Package 'clickb'

July 22, 2025

Title Web Data Analysis by Bayesian Mixture of Markov Models

Version 0.1

Description Designed for web usage data analysis, it implements tools to process web sequences and identify web browsing profiles through sequential classification. Sequences' clusters are identified by using a model-based approach, specifically mixture of discrete time first-order Markov models for categorical web sequences. A Bayesian approach is used to estimate model parameters and identify sequences classification as proposed by Fruehwirth-Schnatter and Pamminger (2010) <doi:10.1214/10-BA606>.

License MIT + file LICENSE

Imports DiscreteWeibull, mclust, MCMCpack, parallel

Suggests seqHMM

Encoding UTF-8

RoxygenNote 7.1.1

NeedsCompilation no

Author Furio Urso [aut, cre],
Reza Mohammadi [aut],
Antonino Abbruzzo [aut],
Maria Francesca Cracolici [aut]

Maintainer Furio Urso <furio.urso@unipa.it>

Repository CRAN

Date/Publication 2023-02-13 09:20:10 UTC

Contents

clickb-package *Web Data Analysis by Bayesian Mixture of Markov Models*

Description

The R package **clickb** is for web sequences analysis and classification. One way to identify users' activity stages is relying on sequential data analysis, that aims to discover statistically relevant temporal structure where the values are delivered in sequences. In model-based clustering approach, longitudinal data are analysed and a model is trained to understand the underlying process generating the sequences. Markov processes assume sequences having a parametric distribution and depend on each sequence elements conditional probability of occurrence while satisfy the Markov property. In other words, these models allow to analyse patterns taking into account the probabilistic aspect of subjects movements in the sequence by considering the possible realization of a categorical variable as states of the Markov chain. Furthermore, We can use their extension to mixture of first-order Markov models to identify clusters of sequences generated by the same Markov model representing subpopulations in the data. Each subpopulation has its Markov model so that they may differ for the initial, transition or a combination. In clickstream data context, these differences mean that sequences belonging to the same cluster describe different browsing behaviour, as they display different preferences in starting the web path from a specific pages or to what web pages will be accessed in the next steps of the website exploration. Parameter estimation can be obtained through the Expectation-Maximization algorithm (Baum et al., 1970) as used by Cadez et al. (2003) in clickstream analysis context. However, we will use a Bayesian approach based on MCMC sampler considering Dirichlet priors for transition matrices rows as proposed by Fruehwirth-Schnatter and Pamminger (2010). This approach overcome a limitation of the EM algorithm that may struggle during the M-step if there are no transitions between two states. The package contains suitable tools to estimate parameters in mixture of first-order time-discrete Markov models for categorical response in a Bayesian framework, identifying clusters of sequences under the assumption that number of mixture components is considered fixed. The algorithm is based on Gibbs sampler moves and at each iteration assign web sequences in the new clusters based on a posterior probability. If simulated data are used or a previous classification is available, the algorithm compare the original cluster labels and the new ones through the Danon similarity index (Danon et al., 2005).

Author(s)

Furio Urso <furio.urso@unipa.it>,
Reza Mohammadi <a.mohammadi@uva.nl>,
Antonino Abbruzzo <antonino.abbruzzo@unipa.it>,
Maria Francesca Cracolici <mariafrancesca.cracolici@unipa.it>

References

Baum LE, Petrie T, Soules G, Weiss N (1970) A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *The annals of mathematical statistics* **41(1)**, 164–171

Cadez I, Heckerman D, Meek C, Smyth P, White S (2003) Model-based clustering and visualization of navigation patterns on a web site. *Data mining and knowledge discovery* 7(4), 399–424

Danon L, Diaz-Guilera A, Duch J, Arenas A (2005) Comparing community structure identification. *Journal of statistical mechanics: Theory and experiment* **2005(09)**, P09008

Fruehwirth-Schnatter S, Pamminger C (2010) Model-based clustering of categorical time series. *Bayesian Analysis* **5(2)**, 345–368

Examples

```
# Generate a sequential dataset from a mixture of Markov models
M <- 3    # number of components
K <- 6    # number of states
# define initial and transition probabilities for each component
ini1<-c(0.15,0.4,0.2, 0.15, 0,0.1)
A1<-matrix(c(0,0.45,0.1,0.15,0.15,0.15,
             0.1,0,0.15,0.15,0.1,0.5,
             0.25,0.2,0,0.2,0.15,0.2,
             0.15,0.2,0.05,0,0.4,0.2,
             0.15,0.05,0.45,0.15,0,0.2,
             0.4,0.15,0.2,0.05,0.2,0),byrow=TRUE,nrow=6)

ini2<-c(0.3,0.2,0.25, 0.15, 0, 0.1)
A2<-matrix(c(0,0.8,0,0,0,0.2,
             0.2,0,0.8,0,0,0,
             0,0.2,0,0.8,0,0,
             0,0,0.2,0,0.8,0,
             0,0,0,0.2,0,0.8,
             0.8,0,0,0,0.2,0),byrow=TRUE,nrow=6)

ini3<-c(0.2,0.1,0.2, 0.1, 0, 0.4)
A3<-matrix(c(0,0.1,0,0,0,0.9,
             0.8,0,0.15,0.05,0,0,
             0,0.9,0,0.1,0,0,
             0.05,0.05,0.8,0,0,0.1,
             0,0,0.05,0.9,0,0.05,
             0.05,0.05,0,0,0.9,0),byrow=TRUE,nrow=6)

trans.prob <- list(A1, A2, A3)
ini.prob <- list(ini1, ini2, ini3)

# sizes i.e. number of sequences in each component
N.sim1<-20
N.sim2<-30
N.sim3<-50

clust.size <- list(N.sim1, N.sim2, N.sim3)

T.range <- c(5, 30)  # sequences minimum length and maximum length

data<- sim_seq( M, K, ini.prob, trans.prob, clust.size, T.range)

### Estimate model parameters and identify cluster of sequences

# Set up initial values and hyper parameters (either fixed or random)
```

```
iter<-5   # number of iterations for the Gibbs sampler
burn<-0

num.cluster <- 3 # number of components
states <- 6 # number of states

ini.constr<-c(1, 1, 1, 1, 0, 1)           # constrains on initial probabilities
trans.constr<-matrix(c(0,1,1,1,1,1,       # constrains on transition probabilities
             1,0,1,1,1,1,
             1,1,0,1,1,1,
             1,1,1,0,1,1,
             1,1,1,1,0,1,
             1,1,1,1,1,0),byrow=TRUE,nrow=6)

# parameters initial values
A.ini <- 1/states*matrix(rep(1, length = (states^2)),
                         nrow = states, byrow = TRUE,
                         dimnames = list(as.character(c(1:states))) )
pi.ini <- rep(1/states, length = states)

# Prior distributions' hyperparameters
prior.ini<- prior.transrow <- prior.mixcoef <- 1

# Run the MCMC to estimate parameters
MMM_1 <- fit_mixmar(data, iter, burn, num.cluster = num.cluster, states = states,
                    A.ini = A.ini, pi.ini = pi.ini, prior.ini = prior.ini,
                    prior.transrow = prior.transrow, prior.mixcoef = prior.mixcoef,
                    ini.constr = ini.constr, trans.constr = trans.constr)

str(MMM_1$pi.list)            #Initial Markov chain probabilities for MCMC
str(MMM_1$A.list)             #Transition Markov chain probabilities for MCMC
str(MMM_1$Sim.index.Danon)    #Danon similarity between two partitions
str(MMM_1$Sim.index.Rand)     #Rand similarity between two partitions
MMM_1$Conf.mat                # Confusion matrix between two partitions
```

`fit_mixmar` *Bayesian estimation for mixture of Markov models with fixed number of components*

Description

The function use a sampling algorithm for estimating bayesian mixture of Markov models with fixed number of components. If simulated data are used, quality of classification is assessed comparing the original partition of the sequences with the one identified by the algorithm through Danon similarity and adjusted Rand indices.

Usage

```
fit_mixmar(
  data,
  iter = 1000,
  burn = 500,
  num.cluster,
  states,
  A.ini = NULL,
  pi.ini = NULL,
  prior.ini = 1,
  prior.transrow = 1,
  prior.mixcoef = 1,
  ini.constr = NULL,
  trans.constr = NULL
)
```

Arguments

data	are dataset containing a variable for categorical observations and an ID variable to identify subjects. No missing values allowed
iter	is the number of iterations
burn	is the number of burn-in iterations
num.cluster	is the number of clusters
states	is the number of states of the Markov chain
A.ini	is the list of initial values for the transition matrices
pi.ini	is the list of initial values for the initial probability vectors
prior.ini	is the hyperparameter for the prior distribution of initial probability vector (Dirichlet)
prior.transrow	is the hyperparameter for the prior distribution of transition matrices rows (Dirichlet)
prior.mixcoef	is the hyperparameter for the prior distribution of mixture coefficients (Dirichlet)
ini.constr	is the vector of zeros and ones indicating constrains in initial probabilities (zero: not possible to start from that state)
trans.constr	is the matrix of zeros and ones indicating constrains in transition probabilities (zero: transition is not allowed)

Value

pi.list	Initial Markov chain probabilities for MCMC
A.list	Transition Markov chain probabilities for MCMC
Sim.index.Danon	Danon similarity between two partitions
Sim.index.Rand	Rand similarity between two partitions
Conf.mat	Confusion matrix between two partitions

Author(s)

Furio Urso <furio.urso@unipa.it>

Examples

```
#Compare model-based clustering with respect to another classification

# Set up initial values and hyper parameters (either fixed or random)

iter<-5   # number of iterations for the Gibbs sampler
burn<-0

num.cluster <- 3 # number of components
states <- 5 # number of states

ini.constr<-c(1, 0, 1, 1, 1)           # constrains on initial probabilities
trans.constr<-matrix(c(1, 1, 1, 0, 1,  # constrains on transition probabilities
             1, 0, 1, 1, 1,
             1, 1, 1, 1, 0,
             0, 1, 1, 1, 1,
             1, 1, 0, 1, 1),byrow=TRUE,nrow=5)

# parameters initial values
A.ini <- 1/states*matrix(rep(1, length = (states^2)),
                         nrow = states, byrow = TRUE,
                         dimnames = list(as.character(c(1:states))) )
pi.ini <- rep(1/states, length = states)

# Prior distributions' hyperparameters
prior.ini<- prior.transrow <- prior.mixcoef <- 1

# data is the simulated sequential dataset obtained in the sim_seq() example

# Run the MCMC to estimate parameters
MMM_1 <- fit_mixmar(data, iter, burn, num.cluster = num.cluster, states = states,
                    A.ini = A.ini, pi.ini = pi.ini, prior.ini = prior.ini,
                    prior.transrow = prior.transrow, prior.mixcoef = prior.mixcoef,
                    ini.constr = ini.constr, trans.constr = trans.constr)
```

`sim_seq` *Simulate data*

Description

This function simulate a sequential dataset from a mixture of first-order Markov models generating categorical sequences. The output is a dataframe, columns are "id" to identify a subject/sequence, "y" to identify a categorical observation related to the sequence and "clus" the cluster label.

Usage

```
sim_seq(M, K, ini.prob, trans.prob, clust.size, T.range)
```

Arguments

M	is the number of components
K	is the number of Markov model states
ini.prob	is a list of initial probability vectors for each component
trans.prob	is a list of transition matrices for each component
clust.size	is a list of components' sizes
T.range	is a vector of two elements: minimum and maximum sequence length

Value

Object of class data.frame

Author(s)

Furio Urso <furio.urso@unipa.it>

Examples

```
# Simulate dataset from a mixture of Markov models
M <- 3    # number of components
K <- 5    # number of states
# define initial and transition probabilities for each component
ini1<-c(0.35, 0, 0.3, 0.2, 0.15)
A1<-matrix(c(0.15, 0.1, 0.5, 0, 0.25,
             0.2, 0, 0.1, 0.2, 0.5,
             0.6, 0.1, 0.1, 0.2, 0,
             0, 0.45, 0.35, 0.1, 0.1,
             0.15, 0.25, 0, 0.1, 0.5),byrow=TRUE,nrow=5)

ini2<-c(0.25, 0, 0.2, 0.25, 0.3)
A2<-matrix(c(0,0.8,0,0,0.2,
             0.2,0,0.8,0,0,
             0,0.2,0,0.8,0,
             0,0,0.2,0,0.8,
             0.8,0,0,0.2,0),byrow=TRUE,nrow=5)

ini3<-c(0.3, 0, 0.25, 0.3, 0.15)
A3<-matrix(c(0,0.1,0.2,0,0.7,
             0.7,0,0.2,0.1,0,
             0.1,0.8,0,0.1,0,
             0,0.1,0.7,0,0.2,
             0.2,0,0,0.8,0),byrow=TRUE,nrow=5)

trans.prob <- list(A1, A2, A3)
ini.prob <- list(ini1, ini2, ini3)
```

```
# sizes i.e. number of sequences in each component
N.sim1<-20
N.sim2<-30
N.sim3<-50

clust.size <- list(N.sim1, N.sim2, N.sim3)

T.range <- c(5, 30)  # sequences minimum length and maximum length

data<- sim_seq( M, K, ini.prob, trans.prob, clust.size, T.range)
```

Index