Package 'sigFeature'

May 6, 2026

Type Package

Title sigFeature: Significant feature selection using SVM-RFE & t-statistic

Version 1.31.0

Date 2021-11-21

Depends R (>= 3.5.0)

Suggests RUnit, BiocGenerics, knitr, rmarkdown

Description This package provides a novel feature selection algorithm for binary classification using support vector machine recursive feature elimination SVM-RFE and t-statistic. In this feature selection process, the selected features are differentially significant between the two classes and also they are good classifier with higher degree of classification accuracy.

License GPL (>= 2)

Encoding UTF-8

LazyData TRUE

biocViews FeatureExtraction, GeneExpression, Microarray, Transcription, mRNAMicroarray, GenePrediction, Normalization, Classification, SupportVectorMachine

Imports biocViews, nlme, e1071, openxlsx, pheatmap, RColorBrewer, Matrix, SparseM, graphics, stats, utils, SummarizedExperiment, BiocParallel, methods

VignetteBuilder knitr

git_url https://git.bioconductor.org/packages/sigFeature

git_branch devel

git_last_commit 7a6d947

git_last_commit_date 2026-04-28

Repository Bioconductor 3.24

Date/Publication 2026-05-05

Author Pijush Das Developer [aut, cre], Dr. Susanta Roychudhury User [ctb], Dr. Sucheta Tripathy User [ctb]

Maintainer Pijush Das Developer <topijush@gmail.com>

Contents

`ExampleRawData` *Example dataset to test the performance of the sigFeature package.*

Description

For this significant feature selection procedure, microarray data (GSE2280) from patients with squamous cell carcinoma of the oral cavity (OSCC) has been used(O'Donnell RK et al. (2005)). Affymetrix Human Genome Array, U133A was selected for genome-wide transcription analysis for this data set. In this paper, the gene expression profiles obtained from primary squamous cell carcinoma of the oral cavity (OSCC) that were metastatic to lymph nodes (N+) compared to those that were not metastatic (N-). A total of 18 OSCCs were analyzed for gene expression. In their analysis a predictive rule was built using a support vector machine, and the accuracy of the rule was evaluated using cross-validation the original data set and prediction of an independent set of four patients. A signature gene set is produced which is able to predict the four independent patients correctly as well as associating five lymph node metastases from the original patient set with the metastatic primary tumour group.

Usage

```
data("ExampleRawData")
```

Format

```
The format is:
num [1:27, 1:2205] 72.5 177.2 75.7 128.9 142 ...
- attr(*, "dimnames")=List of 2
..$ : chr [1:27] "GSM42246" "GSM42248" "GSM42250" "GSM42252" ...
..$ : chr [1:2205] "1494_f_at" "179_at" "200014_s_at" "200059_s_at" ...
```

Details

For "sigFeature" package evaluation, the microarray dataset has been classified into two classes such as lymph node metastatic (N+) and No lymph node metastatic (N-) (according to the TNM staging), provided in the dataset. After downloading the data set from GEO database firstly, it was normalized using the "quantile" normalization method using the Bioconductor package "limma". To reduce the runtime of this sigFeature function, a subset of the total dataset is taken by the ratio between the difference between two groups with cut off value (p-value 0.07). Now the expression value of the sub-dataset is considered here as "x". The patients without lymph node metastasis are represented -1, and the patients with lymph node metastasis are represented as 1. Those -1 and 1 value is incorporated with "y" as sample labels.

Value

`ExampleRawData` Return the values stored in the variable.

Source

https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE2280

References

O'Donnell RK, Kupferman M, Wei SJ, Singhal S et al. Gene expression signature predicts lymphatic metastasis in squamous cell carcinoma of the oral cavity. Oncogene 2005 Feb 10;24(7):1244-51. PMID: 15558013

Examples

```
data("ExampleRawData")
```

---

featsweepSigFe *Processed output data after using the function named "sigCVError()".*

Description

The variable "featsweepSigFe" contains the output of the function named "sigCVError()". The features which are produced on the basis of frequency are used here to enumerate mean external cross validation (k-fold) errors and the standard deviation of the errors. Training and test samples are same which are initially produced after splitting the main sample set in to k-folds. In each iteration, k-1 folds are considered as training samples and remaining one fold is considered as testing samples. In this external cross validation procedure, feature numbers are increased one by one by using the expression values from training dataset as well as test dataset. After that, traning samples are trained to test the tesing samples dynamically. The number of un classified samples are averaged and are called as external cross validation error rate.

Usage

```
data("featsweepSigFe")
```

Format

```
The format is:
    List of 400
    $ :List of 3
        ..$ svm.list:List of 10
        .. ..$ :'data.frame':  1 obs. of  4 variables:
        .. .. ..$ gamma     : num 0.000244
        .. .. ..$ cost      : num 0.001
        .. .. ..$ error     : num 0.333
        .. .. ..$ dispersion: num NA
        .. ..$ :'data.frame':  1 obs. of  4 variables:
        .. .. ..$ gamma     : num 0.000244
        .. .. ..$ cost      : num 0.001
        .. .. ..$ error     : num 0.333
        .. .. ..$ dispersion: num NA
        .. ..$ :'data.frame':  1 obs. of  4 variables:
        .. .. ..$ gamma     : num 0.000244
        .. .. ..$ cost      : num 0.001
        .. .. ..$ error     : num 0.667
        .. .. ..$ dispersion: num NA
        .. ..$ :'data.frame':  1 obs. of  4 variables:
        .. .. ..$ gamma     : num 0.000244
        .. .. ..$ cost      : num 0.001
        .. .. ..$ error     : num 0.333
        .. .. ..$ dispersion: num NA
        .. ..$ :'data.frame':  1 obs. of  4 variables:
        .. .. ..$ gamma     : num 0.000244
        .. .. ..$ cost      : num 0.001
        .. .. ..$ error     : num 0.333
        .. .. ..$ dispersion: num NA
        .. ..$ :'data.frame':  1 obs. of  4 variables:
        .. .. ..$ gamma     : num 1
        .. .. ..$ cost      : num 1.78
        .. .. ..$ error     : num 0.667
        .. .. ..$ dispersion: num NA
        .. ..$ :'data.frame':  1 obs. of  4 variables:
        .. .. ..$ gamma     : num 0.000244
        .. .. ..$ cost      : num 0.001
        .. .. ..$ error     : num 0
        .. .. ..$ dispersion: num NA
        .. ..$ :'data.frame':  1 obs. of  4 variables:
        .. .. ..$ gamma     : num 0.000244
        .. .. ..$ cost      : num 0.001
        .. .. ..$ error     : num 0
        .. .. ..$ dispersion: num NA
        .. ..$ :'data.frame':  1 obs. of  4 variables:
        .. .. ..$ gamma     : num 0.000244
        .. .. ..$ cost      : num 0.001
        .. .. ..$ error     : num 0.5
        .. .. ..$ dispersion: num NA
        .. ..$ :'data.frame':  1 obs. of  4 variables:
        .. .. ..$ gamma     : num 0.000244
```

```
.. .. ..$ cost      : num 0.001
.. .. ..$ error     : num 0.5
.. .. ..$ dispersion: num NA
..$ error   : num 0.367
..$ errorSD : num 0.233
...
```

Value

`featsweepSigFe` Return the values stored in the variable.

Source

https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE2280

References

O'Donnell RK, Kupferman M, Wei SJ, Singhal S et al. Gene expression signature predicts lymphatic metastasis in squamous cell carcinoma of the oral cavity. Oncogene 2005 Feb 10; 24(7):1244-51. PMID: 15558013

Examples

```
data(featsweepSigFe)
## maybe str(featsweepSigFe) ; plot(featsweepSigFe) ...
```

---

featureRankedList *Processed output data after using the function named "svmrfeFeatureRanking()".*

Description

The variable "featureRankedList" contains the output of the function named "svmrfeFeatureRanking()".

Usage

```
data("featureRankedList")
```

Format

```
The format is:
int [1:2204] 1073 1404 1152 5 1253 1557 105 1207 792 57 ...
```

Details

To solve the classification problem with the help of ranking the features an algorithm was proposed by Guyon named SVM-RFE. In this algorithm the dataset has been trained with SVM linear kernel model and removed the feature with smallest ranking criterion. This criterion is the w value of the decision hyperplane given by the SVM.

Value

`featureRankedList`
: returns the feature list.

Source

http://www.uccor.edu.ar/paginas/seminarios/Software/SVM-RFE.zip

References

Guyon, I., et al. (2002) Gene selection for cancer classification using support vector machines, Machine learning, 46, 389-422.

Examples

```
data(featureRankedList)
## maybe str(featureRankedList) ; plot(featureRankedList) ...
```

PlotErrors *Plot the mean CV errors.*

Description

A useful function for plotting the errors which is enumerated by using the function sigCVError().

Usage

```
PlotErrors(featsweepSigFe, ylim.min=0, ylim.max=0)
```

Arguments

`featsweepSigFe` a list variable containing the gamma, cost, error, dispersion values.

```
The format is:
    List of 1
    ...
    $ :List of 3
        ..$ svm.list:List of 1
        .. ..$ :data.frame: 1 obs. of  4 variables:
        .. .. ..$ gamma     : num 0.000244
        .. .. ..$ cost      : num 0.001
        .. .. ..$ error     : num 0.5
        .. .. ..$ dispersion: num NA
        ..$ error   : num 0.367
        ..$ errorSD : num 0.233
```

`ylim.min` minimum y label value in the graph.

`ylim.max` maximum y label value in the graph.

Details

This plot function will show the errors.

Value

returns plot.

Author(s)

Pijush Das <topijush@gmail.com>, et al.

References

Chang, Chih-Chung and Lin, Chih-Jen: LIBSVM 2.0: Solving Different Support Vector Formulations. http://www.csie.ntu.edu.tw/~cjlin/papers/libsvm2.ps.gz

Chang, Chih-Chung and Lin, Chih-Jen:Libsvm: Introduction and Benchmarks, http://www.csie.ntu.edu.tw/~cjlin/papers/

See Also

svm, svm.fs

Examples

```
# Example for PlotErrors()
# Data set taken from GSE2280

data(featsweepSigFe)
dim(featsweepSigFe)

#For plotting the mean external cross validation
#error and the standard deviation of the of the
#miss classifications of top 400 features.
PlotErrors(featsweepSigFe, 0, 0.4)
```

---

results *Processed output data after using the function named "sigFeature.enfold()".*

Description

The variable "results" contains the output of the function named "sigFeature.enfold()". To produce the output using "sigFeature.enfold(x,y,"kfold",10)" function the dataset is devided into 10 folds. Each time one fold is kept and the remaining k-1 folds are used to generate the features. Later the one fold is used as test sample and the remaining k-1 fold samples are used as training samples. The "results" variable contains feature.ids (address of the features), train.data.ids (training dataset ids), test.data.ids (test dataset ids), train.data.level (training dataset levels) and test.data.level (test dataset levels).

Usage

```
data("results")
```

Format

```
The format is:
List of 10
    $ :List of 5
    ..$ feature.ids     : int [1:2204] 2064 2031 2035 1573 370  ...
    ..$ train.data.ids  : chr [1:24] "GSM42246" "GSM42248" ...
    ..$ test.data.ids   : chr [1:3] "GSM42263" "GSM42251" "GSM42260"
    ..$ train.data.level: Named num [1:24] 1 1 1 1 1 -1 -1 -1 -1 -1 ...
    .. ..- attr(*, "names")= chr [1:24] "GSM42246" "GSM42248" ...
    ..$ test.data.level : Named num [1:3] -1 1 1
    .. ..- attr(*, "names")= chr [1:3] "GSM42263" "GSM42251" "GSM42260"
    $ :List of 5
    ..$ feature.ids     : int [1:2204] 1577 2064 370 2035 2032 605 ...
    ..$ train.data.ids  : chr [1:24] "GSM42246" "GSM42248" ...
    ..$ test.data.ids   : chr [1:3] "GSM42267" "GSM42256" "GSM42261"
    ..$ train.data.level: Named num [1:24] 1 1 1 1 1 -1 -1 -1 -1 -1 ...
    .. ..- attr(*, "names")= chr [1:24] "GSM42246" "GSM42248" ...
    ..$ test.data.level : Named num [1:3] -1 1 1
    .. ..- attr(*, "names")= chr [1:3] "GSM42267" "GSM42256" "GSM42261"
    $ :List of 5
    ..$ feature.ids     : int [1:2204] 246 2006 1174 2032 1502 ...
    ..$ train.data.ids  : chr [1:24] "GSM42246" "GSM42248" ...
    ..$ test.data.ids   : chr [1:3] "GSM42265" "GSM42272" "GSM42255"
    ..$ train.data.level: Named num [1:24] 1 1 1 1 1 -1 -1 -1 ...
    .. ..- attr(*, "names")= chr [1:24] "GSM42246" "GSM42248" ...
    ..$ test.data.level : Named num [1:3] -1 -1 1
    .. ..- attr(*, "names")= chr [1:3] "GSM42265" "GSM42272" "GSM42255"
    $ :List of 5
    ..$ feature.ids     : int [1:2204] 2064 611 525 2035 2106  ...
    ..$ train.data.ids  : chr [1:24] "GSM42248" "GSM42250" ...
    ..$ test.data.ids   : chr [1:3] "GSM42246" "GSM42262" "GSM42253"
    ..$ train.data.level: Named num [1:24] 1 1 1 1 -1 -1 -1 ...
    .. ..- attr(*, "names")= chr [1:24] "GSM42248" "GSM42250" ...
    ..$ test.data.level : Named num [1:3] 1 -1 1
    .. ..- attr(*, "names")= chr [1:3] "GSM42246" "GSM42262" "GSM42253"
    $ :List of 5
    ..$ feature.ids     : int [1:2204] 370 726 2064 960 1519 2035 751 ...
    ..$ train.data.ids  : chr [1:24] "GSM42246" "GSM42248" ...
    ..$ test.data.ids   : chr [1:3] "GSM42252" "GSM42264" "GSM42257"
    ..$ train.data.level: Named num [1:24] 1 1 1 1 -1 -1 -1 ...
    .. ..- attr(*, "names")= chr [1:24] "GSM42246" "GSM42248" ...
    ..$ test.data.level : Named num [1:3] 1 -1 1
    .. ..- attr(*, "names")= chr [1:3] "GSM42252" "GSM42264"
    $ :List of 5
    ..$ feature.ids     : int [1:2204] 2064 1519 2032 370 1550 2035 805 ...
    ..$ train.data.ids  : chr [1:24] "GSM42246" "GSM42248" ...
    ..$ test.data.ids   : chr [1:3] "GSM42250" "GSM42269" "GSM42270"
    ..$ train.data.level: Named num [1:24] 1 1 1 1 -1 -1 -1 -1 -1 -1 ...
    .. ..- attr(*, "names")= chr [1:24] "GSM42246" "GSM42248" "GSM42252" ...
    ..$ test.data.level : Named num [1:3] 1 1 1
    .. ..- attr(*, "names")= chr [1:3] "GSM42250" "GSM42269" "GSM42270"
    $ :List of 5
```

```
..$ feature.ids     : int [1:2204] 2064 1016 370 2105 1519 611 997 ...
..$ train.data.ids  : chr [1:24] "GSM42246" "GSM42250" "GSM42252" ...
..$ test.data.ids   : chr [1:3] "GSM42248" "GSM42249" "GSM42271"
..$ train.data.level: Named num [1:24] 1 1 1 1 -1 -1 -1 -1 -1 -1 ...
.. ..- attr(*, "names")= chr [1:24] "GSM42246" "GSM42250"  ...
..$ test.data.level : Named num [1:3] 1 1 1
.. ..- attr(*, "names")= chr [1:3] "GSM42248" "GSM42249" "GSM42271"
$ :List of 5
..$ feature.ids     : int [1:2204] 2064 370 2032 1446 1174 2105 ...
..$ train.data.ids  : chr [1:25] "GSM42246" "GSM42248" ...
..$ test.data.ids   : chr [1:2] "GSM42258" "GSM42259"
..$ train.data.level: Named num [1:25] 1 1 1 1 1 -1 -1 -1 ...
.. ..- attr(*, "names")= chr [1:25] "GSM42246" "GSM42248" ...
..$ test.data.level : Named num [1:2] 1 1
.. ..- attr(*, "names")= chr [1:2] "GSM42258" "GSM42259"
$ :List of 5
..$ feature.ids     : int [1:2204] 2064 1913 781 1164 533 370 1914 ...
..$ train.data.ids  : chr [1:25] "GSM42246" "GSM42248" "GSM42250" ...
..$ test.data.ids   : chr [1:2] "GSM42268" "GSM42247"
..$ train.data.level: Named num [1:25] 1 1 1 1 1 -1 -1 -1 ...
.. ..- attr(*, "names")= chr [1:25] "GSM42246" "GSM42248" ...
..$ test.data.level : Named num [1:2] -1 1
.. ..- attr(*, "names")= chr [1:2] "GSM42268" "GSM42247"
$ :List of 5
..$ feature.ids     : int [1:2204] 2064 1519 625 1996 2032 ...
..$ train.data.ids  : chr [1:25] "GSM42246" "GSM42248" ...
..$ test.data.ids   : chr [1:2] "GSM42254" "GSM42266"
..$ train.data.level: Named num [1:25] 1 1 1 1 -1 -1 -1 -1 -1 -1 ...
.. ..- attr(*, "names")= chr [1:25] "GSM42246" "GSM42248" ...
..$ test.data.level : Named num [1:2] 1 -1
.. ..- attr(*, "names")= chr [1:2] "GSM42254" "GSM42266"
```

Value

`results` Return the values stored in the variable.

Source

https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE2280

References

O'Donnell RK, Kupferman M, Wei SJ, Singhal S et al. Gene expression signature predicts lymphatic metastasis in squamous cell carcinoma of the oral cavity. Oncogene 2005 Feb 10;24(7):1244-51. PMID: 15558013

Examples

```
data(results)
## maybe str(results) ; plot(results) ...
```

`sigCVError` *Mean external cross validation (k-fold) error calculation.*

Description

The function "sigCVError()" computes the mean external cross validation (k-fold) errors and the standered deviation of the errors.

Usage

```
sigCVError(i, results, input)
```

Arguments

`i`	number of features/genes from top of the ranked list.
`results`	the results is produced from sigFeatureRanking.enfold() function.
`input`	vector of class labels -1 or 1's (for n samples/patients). n-by-d data matrix to train (n chips/patients, d features/genes).

Details

The features which are produced on the basis of frequency are used here to enumerate mean external cross validation (k-fold) errors and the standard deviation of the errors. Training and test samples are same which are initially produced after splitting the main sample set in to k-fold. In each iteration k-1 folds are considered as training samples and remaining one fold is considered as test samples.In this external cross validation procedure, feature numbers are increased one by one by using the expression values from training dataset as well as test dataset. After that, traning samples are trained to test the tesing samples dynamically. The number of un classified samples are averaged and are called as external cross validation error rate.

Value

`error`	Return the error list.

Author(s)

Pijush Das <topijush@gmail.com>, et al.

References

Zhang, H. H., Ahn, J., Lin, X. and Park, C. (2006). Gene selection using support vector machines with nonconvex penalty. Bioinformatics, 22, pp. 88-95.

See Also

findgacv.scad, predict.penSVM, sim.data

Examples

```
#Example for sigCVError()
#Data set taken from GSE2280
#library(SummarizedExperiment)
#data(ExampleRawData, package="sigFeature")

#x  <- t(assays(ExampleRawData)$counts)
#y  <- colData(ExampleRawData)$sampleLabels

#inputdata <- data.frame(y=as.factor(y) ,x=x)

#For 10 fold cross validation.
#results = sigFeature.enfold(x, y, "kfold",10)

#Find out the frequency of the top 400 features selected by every iteration.
#FeatureBasedonFrequency <- sigFeatureFrequency(results, 400, 400, pf=FALSE)
#str(FeatureBasedonFrequency[1])


#To run the code given bellow will take huge time.
#featsweepSigFe = lapply(1:400, sigCVError, FeatureBasedonFrequency, inputdata)

#Thus the process data is given below.
data("featsweepSigFe")
str(featsweepSigFe[1])
```

`sigFeature` *Significant Feature Selection by using SVM-RFE & t-statistic.*

Description

Significant Feature selected by using SVM-RFE and t-statistic.

Usage

```
sigFeature(X,Y)
```

Arguments

`X`	n-by-d data matrix to train (n samples/patients, d features/genes)
`Y`	vector of class labels -1 or 1's (for n samples/patients)

Details

The idea of "sigFeature" (Significant Feature Selection) begins from the lack of support vector machine recursive feature (SVM-RFE) method. The feature ranked by the SVM-RFE algorithm may or may not be differentially significant among the classes (in case of binary classification). Significant features which have good classification accuracy and which are differentially significant have an impotent role in biological aspect.

In data mining and optimisation, the feature selection is a very active field of research where the SVM-RFE is distinguished as one of the most effective methods. This is a greedy method that only

hopes to find the best possible combination for classification. In this algorithm, greedy method similar to SVM-RFE is used. The prime intention of this algorithm is to enumerate the ranking weights for all the features and sort the features according to weight vectors as the basis for classification. The coefficient and the expression mean differences between two compared groups are used to calculate the weight value of that particular feature. The iteration process is followed by backward removal of the feature. The iteration process is continued until there is only one feature remaining in the dataset. The smallest ranking weight will be removed by the algorithm while the feature variables with significant impact remains. Finally, the feature variables will be listed in the descending order of descriptive difference degree.

Value

returns the feature list.

Author(s)

Pijush Das <topijush@gmail.com>, et al.

References

1. Karatzoglou, Alexandros, David Meyer, and Kurt Hornik. "Support vector machines in R." (2005).

2. O'Donnell RK, Kupferman M, Wei SJ, Singhal S et al. Gene expression signature predicts lymphatic metastasis in squamous cell carcinoma of the oral cavity. Oncogene 2005 Feb 10;24(7):1244-51.

See Also

SVM, predict.penSVM

Examples

```
#Data set taken from GSE2280
library(SummarizedExperiment)
data(ExampleRawData, package="sigFeature")

x  <- t(assays(ExampleRawData)$counts)
y  <- colData(ExampleRawData)$sampleLabels


#Number of features are reduced to minimized the build time.
x <- x[ , 1:100]

#Feature selection with sigFeature functio.
system.time(sigfeatureRankedList <- sigFeature(x,y))
str(sigfeatureRankedList)
```

`sigFeature.enfold` *Significant feature selection with k-fold data.*

Description

After converting the dataset into k-folds the function named "sigFeature.enfold()" is used to select significant features from the classes. The randomization process is used to sub-sample the dataset.

Usage

```
sigFeature.enfold(x, y, CV, CVnumber=0)
```

Arguments

`x`	n-by-d data matrix to train (n chips/patients, d clones/genes)
`y`	vector of class labels -1 or 1 s (for n chips/patiens)
`CV`	the number of folds in case of k-fold cross validation.
`CVnumber`	the number of folds in case of n fold cross validation.

Details

The "sigFeature()" function is further enhanced by incorporating one cross validation methods such as k-fold external cross validation. In this k-fold cross validation procedure k-1 fold are used for selecting the feature and one fold remain untouched which will latter used as test sample set.

Value

`feature.ids`	selected significant features.
`train.data.ids`	training chips/patients ids.
`test.data.ids`	testng chips/patients ids.
`train.data.level`	vector of class labels -1 or 1s (for n chips/patiens) for train da.
`test.data.level`	vector of class labels -1 or 1s (for n chips/patiens) for test da.

Note

This function will compute the feature with cross checking.

Author(s)

Pijush Das <topijush@gmail.com>, et al.

References

Zhang, H. H., Ahn, J., Lin, X. and Park, C. (2006). Gene selection using support vector machines with nonconvex penalty. Bioinformatics, 22, pp. 88-95.

See Also

findgacv.scad, predict.penSVM, sim.data

Examples

```
#Example for sigFeature.enfold()
#Data set taken from GSE2280
#library(SummarizedExperiment)
#data(ExampleRawData, package="sigFeature")

#x  <- t(assays(ExampleRawData)$counts)
#y  <- colData(ExampleRawData)$sampleLabels

#For ten fold external cross validation.
#results = sigFeature.enfold(x,y,"kfold",10)


#Compactly display the internal structure of an R object named "results"
data(results)
str(results)
```

`sigFeatureFrequency` *Arrange the features on the basis of frequency.*

Description

Arrange the features on the basis of frequency.

Usage

```
sigFeatureFrequency(x, results, n, m, pf=FALSE)
```

Arguments

`x`	n-by-d data matrix to train (n samples/patients, d features/genes.
`results`	The "results" variable contains the output produced by the function "sigFeature.enfold()".

```
List of 1
$ :List of 5
    ..$ feature.ids    : int [1:2204] 2064 2031 2035 1573 370  ...
     ..$ train.data.ids  : chr [1:24] "GSM42246" "GSM42248" ...
   ..$ test.data.ids  : chr [1:3] "GSM42263" "GSM42251" "GSM42260"
   ..$ train.data.level: Named num [1:24] 1 1 1 1 1 -1 -1 -1 -1 -1 ...
    .. ..- attr(*, "names")= chr [1:24] "GSM42246" "GSM42248"  ...
     ..$ test.data.level : Named num [1:3] -1 1 1
   .. ..- attr(*, "names")= chr [1:3] "GSM42263" "GSM42251" "GSM42260"
```

`n`	n number of top features which will be taken from each feature lists.
`m`	m number of top features which will be selected on the basis of frequency.
`pf`	this variable used to print the all the output data into a .csv file.

Details

In this example a new function is introduced named as "sigFeatureFrequency()". The main purpose of this function is to arrange the features on the basis of its frequency. In the previous session the dataset is divided into k-folds. Out of which k-1 folds are used for training purpose and one fold is kept for test purpose. Thus each time the algorithm will produce a set of feature lists which finally end up with k number of feature lists. "sigFeatureFrequency()" function is used which will rank all the feature according to its frequency. Details description of this function is given in the help file.

Value

```
List of 1
$ :List of 5
    ..$ feature.ids     : num [1:400] 187 225 246 303 313 370 394 469  ...
    ..$ train.data.ids  : chr [1:24] "GSM42246" "GSM42248" "GSM42250" ...
    ..$ test.data.ids   : chr [1:3] "GSM42263" "GSM42251" "GSM42260"
    ..$ train.data.level: Named num [1:24] 1 1 1 1 1 -1 -1 -1 -1 -1 ...
    .. ..- attr(*, "names")= chr [1:24] "GSM42246" "GSM42248" "GSM42250" ...
    ..$ test.data.level : Named num [1:3] -1 1 1
    .. ..- attr(*, "names")= chr [1:3] "GSM42263" "GSM42251" "GSM42260"
```

Author(s)

Pijush Das <topijush@gmail.com>, et al.

References

Zhang, H. H., Ahn, J., Lin, X. and Park, C. (2006). Gene selection using support vector machines with nonconvex penalty. Bioinformatics, 22, pp. 88-95.

See Also

findgacv.scad, predict.penSVM, sim.data

Examples

```
#Example for sigFeatureFrequency()
#Data set taken from GSE2280
library(SummarizedExperiment)
data(ExampleRawData, package="sigFeature")

x  <- t(assays(ExampleRawData)$counts)
y  <- colData(ExampleRawData)$sampleLabels


#For ten fold external cross validation.
#results = sigFeature.enfold(x,y,"kfold",10)


data(results)
FeatureBasedonFrequency <- sigFeatureFrequency(x, results, 400, 400, pf=FALSE)
str(FeatureBasedonFrequency[1])
```

`sigFeaturePvalue` *Find the p-value of those ranked features by using t-statistic*

Description

This function will compute the p-value of those ranked features between the two classes by using t-statistic.

Usage

```
sigFeaturePvalue(x, y, NumberOfSignificantGene=0, SignificantGeneLilt=0)
```

Arguments

`x`	n-by-d data matrix to train (n chips/patients, d clones/genes)
`y`	vector of class labels -1 or 1\'s (for n chips/patients)
`NumberOfSignificantGene`	Number of the selected features.
`SignificantGeneLilt`	Selected feature list.

Details

This function will calculate the p-value.

Value

returns p-value list.

Author(s)

Pijush Das <topijush@gmail.com>, et al.

References

Peng CH, Liao CT, Peng SC, Chen YJ et al. A novel molecular signature identified by systems genetics approach predicts prognosis in oral squamous cell carcinoma. PLoS One 2011;6(8):e23452. PMID: 21853135

See Also

svm, svm.fs

Examples

```
#Example for sigFeaturePvalue() function
#Data set taken from GSE2280
library(SummarizedExperiment)
data(ExampleRawData, package="sigFeature")

x  <- t(assays(ExampleRawData)$counts)
y  <- colData(ExampleRawData)$sampleLabels
```

```
#Claculating the p-value.
pvalues <- sigFeaturePvalue(x,y)
#Histogram plot of those p-values.
hist(unlist(pvalues),breaks=seq(0,0.08,0.0015),
xlab = "p-value", ylab = "Frequency", main = "")


#Load the process "sigfeatureRankedList" data.
data("sigfeatureRankedList")
#Claculating the p-value.
pvalues <- sigFeaturePvalue(x, y, 50, sigfeatureRankedList)
#Histogram plot of those p value.
hist(unlist(pvalues),breaks=seq(0,0.08,0.0015), xlab =
"p-value", ylab = "Frequency", main = "")


#Load the process "featureRankedList" data.
data("featureRankedList")
#Claculating the p-value.
pvalues <- sigFeaturePvalue(x, y, 50, featureRankedList)
#Histogram plot of those p value.
hist(unlist(pvalues),breaks=seq(0,0.08,0.0015), xlab =
"p-value", ylab = "Frequency", main = "")
```

`sigfeatureRankedList` *Processed output data after using the function named "sigFeature()".*

Description

The variable "sigfeatureRankedList" contains the output of the function named "sigFeature()".

Usage

```
data("sigfeatureRankedList")
```

Format

```
The format is:
int [1:2204] 2064 370 2032 2035 1519 1573 1446 2105 997 611 ...
```

Details

The dataset contains the ranked feature address which can indicate the expression values inside the expression dataset.

Value

`sigfeatureRankedList`
returns the feature list.

Source

https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE2280

References

Guyon, I., et al. (2002) Gene selection for cancer classification using support vector machines, Machine learning, 46, 389-422.

Examples

```
data(sigfeatureRankedList)
## maybe str(sigfeatureRankedList) ; plot(sigfeatureRankedList) ...
```

---

svmrfeFeatureRanking *R implementation of the SVM-RFE algorithm for binary classification problems*

Description

To solve the classification problem with the help of ranking the features an algorithm was proposed by Guyon, Isabelle, et al. named SVM-RFE. In this algorithm the dataset has been trained with SVM linear kernel model and the feature containing the smallest ranking is removed. This criterion is the w value of the decision hyperplane given by the SVM.

Usage

```
svmrfeFeatureRanking(x,y)
```

Arguments

x	x n-by-d data matrix to train (n samples/patients, d clones/genes)
y	y vector of class labels -1 or 1\'s (for n chips/patients)

Details

Adopted from R code: http://www.uccor.edu.ar/busquedas/?txt_palabra=seminarios

Value

returns the feature list.

Note

This function also rank the feature.

Author(s)

Guyon, Isabelle, et al.

References

Guyon, Isabelle, et al. "Gene selection for cancer classification using support vector machines." Machine learning 46.1-3 (2002): 389-422.

Zhang, H. H., Ahn, J., Lin, X. and Park, C. (2006). Gene selection using support vector machines with nonconvex penalty. Bioinformatics, 22, pp. 88-95.

See Also

scadsvc, predict.penSVM, sim.data

Examples

```
#Example for svmrfeFeatureRanking()
#Data set taken from GSE2280
library(SummarizedExperiment)
data(ExampleRawData, package="sigFeature")

x  <- t(assays(ExampleRawData)$counts)
y  <- colData(ExampleRawData)$sampleLabels

x <- x[ ,1:500]

#featureRankedList = svmrfeFeatureRanking(x,y)
print(featureRankedList[1:10])

#Train the data with ranked frature
#library(e1071)
#svmmodel = svm(x[ , featureRankedList[1:50]], y, cost = 10, kernel="linear")
#summary(svmmodel)
```

WritesigFeature *Write the features and sample IDs.*

Description

This function will write the output data produce from the function sigFeatureRanking.enfold.

Usage

```
WritesigFeature(results, x,  fileName="Result")
```

Arguments

results	the object produce by the function named sigFeatureRanking.enfold
x	n-by-d data matrix to train (n chips/patients, d clones/genes).
fileName	name of the output file.

Details

This function will write the variables.

Value

results output file.

Author(s)

Pijush Das <topijush@gmail.comn>, et al.

References

Becker, N., Werft, W., Toedt, G., Lichter, P. and Benner, A.(2009) PenalizedSVM: a R-package for feature selection SVM classification, Bioinformatics, 25(13),p 1711-1712

See Also

predict.penSVM, svm (in package e1071)

Examples

```
#Example for WritesigFeature()
#Data set taken from GSE2280
library(SummarizedExperiment)
data(ExampleRawData, package="sigFeature")

x  <- t(assays(ExampleRawData)$counts)
y  <- colData(ExampleRawData)$sampleLabels


#For 10 fold cross validation.
#results = sigFeature.enfold(x,y,"kfold",10)


#to write the output
#data(results)
#WritesigFeature(results, x)
```

Index