \) - number of equivalence blocks.

Value

- `z`: matrix of posterior probabilities (n x K)
- `id`: classification vector (length n)
- `alpha`: vector of mixing proportions (length K)
- `beta`: matrix of initial state probabilities (K x p)
- `gamma`: array of transition probabilities (p x p x K)
- `logl`: log likelihood value
- `BIC`: Bayesian Information Criterion
References

Melnykov, V. (2016) Model-Based Biclustering of Clickstream Data, Computational Statistics and Data Analysis, 93, 31-45.


See Also

click.plot, click.forward, click.backward

Examples

```r
set.seed(123)
n.seq <- 50
p <- 5
K <- 2
mix.prop <- c(0.3, 0.7)

TP1 <- matrix(c(0.20, 0.10, 0.15, 0.15, 0.40,
                0.20, 0.20, 0.20, 0.20, 0.20,
                0.15, 0.10, 0.20, 0.20, 0.35,
                0.15, 0.10, 0.20, 0.20, 0.35,
                0.30, 0.30, 0.10, 0.10, 0.20), byrow = TRUE, ncol = p)

TP2 <- matrix(c(0.15, 0.15, 0.20, 0.20, 0.30,
                0.20, 0.10, 0.30, 0.30, 0.10,
                0.25, 0.20, 0.15, 0.15, 0.25,
                0.25, 0.20, 0.15, 0.15, 0.25,
                0.10, 0.30, 0.20, 0.20, 0.20), byrow = TRUE, ncol = p)

TP <- array(rep(NA, p * p * K), c(p, p, K))
TP[,,1] <- TP1
TP[,,2] <- TP2

# DATA SIMULATION
A <- click.sim(n = n.seq, int = c(10, 50), alpha = mix.prop, gamma = TP)
C <- click.read(A$S)

# EM ALGORITHM (without initial state probabilities)
N2 <- click.EM(X = C$X, K = 2)
N2$BIC
```
# EM ALGORITHM (with initial state probabilities)

```r
M2 <- click.EM(X = C$X, y = C$y, K = 2)
M2$BIC
```

## click.forward

**Forward search for equivalent states**

### Description

Runs forward search to detect blocks of equivalent states.

### Usage

```
click.forward(X, K, eps = 1e-10, r = 100, iter = 5, bic = TRUE,
              min.gamma = 1e-3, scale.const = 1.0, silent = FALSE)
```

### Arguments

- `X`: dataset array (p x p x n)
- `K`: number of mixture components
- `eps`: tolerance level
- `r`: number of restarts for initialization
- `iter`: number of iterations for each short EM run
- `bic`: flag indicating whether BIC or AIC is used
- `min.gamma`: lower bound for transition probabilities
- `scale.const`: scaling constant for avoiding numerical issues
- `silent`: output control

### Details

Runs forward search to detect blocks of equivalent states. States i and j are called equivalent if their behavior expressed in terms of transition probabilities is identical, i.e., the probabilities of leaving i and j to visit another state h are the same as well as the probabilities of coming to i and j from another state h are the same; this condition should hold for all mixture components. Notation: p - number of states, n - sample size, K - number of mixture components, d - number of equivalence blocks.
Value

- **z**: matrix of posterior probabilities (n x K)
- **alpha**: vector of mixing proportions (length K)
- **gamma**: array of transition probabilities (d x d x K)
- **states**: detected equivalence blocks (length p)
- **logl**: log likelihood value
- **BIC**: Bayesian Information Criterion
- **AIC**: Akaike Information Criterion
- **id**: classification vector (length n)

Author(s)

Melnykov, V.

References

Melnykov, V. (2016) Model-Based Biclustering of Clickstream Data, Computational Statistics and Data Analysis, 93, 31-45.


See Also

backward.search, click.EM

Examples

```r
set.seed(123)

n.seq <- 50

p <- 5
K <- 2
mix.prop <- c(0.3, 0.7)

TP1 <- matrix(c(0.20, 0.10, 0.15, 0.40,
    0.20, 0.20, 0.20, 0.20, 0.20,
    0.15, 0.10, 0.20, 0.20, 0.35,
    0.15, 0.10, 0.20, 0.20, 0.35,
    0.30, 0.30, 0.10, 0.10, 0.20), byrow = TRUE, ncol = p)

TP2 <- matrix(c(0.15, 0.15, 0.20, 0.30,
    0.20, 0.10, 0.30, 0.30, 0.10,
    0.25, 0.20, 0.15, 0.15, 0.25,
    0.25, 0.20, 0.15, 0.15, 0.25,
    0.10, 0.30, 0.20, 0.20, 0.20), byrow = TRUE, ncol = p)
```
TP <- array(rep(NA, p * p * K), c(p, p, K))
TP[,,1] <- TP1
TP[,,2] <- TP2

# DATA SIMULATION
A <- click.sim(n = n.seq, int = c(10, 50), alpha = mix.prop, gamma = TP)
C <- click.read(A$S)

# FORWARD SEARCH
click.forward(X = C$X, K = 2)

---

### click.plot

*Plot of the obtained clustering solution*

#### Description

Constructs a click-plot for the clustering solution.

#### Usage

```r
click.plot(X, y = NULL, file = NULL, id, states = NULL, marg = 1,
font.cex = 2, font.col = "black", cell.cex = 1, cell.lwd = 1.3,
cell.col = "black", sep.lwd = 1.3, sep.col = "black",
obs.lwd = NULL, colors = c("lightcyan", "pink", "darkred"),
col.levels = 8, legend = TRUE, leg.cex = 1.3, top.srt = 0,
frame = TRUE)
```

#### Arguments

- **X**: dataset array (p x p x n)
- **y**: vector of initial states (length n)
- **file**: name of the output pdf-file
- **id**: classification vector (length n)
- **states**: vector of state labels (length p)
- **marg**: plot margin value (for the left and top)
- **font.cex**: magnification of labels
- **font.col**: color of labels
- **cell.cex**: magnification of cells
- **cell.lwd**: width of cell frames
cell.col  color of cell frames
sep.lwd  width of separator lines
sep.col  color of separator lines
obs.lwd  width of observation lines
colors  edge colors for interpolation
col.levels  number of colors obtained by interpolation
legend  legend of color hues
leg.cex  magnification of legend labels
top.srt  rotation of state names in the top
frame  frame around the plot

Details
Constructs a click-plot for the provided clustering solution. Click-plot is a graphical display representing relative transition frequencies for the partitioning specified via the parameter 'id'. If the parameter 'file' is specified, the constructed plot will be saved in the pdf-file with the name 'file'. If the width of observation lines 'obs.lwd' is not specified, median colors will be used for all cell segments.

Author(s)
Melnykov, V.

References
Melnykov, V. (2016) Model-Based Biclustering of Clickstream Data, Computational Statistics and Data Analysis, 93, 31-45.

See Also
click.EM

Examples
set.seed(123)

n.seq <- 200

p <- 5
K <- 2
mix.prop <- c(0.3, 0.7)

TP1 <- matrix(c(0.20, 0.10, 0.15, 0.15, 0.40, c(0.20, 0.10, 0.15, 0.15, 0.40, ...
click.predict

Prediction of future state visits

Description
Calculates the transition probability matrix associated with the M-step transition.

Usage
```
click.predict(M = 1, gamma, pr = NULL)
```

Arguments
- **M**: number of transition steps (M = 1 by default)
- **gamma**: array of transition probabilities (p x p x K)
- **pr**: vector of probabilities associated with components (length K)
Details

Returns a transition probability matrix associated with the M-step transition. If the vector pr is not specified, all components are assumed equally likely.

Author(s)

Melnykov, V.

References

Melnykov, V. (2016) Model-Based Biclustering of Clickstream Data, Computational Statistics and Data Analysis, 93, 31-45.


See Also

click.EM

Examples

```r
set.seed(123)
n.seq <- 200
p <- 5
K <- 2
mix.prop <- c(0.3, 0.7)

TP1 <- matrix(c(0.20, 0.10, 0.15, 0.15, 0.40,
                0.20, 0.20, 0.20, 0.20, 0.20,
                0.15, 0.10, 0.20, 0.20, 0.35,
                0.15, 0.10, 0.20, 0.20, 0.35,
                0.30, 0.30, 0.10, 0.10, 0.20), byrow = TRUE, ncol = p)

TP2 <- matrix(c(0.15, 0.15, 0.20, 0.20, 0.30,
                0.20, 0.10, 0.30, 0.30, 0.10,
                0.25, 0.20, 0.15, 0.15, 0.25,
                0.25, 0.20, 0.15, 0.15, 0.25,
                0.10, 0.30, 0.20, 0.20, 0.20), byrow = TRUE, ncol = p)

TP <- array(rep(NA, p * p * K), c(p, p, K))
TP[,1] <- TP1
TP[,2] <- TP2

# DATA SIMULATION
```
A <- click.sim(n = n.seq, int = c(10, 50), alpha = mix.prop, gamma = TP)
C <- click.read(A$S)

# EM ALGORITHM
M2 <- click.EM(X = C$X, y = C$y, K = 2)

# Assuming component probabilities given by mixing proportions, predict the next state
click.predict(M = 1, gamma = M2$gamma, pr = M2$alpha)

# For the last location in the first sequence, predict the three-step transition
# location, given corresponding posterior probabilities
click.predict(M = 3, gamma = M2$gamma, pr = M2$z[1,][A$S[[1]][length(A$S[[1]])]])

---

**click.read**  

Reading sequences of visited states

**Description**

Prepares sequences of visited states for running the EM algorithm.

**Usage**

```r
click.read(S)
```

**Arguments**

- `S`  
  - list of numeric sequences

**Details**

Prepares sequences of visited states for running the EM algorithm by means of the `click.EM()` function.

**Value**

- `X`  
  - dataset array \((p \times p \times n)\) (\(p\) - # of states, \(n\) - # of sequences)
- `y`  
  - vector of initial states (length \(n\))

**Author(s)**

Melnykov, V.
References

Melnykov, V. (2016) Model-Based Biclustering of Clickstream Data, Computational Statistics and Data Analysis, 93, 31-45.


See Also

click.sim, click.EM

Examples

```r
set.seed(123)

n.seq <- 20

p <- 5
K <- 2
mix.prop <- c(0.3, 0.7)

TP1 <- matrix(c(0.20, 0.10, 0.15, 0.15, 0.40,
                0.20, 0.20, 0.20, 0.20, 0.20,
                0.15, 0.10, 0.20, 0.20, 0.35,
                0.15, 0.10, 0.20, 0.20, 0.35,
                0.30, 0.30, 0.10, 0.10, 0.20), byrow = TRUE, ncol = p)

TP2 <- matrix(c(0.15, 0.15, 0.20, 0.20, 0.30,
                0.20, 0.10, 0.30, 0.30, 0.10,
                0.25, 0.20, 0.15, 0.15, 0.25,
                0.25, 0.20, 0.15, 0.15, 0.25,
                0.10, 0.30, 0.20, 0.20, 0.20), byrow = TRUE, ncol = p)

TP <- array(rep(NA, p * p * K), c(p, p, K))
TP[,,1] <- TP1
TP[,,2] <- TP2

# DATA SIMULATION
A <- click.sim(n = n.seq, int = c(10, 50), alpha = mix.prop, gamma = TP)
C <- click.read(A$S)
C$X
C$y
```
### Description

Simulates sequences of visited states.

### Usage

```r
click.sim(n, int = c(5, 100), alpha, beta = NULL, gamma)
```

### Arguments

- `n`: number of sequences
- `int`: interval defining the lower and upper bounds for the length of sequences
- `alpha`: vector of mixing proportions (length K)
- `beta`: matrix of initial state probabilities (K x p)
- `gamma`: array of K p x p transition probability matrices (p x p x K)

### Details

Simulates 'n' sequences of visited states according to the following mixture model parameters: 'alpha' - mixing proportions, 'beta' - initial state probabilities, 'gamma' - transition probability matrices. If the matrix 'beta' is not provided, all initial states are assumed to be equal to 1 / p.

### Value

- `S`: list of simulated sequences
- `id`: true classification of simulated sequences

### Author(s)

Melnykov, V.

### References

Melnykov, V. (2016) Model-Based Biclustering of Clickstream Data, Computational Statistics and Data Analysis, 93, 31-45.


### See Also

`click.read`, `click.EM`
Examples

```r
# SPECIFY MODEL PARAMETERS
set.seed(123)

n.seq <- 20

p <- 5
K <- 2
mix.prop <- c(0.3, 0.7)

TP1 <- matrix(c(0.20, 0.10, 0.15, 0.15, 0.40,
                0.20, 0.20, 0.20, 0.20, 0.20,
                0.15, 0.10, 0.20, 0.20, 0.35,
                0.15, 0.10, 0.20, 0.20, 0.35,
                0.30, 0.30, 0.10, 0.10, 0.20),
               byrow = TRUE, ncol = p)

TP2 <- matrix(c(0.15, 0.15, 0.20, 0.20, 0.30,
                0.20, 0.10, 0.30, 0.30, 0.10,
                0.25, 0.20, 0.15, 0.15, 0.25,
                0.25, 0.20, 0.15, 0.15, 0.25,
                0.10, 0.30, 0.20, 0.20, 0.20),
               byrow = TRUE, ncol = p)

TP <- array(rep(NA, p * p * K), c(p, p, K))
TP[,1] <- TP1
TP[,2] <- TP2

# DATA SIMULATION
A <- click.sim(n = n.seq, int = c(10, 50), alpha = mix.prop, gamma = TP)
A
```

---

**click.var**

*Variance-covariance matrix estimation*

**Description**

Estimates the variance-covariance matrix for model parameter estimates.

**Usage**

`click.var(X, y = NULL, alpha, beta = NULL, gamma, z)`
Arguments

- `X` dataset array (p x p x n)
- `y` vector of initial states (length n)
- `alpha` vector of mixing proportions (length K)
- `beta` matrix of initial state probabilities (K x p)
- `gamma` array of transition probabilities (p x p x K)
- `z` matrix of posterior probabilities (n x K)

Details

Returns an estimated variance-covariance matrix for model parameter estimates.

Author(s)

Melnykov, V.

References

Melnykov, V. (2016) Model-Based Biclustering of Clickstream Data, Computational Statistics and Data Analysis, 93, 31-45.


See Also

click.EM

Examples

```r
set.seed(123)

n.seq <- 200

p <- 5
K <- 2
mix.prop <- c(0.3, 0.7)

TP1 <- matrix(c(0.20, 0.10, 0.15, 0.15, 0.40,
                0.20, 0.20, 0.20, 0.20, 0.20,
                0.15, 0.10, 0.20, 0.20, 0.35,
                0.15, 0.10, 0.20, 0.20, 0.35,
                0.30, 0.30, 0.10, 0.10, 0.20), byrow = TRUE, ncol = p)

TP2 <- matrix(c(0.15, 0.15, 0.20, 0.20, 0.30,
                0.20, 0.10, 0.30, 0.30, 0.10,
                0.25, 0.20, 0.15, 0.15, 0.25,
                0.25, 0.20, 0.15, 0.15, 0.25,
                0.25, 0.20, 0.15, 0.15, 0.25))
```
```
TP <- array(rep(NA, p * p * K), c(p, p, K))
TP[,1] <- TP1
TP[,2] <- TP2

# DATA SIMULATION
A <- click.sim(n = n.seq, int = c(10, 50), alpha = mix.prop, gamma = TP)
C <- click.read(A$S)

# EM ALGORITHM
M2 <- click.EM(X = C$X, y = C$y, K = 2)

# VARIANCE ESTIMATION
V <- click.var(X = C$X, y = C$y, alpha = M2$alpha, beta = M2$beta,
gamma = M2$gamma, z = M2$z)

# 95% confidence intervals for all model parameters
Estimate <- c(M2$alpha[-K], as.vector(t(M2$beta[-p])),
               as.vector(apply(M2$gamma[-p,,], 3, t)))
Lower <- Estimate - qnorm(0.975) * sqrt(diag(V))
Upper <- Estimate + qnorm(0.975) * sqrt(diag(V))
cbind(Estimate, Lower, Upper)
```

**Dataset: msnbc323**

**Description**

A portion of the msnbc dataset containing 323 clickstream sequences. This version of the original dataset (David Heckerman) was used in Melnykov (2014).

There are 17 states representing the following categories:
1: frontpage
2: news
3: tech
4: local
5: opinion
6: on-air
7: misc
8: weather
9: msn-news
10: health
11: living
12: business
13: msn-sports
14: sports
15: summary
16: bbs
17: travel

Usage

data(msnbc323)

Format

List of 323 numeric vectors representing categorical sequences.

Source

Melnykov, V. (2014)

References


Melnykov, V. (2016) Model-Based Biclustering of Clickstream Data, Computational Statistics and Data Analysis, 93, 31-45.


See Also

synth

Description

EM and search classes for printing and summarizing objects.
Usage

```r
## S3 method for class 'EM'
print(x, ...)
## S3 method for class 'EM'
summary(object, ...)
## S3 method for class 'search'
print(x, ...)
## S3 method for class 'search'
summary(object, ...)
```

Arguments

- `x` an object with the 'EM' (or 'search') class attributes.
- `object` an object with the 'EM' (or 'search') class attributes.
- `...` other possible options.

Details

Some useful functions for printing and summarizing results.

Author(s)

Melnykov, V.

References

Melnykov, V. (2016) Model-Based Biclustering of Clickstream Data, Computational Statistics and Data Analysis, 93, 31-45.


See Also

- `click.EM`

Description

The data represents the synthetic dataset used as an illustrative example in the Journal of Statistical Software paper discussing the use of the package. There are 5 states denoted as A, B, C, D, and E. Categorical sequences have lengths varying from 10 to 50.
Usage
data(synth)

Format
$\text{data contains a vector of 250 strings representing categorical sequences; }$ $\text{sid is the original classification vector.}$

Source
Melnykov, V. (2015)

References
Melnykov, V. (2016) Model-Based Biclustering of Clickstream Data, Computational Statistics and Data Analysis, 93, 31-45.

See Also
click.read

Examples

data(synth)
head(synth$data)

# FUNCTION THAT REPLACES CHARACTER STATES WITH NUMERIC VALUES
repl.levs <- function(x, ch.lev){
  for (j in 1:length(ch.lev)) x <- gsub(ch.lev[j], j, x)
  return(x)
}

# DETECT ALL STATES IN THE DATASET
d <- paste(synth$data, collapse = " ")
d <- strsplit(d, " ")[[1]]
ch.levs <- levels(as.factor(d))

# CONVERT DATA TO THE FORM USED BY click.read()
S <- strsplit(synth$data, " ")
S <- sapply(S, repl.levs, ch.levs)
S <- sapply(S, as.numeric)
head(S)
Index

* EM algorithm
  click.backward, 5
  click.EM, 7
  click.forward, 9
  click.plot, 11
  click.predict, 13
  click.read, 15
  click.sim, 17
  click.var, 18

* Markov model
  click.backward, 5
  click.EM, 7
  click.forward, 9
  click.plot, 11
  click.predict, 13
  click.read, 15
  click.sim, 17
  click.var, 18

* backward search
  click.backward, 5

* click-plot
  click.EM, 7
  click.plot, 11

* dataset
  B3, 3
  C, 4
  msnbc323, 20
  synth, 22

* forward search
  click.forward, 9

* prediction
  click.predict, 13

* variance estimation
  click.var, 18

B3, 3
C, 4
click.backward, 5
click.EM, 7

msnbc323, 20
print.EM(print.object), 21
print.object, 21
print.search(print.object), 21

summary.EM(print.object), 21
summary.search(print.object), 21
synth, 22