Package ‘supcluster’

May 18, 2015

Type Package
Title Supervised Cluster Analysis
Version 1.0
Date 2015-05-18
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
NeedsCompilation no
Repository CRAN
Date/Publication 2015-05-18 22:45:09

R topics documented:

supcluster-package .................................................. 2
beta.by.gene .......................................................... 3
compare.chains ....................................................... 4
concordmap ............................................................. 5
generate.cluster.data ............................................. 6
gene_names ........................................................... 7
supcluster ............................................................. 7
tab1 ................................................................. 9
trauma_data .......................................................... 10
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 symmetric 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 performs a cluster analysis that is supervised by the outcome as described above. The cluster analysis is performed using a Markov Chain Monte Carlo algorithm, 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 an 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>

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 $\beta$ with the Feature it is Associated 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**

```r
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**

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

```R
# 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))
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 algorithm 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 algorithm 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**

```r
# 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))
```
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,ratio=ratio,npats=80,clusts=c(12,8,12,12,6),
sig=1,gamma=1,beta=c(-5,-2.5,0,2.5,5))

Arguments

- **ratio**: The ratio $\tau^2/\sigma^2$ 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

Value

A data frame which is npats times ngens+1 the last column is the outcome.

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`.

**Usage**

`gene_names`

**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.symbol**  The gene symbol

**Source**

To be added from Glue grant

**References**

To be added

---

**supcluster**  
*Clustering of Features Supervised by an Outcome*

**Description**

We assume that each individual has a 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.

**Usage**

```r
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)
```
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.

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 expressions

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 parameter 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 cluster 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

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 two 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 $\sigma^2, \tau^2,$ and $\gamma^2$ 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 slowly. In the example only 20 iterations were 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

To be added once the paper is published

See Also

cordmap, compare.chains,beta.by.gene

Examples

```r
# Should be DIRECTLY executable !! ----
# >>> Define data, use random,
#TOR 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[1:3:22])
outpt=data.frame(cluster.number=usm$clusters,beta=betas[usm$clusters],gene_names[usm$order,])
```

---

**tab1**

*Simulates Supcluster Function*

Description

Produces summary statistics from a simulation of supcluster

Usage

```r
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)
```
trauma_data

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.
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

Index

*Topic cluster
  concordmap, 5
  supcluster, 7
  tab1, 9
*Topic datasets
  gene_names, 7
  trauma_data, 10
*Topic package
  supcluster-package, 2
*Topic supervised clustering
  beta.by.gene, 3
  compare.chains, 4

beta.by.gene, 2, 3, 4, 5, 9

compare.chains, 2–4, 4, 5, 9, 10
concordmap, 2, 3, 5, 9, 10

gene_names, 7
generate.cluster.data, 6

supcluster, 2–6, 7, 10
supcluster-package, 2

tab1, 9
trauma_data, 10