Package 'SpNMF'

January 20, 2025

Type Package

Title Supervised NMF

Version 0.1.1

Description Non-negative Matrix Factorization(NMF) is a powerful tool for identifying the key features of microbial communities and a dimension-reduction method. When we are interested in the differences between the structures of two groups of communities, supervised NMF(Yun Cai, Hong Gu and Tobby Kenney (2017),<doi:10.1186/s40168-017-0323-1>) provides a better way to do this, while retaining all the advantages of NMF -- such as interpretability, and being based on a simple biological intuition.

Depends R (>= 3.2.3),

Imports NMF, stats

License GPL-3

Encoding UTF-8

LazyData true

NeedsCompilation no

Maintainer Yun Cai <Yun.Cai@dal.ca>

Author Yun Cai [aut, cre],
Hong Gu [aut],
Toby Kenney [aut]

Repository CRAN

Date/Publication 2018-10-03 07:50:03 UTC

Contents

chty *chty*

Description

chty is used to get number of types for the data.

Usage

```
chty(data,y,k,maxr)
```

Arguments

`data` an optional n by p count data matrix. The p columns of the matrix are different variables and the n rows are samples. Each column should contain at lest one none zero entry. When n = 1, it is a row vector.

`y` a binary variable contains classification information of the data. Usually one group is labelled as "0" and the other as "1".

`k` a value gives the number of folds used in cross validation when choosing number of types.

`maxr` a number gives the upper bound of the number of types.

Value

`r1` the suggested number of types for class labeled as 1.

`r2` the suggested number of types for class labeled as 0.

Author(s)

Yun Cai, Hong Gu and Toby Kenney

References

Learning Microbial Community Structures with Supervised and Unsupervised Non-negative Matrix Factorization

Examples

```
##we use the simulated data spdata here
##the spdata is simulated from feature matrix combined by 2
#3types features from one group and 3 types from the other.
##choose number of types using our function
##2-folds cross validation is used here
##the upper bound of number of types for both classes is 2
##remove all zero variables from the data
spdata.rm=spdata[c(1:4,41:44),colSums(spdata)!=0]
y=c(rep(1,4),rep(0,4))
```

```
types=chty(spdata.rm,y,2,2)
#number of types for class labeled as 1
nmb1 = types$r1
#number of types for class labeled as 0
nmb2 = types$r2
```

getT *getT*

Description

getT is used to calculate the combined feature matrix.

Usage

```
getT(data,y,Tr1,Tr2)
```

Arguments

`data` an optional n by p count data matrix. The p columns of the matrix are different variables and the n rows are samples. Each column should contain at lest one none zero entry. When n = 1, it is a row vector.

`y` a binary variable contains classification information of the data. Usually one group is labelled as "0" and the other as "1".

`Tr1` a value gives the number of types for class labeled as 1. The appropriate Tr1 can also be estimated from function `chty`.

`Tr2` a value gives the number of types for class labeled as 0. The appropriate Tr2 can also be estimated from function `chty`.

Details

getT is used to calculate the combined feature matrix. The data used in getT should contain samples from both classes. If feature matrix is needed for only one class, `basis(NMF(data; Tr; ”KL”))` can be used.

Value

`T` a feature matrix in dimension p by r. It is a combined feature matrix contains information from both classes.

Author(s)

Yun Cai, Hong Gu and Tobby Kenney

References

Learning Microbial Community Structures with Supervised and Unsupervised Non-negative Matrix Factorization

Examples

```
#get feature matrix with rank 2 for one group and rank 3 for the other of the simualted spdata
y=c(rep(1,4),rep(0,4))
T.eg=getT(spdata,y,2,3)
```

spdata *spdata*

Description

the spdata is simulated from poisson distribution with mean as the product of feature and weight matrix. The feature matrix has 2804 variables and is combined by 2 types features from one group and 3 types from the other. The weight matrix is generated from uniform distribution on 0,1.

Format

The format is: int [1:80, 1:2804] 5 12 7 10 14 1 12 18 4 26 ... - attr(*, "dimnames")=List of 2 ..$: chr [1:80] "ibd.old0" "ibd.old0" "ibd.old0" "ibd.old0" ... ..$: NULL

Details

The spdata has a dimention of 80 by 2804, 40 labeled as class one and the left labeled as class two.

Examples

```
data(spdata)
```

spnmf *spnmf*

Description

The spnmf is used to fit supervised Non-negative Matrix Factorization model on data when the combined feature matrix is known.

Usage

```
spnmf(data,Tp)
```

Arguments

`data` an optional n by p count data matrix. The p columns of the matrix are different variables and the n rows are samples. Each column should contain at lest one none zero entry. When n = 1, it is a row vector.

`Tp` a combined feature matrix in dimension p by r. p is the number of variables and r is the number of types. Tp can also be calculated from function `getT`.

Details

The function is based on R package NMF.

Value

`W` the supervised weight matrix in dimension n by r. n is the number of observations. r is the number of type for the data. It is the coefficients of the feature matrix.

`loglh` the log-likelihood of the supervised NMF model.

Author(s)

Yun Cai, Hong Gu and Toby Kenney

References

Learning Microbial Community Structures with Supervised and Unsupervised Non-negative Matrix Factorization

Examples

```
##an example of classification based on supervised nmf results
#spdata consists of two classes, the first 40 samples are from class 1 and the left from class 2
##label each observation's class as 1 or 0
y=c(rep(1,4),rep(0,4))
##split the data half as training data the other half as test data
y.train=y.test=c(rep(1 ,2),rep(0,2))
spdata.train=spdata[c(1:2,41:42),]
spdata.test=spdata[c(21:22,61:62),]
#remove all zero columns
spdata.train.rm=spdata.train[,colSums(spdata.train)!=0]
#remove the same variables from test data
spdata.test.rm=spdata.test[,colSums(spdata.train)!=0]

#get feature matrix with rank 2 and 3 for the two groups
T.eg=getT(spdata.train.rm,y.train,2,3)

#get weight matrix
rs.train=spnmf(spdata.train.rm,T.eg)
w.train=rs.train$W
rs.test=spnmf(spdata.test.rm,T.eg)
w.test=rs.test$W
##the weight matrix can be used to do classification
md.train=glm(y.train~.,data=data.frame(w.train),family=binomial(link=logit))
##predict the test data
pred=predict(md.train,newdata=data.frame(w.test),type =”response”)
```

Index