Package ‘supcluster’

October 14, 2022

Type Package

Title Supervised Cluster Analysis

Version 1.0.1

Date 2022-05-19

Author David A. Schoenfeld,
Jesse Hsu

Maintainer David A. Schoenfeld <dschoenfeld@mgh.harvard.edu>

Description Clusters features under the assumption that each cluster has a random effect and there is an outcome variable that is related to the random effects by a linear regression. In this way the cluster analysis is ``supervised'' by the outcome variable. An alternate specification is that features in each cluster have the same compound symmetric normal distribution, and the conditional distribution of the outcome given the features has the same coefficient for each feature in a cluster.

License GPL-2

Imports mvtnorm, gtools

Collate 'supcluster.R' 'concordmap.R' 'generate.cluster.data.R'
'tab1.R' 'binary_link.R'

NeedsCompilation no

Repository CRAN

Date/Publication 2022-05-19 14:50:02 UTC

R topics documented:

`supcluster-package` *Supervised Cluster Anaysis*

Description

The function clusters features under the assumption that each cluster has a random effect and there is an outcome variable that is related to the random effects by a linear regression. In this way the cluster analysis is "supervised" by the outcome variable. An alternate specification is that features in each cluster have the same compound symetric normal distribution, and the conditional distribution of the outcome given the features has the same coefficient for each feature in a cluster.

Details

Package:	supcluster
Type:	Package
Version:	1.0
Date:	2015-03-24
License:	GPL-2

The package consists of a function `supcluster` which reads a data frame whose columns include features and an outcome. It then peforms a cluster analysis that is supervised by the outcome as described above. The cluster analysis is performed using a Markoff Chain Monte Carlo algorythm, the output is a matrix where each row is a parameter vector consisting of the parameters of the multivariate normal distribution described above as well as the cluster membership of each of the features.

In addition there is function `concordmap` which produces a array with the posterior probability that each pair of features are in the same cluster and a function `compare.chains` used to compare these arrays for two chains in order to determine whether different chains have converged to the same set of clusters.

Author(s)

David A. Schoenfeld, Jesse Hsu Maintainer: David A. Schoenfeld <dschoenfeld@mgh.harvard.edu>
~~ The author and/or maintainer of the package ~~

References

~~ Literature or other references for background information ~~

See Also

supcluster, concordmap, compare.chains,beta.by.gene

`beta.by.gene` *Utility to Associate the Value of β with the Feature it is Assocated With*

Description

The model associates the coefficients of the random effects with the cluster number. However the cluster numbers are not unique. This utility associates the coefficient with gene that is in the cluster, for each cluster number.

Usage

```
beta.by.gene(supcluster.list)
```

Arguments

`supcluster.list`	The output of `supcluster`

Value

A matrix is returned with dimensions, the number of MCMC iterations by the number of genes/features +1. The first column is the chain number and the remain columns are the beta value for each of the gene/features

Author(s)

David A. Schoenfeld, Jessie Hsu

References

Added latter

See Also

supcluster,,compare.chains,concordmap

Examples

```
dat=generate.cluster.data(1)[[1]]
us=supcluster(dat,outcome="outcome",features=1:50,maxclusters=6,nstart=20,n=40)
vs=beta.by.gene(us)
colMeans(vs[,2:7])
```

`binaryLink` *Used with* `supcluster` *when the outcome data object is binary*

Description

Calculates the log-likelihood for a logistic model with log-odds $\mu + x$ where x is a frailty

Usage

```
binaryLink(x)
```

Arguments

`x`	A vector of binary data with values of 0 and 1 or `TRUE`, `FALSE`

Value

A function that given a vector of frialties followed by a value of μ calculates the log-likelihood

Author(s)

David Alan Schoenfeld

Examples

```
#Note the number of iterations is small to control run time
generatedData=generate.cluster.data(.25,npats=25,clusts=c(12,8),beta=c(-5,5),
                                    outcomeModel=binaryOutcome(0))
usBinary=supcluster(generatedData[[1]],outcome="outcome",
maxclusters=5,nstart=100,n=200,fbeta=FALSE,
linkLikelihood=binaryLink(generatedData[[2]]))
## The function is currently defined as
function (x)
{
    m = length(x)
    outfcn = function(parm2) {
        bx = parm2[m + 1] + parm2[1:m]
        loglik = sum(x * bx) - sum(log(1 + exp(bx)))
        return(loglik)
    }
    return(outfcn)
  }
```

`binaryOutcome` *Simulates a binary model for use with* `generate.cluster.data`

Description

Given a vector of frailties, say $x_1, \ldots$ it creates a binary variable from a logistic model with log odds ratio $\mu + x$

Usage

```
binaryOutcome(mu)
```

Arguments

`mu`	Constant term μ

Value

A vector of binary variables.

Author(s)

David A. Schoenfeld

Examples

```
generatedData=generate.cluster.data(.25,npats=25,clusts=c(12,8),beta=c(-5,5),
                                    outcomeModel=binaryOutcome(0))
## The function is currently defined as
function (mu)
{
    outpt = function(x) {
        m = length(x)
        p = exp(mu + x)/(1 + exp(mu + x))
        return(rbinom(m, 1, p))
    }
    return(outpt)
  }
```

`compare.chains` *Compare Chains to Test Algorithm Coverage*

Description

Suppose say 4 chains are run, then the first two and the last two are combined and a concord map of each is calculated, for each pair of genes in the concord map the proportion of times these genes are in the same cluster are calculated for each set of chains.

Usage

```
compare.chains(supcluster.list,chains1,chains2)
```

Arguments

`supcluster.list`	The output of `supcluster`
`chains1`	The first vector of the chains to be compared
`chains2`	The second vector of chains to be compared

Value

A N(N-1)/2 by 4 matrix is returned. The first two columns are each pair of genes and the next two are the proportion of times that each where in the same cluster in group of chains indicted by chain1 and chain2

Author(s)

David A. Schoenfeld, Jessie Hsu

See Also

`supcluster`,`compare.chains`, `beta.by.gene`

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.
#NOTE: only a small number of MCMC iterations are done due to time constraints

dat=generate.cluster.data(.2,npats=40,clusts=c(12,8,5),
                sig=1,gamma=1,beta=c(-5,0,6))[[1]]
us=supcluster(dat,outcome="outcome",features=1:25,maxclusters=4,nstart=20,n=40,nchains=2)
ts1=compare.chains(us,chains1=1,chains2=2)
#plot of one chain verses another
plot(ts1[,3],ts1[,4])
```

concordmap — *Calculate the Frequency with which each Pair of Features are in the Same Cluster*

Description

Label switching is a problem in interpreting the results of a cluster analysis that uses MCMC. Two clusterings may be the same but the labels of the clusters may change. In order to avoid this problem we create a square matrix with length and width equal to the number of features. The i,jth element is the proportion of times feature i and j are in the same cluster. A sorting algorythm puts the genes that are clustered together next to each other.

Usage

```
concordmap(supcluster.list, chains=1, sort.genes = FALSE,criteria=1)
```

Arguments

`supcluster.list`	The output of `supcluster`
`chains`	The chains to use in the clustering
`sort.genes`	If `TRUE` Genes that associated are put next to each other
`criteria`	Two genes are in the same cluster when the probability that they are in the same cluster is greater or equal to the criteria.

Value

If sort.genes=TRUE a three element list, the first element is a m x m matrix where m is the number of features and the second element is the ordering created by sorting algorythm that this matrix is in. The final element is the cluster membership for each of the genes. If sort genes=FALSE only the m x m matrix is returned.

Author(s)

David A. Schoenfeld, Jessie Hsu

See Also

`supcluster`,,`compare.chains`,`beta.by.gene`

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.
#NOTE: only a small number of MCMC iterations are done due to time constraints
dat=generate.cluster.data(.2,npats=40,clusts=c(12,8,5),
                 sig=1,gamma=1,beta=c(-5,0,6))[[1]]
```

```
us=supcluster(dat,outcome="outcome",features=1:25,maxclusters=4,nstart=20,n=40,nchains=2)
ts1=concordmap(us,chains=1)
#plot of the concord map
image(1:25,1:25,ts1$map)
```

coxLink — *Used with supcluster when the outcome data object is a censored survival variable.*

Description

Calculates the log-partial likelihood for a proportional hazards model with log-hazard $\mu+\beta x$ where x is a frailty

Usage

```
coxLink(data)
```

Arguments

data	A two variable data frame where the first variable is the survival time and the second variable is a censoring indicator 1-event happened 0-censored

Author(s)

David A. Schoenfeld

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.
generatedData=generate.cluster.data(.25,npats=25,clusts=c(12,8),beta=c(-5,5),
                                    outcomeModel=survivalOutcome(0,1,1,1))
usBinary=supcluster(generatedData[[1]],outcome="outcome",
maxclusters=5,nstart=100,n=200,fbeta=FALSE,
linkLikelihood=coxLink(generatedData[[2]]))
```

`generate.cluster.data` *Function to Generate Data According to the Supcluster Model*

Description

Generates cluster data according to the used for supervised clustering

Usage

```
generate.cluster.data(ratio,npats=80,clusts=c(12,8,12,12,6),
        sig=1,gamma=1,beta=c(-5,-2.5,0,2.5,5),outcomeModel=NULL)
```

Arguments

`ratio`	The ratio τ^2/σ^2 of the variance of the\ random effects to the error variance of the features
`npats`	Number of observations in the data set.
`clusts`	The cluster identity of the features
`sig`	The error variance of the features.
`gamma`	The error variance of the outcome.
`beta`	The value of the regression coefficients
`outcomeModel`	A function that returns a data frame with `npats` observations and rows that depend on the data object chosen. Two outcomeModel programs are provided, `binaryOutcome` and `survivalOutcome`, however users can write their own outcome model. If NULL no data object is returned

Value

A list with one element if outcomeModel=NULL which is a data frame which is `npats` times `ngens+1` the last column is the outcome. Otherwise a list of two data frames, one being the feature data and the other being the outcome data according to what outcomeModel is used.

Author(s)

David A. Schoenfeld

See Also

`supcluster`

gene_names *Trauma Data for Supervised Clustering*

Description

The data in `gene_names` is information on each gene in `trauma_data`

Format

`Gene.number` The number of the gene in the trauma.data set

`Probeset` The Affymetrix probeset code

`Gene.symbol.and.name` The annotation of the probeset

`Gene.sympbol` The gene symbol

Source

To be added from Glue grant https://imcc.mgh.harvard.edu/GlueGrant/trdb.html#nutshell

References

Tompkins, Ronald G. "Genomics of injury: the Glue Grant experience." The journal of trauma and acute care surgery 78.4 (2015): 671.

supcluster *Clustering of Features Supervised by an Outcome*

Description

We assume that each individual has set of features and an outcome, further we assume that the features are organized in clusters with a random effect for each cluster, and that the outcome is related to the random effects by a linear regression. The function supcluster performs an MCMC to determine the parameters of this model including the cluster membership of each feature. The program can also perform the estimation without considering the outcome. The outcome can be any data object, as long as it is related to the individual through a frialty.

Usage

```
supcluster(data,outcome,features,log.transform=TRUE,maxclusters=10,
nstart=100,n=500,shape=1,scale=1,alpha=1,betaP=1,fixj="random",
fbeta=FALSE,starting.value=NULL,nchains=1,linkLikelihood = NULL)
```

Arguments

`data`	A data frame of the input data
`outcome`	Either the variable number or the variable name of the outcome variable. If `fbeta=TRUE`, no outcome variable is used. If NULL we assume the outcome is a data object and there is a likelihood relating it to a per-patient frialty variable. In that case linkLikelihood cannont be NULL
`features`	A list of features either as variable names or column numbers this can't be mixed
`log.transform`	Log transform the feature data. Generally used when the features are gene expressons
`maxclusters`	The maximum number of clusters used
`nstart`	The first nstart-1 values of each MCMC chain are not reported, that is used as a "burn in".
`n`	The number of MCMC iterations for each chain
`shape`	The shape parameter for the prior on the variance components
`scale`	The starting scale parmeter for the prior on the variance components
`alpha`	The value to use for the Dirichelet prior parameter
`betaP`	The prior precision of the regression parameters.
`fixj`	If `"random"`, then the starting value for cluster membership is set at random. If `"kmeans"` it uses kmeans to set the starting value. Otherwise it is matrix of features verses clusters, where a 1 indicates that feature i is in cluser j and the cluster membership is assumed to be known. `fixj` should be set to `"random"` when multiple chains are run.
`fbeta`	If TRUE then the outcome is not used in the clustering algorithm
`starting.value`	Starting value for the MCMC. It should be left as NULL when multiple chains are run, in which case the starting cluster membership is determined by `fixj`. Otherwise it is parameter vector similar to the one described under "value" below.
`nchains`	Number of chains to run
`linkLikelihood`	Likelihood function for model linking actual outcome data to the per-patient frialty. The input of the function is a vector of length `dim(data)[1]+nparms`, where `nparms` is the number of parameters in the outcome model. The first part of the vector are the frailties and the second part are the parameters of the model. If NULL then `outcome` is used.

Value

A compound list is returned. At the first level is the chain number. At the second level there are two elements

`inp`	This has twp values `maxclusters` giving the maximum number of clusters and `ngenes` giving the maximum number of features
`parms`	This is a `n` by `3+maxclusters+ngenes` matrix. Each row is one MCMC iteration. The first three columns are the values of the variance components σ^2, τ^2, and γ^2 the next `maxcluster` values are the regression coefficients for each cluster and the final `ngenes` values are the cluster membership of each feature

Note

When the feature space is large this program runs slowely. In the example only 20 iterations where used for the burn in and only 80 iterations are run. In general this would not be adequate to fully explore the feature space.

Author(s)

David A. Schoenfeld, Jessie Hsu

References

Hsu, Jessie J., Dianne M. Finkelstein, and David A. Schoenfeld. "Outcome-driven cluster analysis with application to microarray data." PloS one 10.11 (2015): e0141874.

See Also

`concordmap`, `compare.chains`,`beta.by.gene`

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets
##--Note you need to change nstart and n in example to get enough iterations
#run supcluster on trauma data.  Note: nstart and n must be increased to,say, 2000,3000
#and maxclusters increased to 20
data("trauma_data")
us=supcluster(trauma_data,outcome="outcome",features=1:87,
              maxclusters=5,nstart=5,n=20,fbeta=FALSE)
#creates plot in paper
usm=concordmap(us,chains=1,sort.genes=TRUE)
image(1:87,1:87,usm$map,xlab='Genes',ylab='Genes',
      main="Trauma Data Example",
      col=gray(16:1 / 16))
#Associate genes with clusters
data("gene_names")
betas=colSums(us[[1]]$parms[,3:22])
outpt=data.frame(cluster.number=usm$clusters,beta=betas[usm$clusters],gene_names[usm$order,])
```

---

survivalOutcome *Simulates a survival model for use with* `generate.cluster.data`

Description

Given a vector of frailties, say $x_1, ...$, this function generates a censored exponentially distributed random variable with rate equal to $\mu + \beta x_i$. The censoring distribution is uniform with from f to $f + a$.

Usage

```
survivalOutcome(mu, beta, accrual, followUp)
```

Arguments

`mu`	The constant term μ
`beta`	The frailty effect β
`accrual`	The accrual time a, in a clinical study
`followUp`	The follow up time f in a clinical study

Value

A data frame is returned with two columns `survival` and `censor`

Author(s)

David A. Schoenfeld

See Also

`coxLink`,`binaryOutcome`,`binaryLink`

Examples

```
generatedData=generate.cluster.data(.25,npats=25,clusts=c(12,8),beta=c(-5,5),
                                    outcomeModel=survivalOutcome(0,1,1,1))
usBinary=supcluster(generatedData[[1]],outcome="outcome",
maxclusters=5,nstart=100,n=200,fbeta=FALSE,
linkLikelihood=coxLink(generatedData[[2]]))
```

`tab1` *Simulates Supcluster Function*

Description

Produces summary statistics from a simulation of supcluster

Usage

```
tab1(ratio=4,reps=100,n=1000,start=500,fbeta=FALSE,
                maxclusters=5,chains=1,clusts=c(15,15,20),
                sig=1,gamma=1,npats=80,beta=seq(-5,5,5),
                plot=FALSE)
```

Arguments

ratio	The ratio of tau to sigma
reps	The number of runs
n	The number of MCMC iterations
start	The first MCMC iteration used
fbeta	If TRUE the outcome is not used
maxclusters	The maximum number of clusters for the estimation step
chains	The number of chains to run
clusts	A list of the number of genes in each cluster
sig,gamma,beta	The parameters sigma,gamma,beta
npats	The number of experimental units(patients)
plot	Plots the first run

Value

A data frame is returned with the mean parameter value, it's standard error and the mean of it's standard error calculated from the MCMC

Author(s)

David A. Schoenfeld, Jessie Hsu

See Also

supcluster,,compare.chains,concordmap

Examples

```
#very few iterations done so that this runs in less than 5 seconds.
#You need to change reps=100,start=2000,n=3000 to get enough iterations
tab1(ratio=2,reps=5,n=10,start=1,maxclusters=5)
```

---

trauma_data *Trauma Data for Supervised Clustering*

Description

This is a genomic data set, saved as an R save file, that loaded with data("trauma_data") and data("gene_names") The data frame trauma_data has 147 observations on patients with trauma. The first 87 columns are gene expression values and the final column labeled outcome is the multiple organ failure score for the patient. The data in gene_names is information on each gene in trauma_data

Usage

```
data("trauma_data");data("gene_names")
```

Format

A data frame `trauma_data` with 147 observations the first 87 columns are gene expression data and the last column labeled `outcome` is the maximum organ failure score. A data frame `gene_names` with the affymetrix description of the probesets in `trauma_data`.

Source

N. Rajicic, Dianne M. Finkelstein, and David A. Schoenfeld.(2007) "Survival analysis of longitudinal microarrays." *Bioinformatics*, 22(21):2643-2649

Index