Package 'BioMM'

October 16, 2019

Type Package

Title BioMM: Biological-informed Multi-stage Machine learning framework for phenotype prediction using omics data

Version 1.0.0

Date 2019-05-01

Author Junfang Chen and Emanuel Schwarz

Maintainer Junfang Chen <junfang.chen33@gmail.com>

Description The identification of reproducible biological patterns from high-dimensional omics data is a key factor in understanding the biology of complex disease or traits. Incorporating prior biological knowledge into machine learning is an important step in advancing such research. We have proposed a biologically informed multi-stage machine learing framework termed BioMM specifically for phenotype prediction based on omics-scale data where we can evaluate different machine learning models with various prior biological meta information.

Imports stats, utils, grDevices, lattice, BiocParallel, glmnet, rms, nsprcomp, ranger, e1071, variancePartition, ggplot2

Depends R (>= 3.6)

Suggests BiocStyle, knitr, RUnit, BiocGenerics

VignetteBuilder knitr

biocViews Genetics, Classification, Regression, Pathways, GO, Software

Encoding UTF-8

LazyData true

License GPL-3

RoxygenNote 6.1.1

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

git_branch RELEASE_3_9

git_last_commit d531675

git_last_commit_date 2019-05-02

Date/Publication 2019-10-15

R topics documented:

`baseGLMnet` *Prediction by generalized linear regression models*

Description

Prediction by generalized regression models with lasso or elastic net regularization.

Usage

```
baseGLMnet(trainData, testData, predMode = c("classification",
  "probability", "regression"), paramlist = list(family = "binomial",
  alpha = 0.5, typeMeasure = "mse", typePred = "class"))
```

Arguments

`trainData`	The input training dataset. The first column is named the 'label'.
`testData`	The input test dataset. The first column is named the 'label'.
`predMode`	The prediction mode. Available options are c('classification', 'probability', 'regression').
`paramlist`	A set of model parameters defined in an R list object. The valid option: list(family, alpha, typeMeasure, typePred).

1. 'family': Response type: 'gaussian','binomial','poisson', 'multinomial','cox','mgaussian'. (Default: 'binomial')
2. 'alpha': The elastic net mixing parameter, with 0<=alpha<= 1.

3. 'typeMeasure': error metrics for internal cross-validation. 'mse' uses squared loss; 'deviance' uses actual deviance; 'mae' uses mean absolute error; 'class' gives misclassification error; 'auc' (for two-class logistic regression ONLY) gives area under the ROC curve.
4. 'typePred': The type of prediction: 'response' and 'class'. (Default: 'class' for binary classification)

Value

The predicted output for the test data.

Author(s)

Junfang Chen

Examples

```
## Load data
methylfile <- system.file('extdata', 'methylData.rds', package='BioMM')
methylData <- readRDS(methylfile)
dataY <- methylData[,1]
## select a subset of genome-wide methylation data at random
methylSub <- data.frame(label=dataY, methylData[,c(2:2001)])
trainIndex <- sample(nrow(methylSub), 30)
trainData = methylSub[trainIndex,]
testData = methylSub[-trainIndex,]
library(glmnet)
## classification
predY <- baseGLMnet(trainData, testData,
                    predMode='classification',
                    paramlist=list(family='binomial', alpha=0.5,
                                   typeMeasure='mse', typePred='class'))
testY <- testData[,1]
accuracy <- classifiACC(dataY=testY, predY=predY)
print(accuracy)
```

`baseModel` *Base supervised machine learning models for prediction*

Description

Prediction using different supervised machine learning models.

Usage

```
baseModel(trainData, testData, classifier = c("randForest", "SVM",
  "glmnet"), predMode = c("classification", "probability", "regression"),
  paramlist)
```

Arguments

trainData	The input training dataset. The first column is the label or the output. For binary classes, 0 and 1 are used to indicate the class member.
testData	The input test dataset. The first column is the label or the output. For binary classes, 0 and 1 are used to indicate the class member.
classifier	Machine learning classifiers. Available options are c('randForest', 'SVM', 'glmnet').
predMode	The prediction mode. Available options are c('classification', 'probability', 'regression'). 'probability' is currently only for 'randForest'.
paramlist	A set of model parameters defined in an R list object. See more details for each individual model.

Value

The predicted output for the test data.

Author(s)

Junfang Chen

Examples

```
## Load data
methylfile <- system.file('extdata', 'methylData.rds', package='BioMM')
methylData <- readRDS(methylfile)
dataY <- methylData[,1]
## select a subset of genome-wide methylation data at random
methylSub <- data.frame(label=dataY, methylData[,c(2:2001)])
trainIndex <- sample(nrow(methylSub), 30)
trainData = methylSub[trainIndex,]
testData = methylSub[-trainIndex,]
library(ranger)
set.seed(123)
predY <- baseModel(trainData, testData,
                   classifier='randForest',
                   predMode='classification',
                   paramlist=list(ntree=300, nthreads=20))
print(table(predY))
testY <- testData[,1]
accuracy <- classifiACC(dataY=testY, predY=predY)
print(accuracy)
```

---

baseRandForest *Prediction by random forest*

Description

Prediction by random forest with different settings: 'probability', 'classification' and 'regression'.

Usage

```
baseRandForest(trainData, testData, predMode = c("classification",
  "probability", "regression"), paramlist = list(ntree = 2000, nthreads =
  20))
```

Arguments

trainData	The input training dataset. The first column is the label or the output. For binary classes, 0 and 1 are used to indicate the class member.
testData	The input test dataset. The first column is the label or the output. For binary classes, 0 and 1 are used to indicate the class member.
predMode	The prediction mode. Available options are c('probability', 'classification', 'regression').
paramlist	A set of model parameters defined in an R list object. The valid option: list(ntree, nthreads). 'ntree' is the number of trees used. The defaul is 2000. 'nthreads' is the number of threads used for computation. The default is 20.

Value

The predicted output for the test data.

Author(s)

Junfang Chen

Examples

```
## Load data
methylfile <- system.file('extdata', 'methylData.rds', package='BioMM')
methylData <- readRDS(methylfile)
dataY <- methylData[,1]
## test a subset of genome-wide methylation data at random
methylSub <- data.frame(label=dataY, methylData[,c(2:2001)])
trainIndex <- sample(nrow(methylSub), 30)
trainData = methylSub[trainIndex,]
testData = methylSub[-trainIndex,]
library(ranger)
predY <- baseRandForest(trainData, testData,
                        predMode='classification',
                        paramlist=list(ntree=300, nthreads=20))
testY <- testData[,1]
accuracy <- classifiACC(dataY=testY, predY=predY)
print(accuracy)
```

---

baseSVM *Prediction by SVM*

Description

Prediction by support vector machine (SVM) with two different settings: 'classification' and 'regression'.

Usage

```
baseSVM(trainData, testData, predMode = c("classification",
  "probability", "regression"), paramlist = list(tuneP = TRUE, kernel =
  "radial", gamma = 10^(-3:-1), cost = 10^(-2:2)))
```

Arguments

trainData	The input training dataset. The first column is the label or the output. For binary classes, 0 and 1 are used to indicate the class member.
testData	The input test dataset. The first column is the label or the output. For binary classes, 0 and 1 are used to indicate the class member.
predMode	The prediction mode. Available options are c('classification', 'probability', 'regression').
paramlist	A set of model parameters defined in an R list object. The valid option: list(kernel, gamma, cost, tuneP).

1. 'tuneP': a logical value indicating if hyperparameter tuning should be conducted or not. The default is FALSE.
2. 'kernel': options are c('linear', 'polynomial', 'radial', 'sigmoid'). The defaut is 'radial'.
3. 'gamma': the parameter needed for all kernels except 'linear'. If tuneP is TRUE, more than one value is suggested.
4. 'cost': is the cost of constraints violation. If tuneP is TRUE, more than one value is suggested.

Details

Hyperparameter tuning is recommended in many biological data mining applications. The best parameters can be determined via an internal cross validation.

Value

The predicted output for the test data.

Author(s)

Junfang Chen

See Also

svm

Examples

```
## Load data
methylfile <- system.file('extdata', 'methylData.rds', package='BioMM')
methylData <- readRDS(methylfile)
dataY <- methylData[,1]
## select a subset of genome-wide methylation data at random
methylSub <- data.frame(label=dataY, methylData[,c(2:2001)])
trainIndex <- sample(nrow(methylSub), 30)
trainData = methylSub[trainIndex,]
```

```
testData = methylSub[-trainIndex,]
library(e1071)
predY <- baseSVM(trainData, testData,
                 predMode='classification',
                 paramlist=list(tuneP=FALSE, kernel='radial',
                                gamma=10^(-3:-1), cost=10^(-3:1)))
testY <- testData[,1]
accuracy <- classifiACC(dataY=testY, predY=predY)
print(accuracy)
```

`BioMM` *BioMM end-to-end prediction*

Description

End-to-end prediction by BioMM framework using either supervised or unsupervised learning at stage-1, then supervised learning at stage-2.

Usage

```
BioMM(trainData, testData, stratify = c("gene", "pathway", "chromosome"),
  pathlistDB, featureAnno, restrictUp, restrictDown, minPathSize,
  supervisedStage1 = TRUE, typePCA, resample1 = "BS",
  resample2 = "CV", dataMode = "allTrain", repeatA1, repeatA2,
  repeatB1, repeatB2, nfolds, FSmethod1, FSmethod2, cutP1, cutP2, fdr1,
  fdr2, FScore = MulticoreParam(), classifier1, classifier2, predMode1,
  predMode2, paramlist1, paramlist2, innerCore = MulticoreParam(),
  outFileA2 = NULL, outFileB2 = NULL)
```

Arguments

`trainData`	The input training dataset. The first column is the label or the output. For binary classes, 0 and 1 are used to indicate the class member.
`testData`	The input test dataset. The first column is the label or the output. For binary classes, 0 and 1 are used to indicate the class member.
`stratify`	The stratification method. Valid options are c('gene', 'pathway', 'chromosome').
`pathlistDB`	A list of pathways with pathway IDs and their corresponding genes ('entrezID' is used). This is only used for pathway-based stratification (only `stratify` is 'pathway').
`featureAnno`	The annotation data stored in a data.frame for probe mapping. It must have at least two columns named 'ID' and 'entrezID'. If it's NULL, then the input probe is from the transcriptomic data. (Default: NULL)
`restrictUp`	The upper-bound of the number of probes or genes in each biological stratified block.
`restrictDown`	The lower-bound of the number of probes or genes in each biological stratified block.
`minPathSize`	The minimal defined pathway size after mapping your own data to GO database. This is only used for pathway-based stratification (only `stratify` is 'pathway').

`supervisedStage1`
: A logical value. If TRUE, then supervised learning models are applied; if FALSE, unsupervised learning.

`typePCA`
: the type of PCA. Available options are c('regular', 'sparse').

`resample1`
: The resampling methods at stage-1. Valid options are 'CV' and 'BS'. 'CV' for cross validation and 'BS' for bootstrapping resampling. The default is 'BS'.

`resample2`
: The resampling methods at stage-2. Valid options are 'CV' and 'BS'. 'CV' for cross validation and 'BS' for bootstrapping resampling. The default is 'CV'.

`dataMode`
: The mode of data used at stage-1. 'subTrain' or 'allTrain'. This is only applicable for bootstrapping resampling. (Default: allTrain).

`repeatA1`
: The number of repeats N is used during resampling procedure. Repeated cross validation or multiple boostrapping is performed if N >=2. One can choose 10 repeats for 'CV' and 100 repeats for 'BS'.

`repeatA2`
: The number of repeats N is used during resampling prediction. The default is 1 for 'CV'.

`repeatB1`
: The number of repeats N is used for generating stage-2 test data prediction scores.

`repeatB2`
: The number of repeats N is used for test data prediction. The default is 1.

`nfolds`
: The number of folds is defined for cross validation.

`FSmethod1`
: Feature selection methods at stage-1. Available options are c(NULL, 'positive', 'wilcox.test', 'cor.test', 'chisq.test', 'posWilcox', or 'top10pCor').

`FSmethod2`
: Feature selection methods at stage-2. Available options are c(NULL, 'positive', 'wilcox.test', 'cor.test', 'chisq.test', 'posWilcox', or 'top10pCor').

`cutP1`
: The cutoff used for p value thresholding at stage-1. Commonly used cutoffs are c(0.5, 0.1, 0.05, 0.01, etc). The default is 0.05. Commonly used cutoffs are c(0.5, 0.1, 0.05, 0.01, etc). The default is 0.05.

`cutP2`
: The cutoff used for p value thresholding at stage-2.

`fdr1`
: Multiple testing correction method at stage-1. Available options are c(NULL, 'fdr', 'BH', 'holm', etc). See also `p.adjust`. The default is NULL.

`fdr2`
: Multiple testing correction method at stage-2. Available options are c(NULL, 'fdr', 'BH', 'holm', etc). See also `p.adjust`. The default is NULL.

`FScore`
: The number of cores used for feature selection.

`classifier1`
: Machine learning classifiers at stage-1.

`classifier2`
: Machine learning classifiers at stage-2.

`predMode1`
: The prediction mode at stage-1. Available options are c('probability', 'classification', 'regression').

`predMode2`
: The prediction mode at stage-2. Available options are c('probability', 'classification', 'regression').

`paramlist1`
: A list of model parameters at stage-1.

`paramlist2`
: A list of model parameters at stage-2.

`innerCore`
: The number of cores used for computation.

`outFileA2`
: The file name of prediction metrics based on resampling with the '.csv' file extension. If it's provided, then the result will be saved. The default is NULL.

`outFileB2`
: The file name of independent test prediction metrics with the '.csv' file extension. If it's provided, then the result will be saved. The default is NULL.

Details

Stage-2 training data can be learned either using bootstrapping or cross validation resampling methods in the supervised learning settting. Stage-2 test data is learned via independent test set prediction.

Value

The CV or BS prediction performance for the training data and test set prediction performance if `testData` is given.

References

Chen, J., & Schwarz, E. (2017). BioMM: Biologically-informed Multi-stage Machine learning for identification of epigenetic fingerprints. arXiv preprint arXiv:1712.00336.

Perlich, C., & Swirszcz, G. (2011). On cross-validation and stacking: Building seemingly predictive models on random data. ACM SIGKDD Explorations Newsletter, 12(2), 11-15.

See Also

`BioMMreconData`; `BioMMstage1pca`; `BioMMstage2pred`

Examples

```
## Load data
methylfile <- system.file('extdata', 'methylData.rds', package='BioMM')
methylData <- readRDS(methylfile)
## Annotation files for Mapping CpGs into chromosome
probeAnnoFile <- system.file('extdata', 'cpgAnno.rds', package='BioMM')
probeAnno <- readRDS(file=probeAnnoFile)
supervisedStage1=TRUE
classifier1=classifier2 <- 'randForest'
predMode1=predMode2 <- 'classification'
paramlist1=paramlist2 <- list(ntree=300, nthreads=30)
library(BiocParallel)
library(ranger)
param1 <- MulticoreParam(workers = 2)
param2 <- MulticoreParam(workers = 20)
## Not Run
## result <- BioMM(trainData=methylData, testData=NULL,
##                 stratify='chromosome', pathlistDB, featureAnno=probeAnno,
##                 restrictUp=10, restrictDown=200, minPathSize=10,
##                 supervisedStage1, typePCA='regular',
##                 resample1='BS', resample2='CV', dataMode='allTrain',
##                 repeatA1=20, repeatA2=1, repeatB1=20, repeatB2=1,
##                 nfolds=10, FSmethod1=NULL, FSmethod2=NULL,
##                 cutP1=0.1, cutP2=0.1, fdr1=NULL, fdr2=NULL, FScore=param1,
##                 classifier1, classifier2, predMode1, predMode2,
##                 paramlist1, paramlist2, innerCore=param2,
##                 outFileA2=NULL, outFileB2=NULL)
```

BioMMreconData *Reconstruct stage-2 data by supervised machine learning prediction*

Description

Reconstruct stage-2 data by supervised machine learning prediction.

Usage

```
BioMMreconData(trainDataList, testDataList, resample = "BS", dataMode,
  repeatA, repeatB, nfolds, FSmethod, cutP, fdr,
  FScore = MulticoreParam(), classifier, predMode, paramlist,
  innerCore = MulticoreParam(), outFileA = NULL, outFileB = NULL)
```

Arguments

trainDataList	The input training data list containing ordered collections of matrices.
testDataList	The input test data list containing ordered collections of matrices.
resample	The resampling methods. Valid options are 'CV' and 'BS'. 'CV' for cross validation and 'BS' for bootstrapping resampling. The default is 'BS'.
dataMode	The mode of data used. 'subTrain' or 'allTrain'.
repeatA	The number of repeats N is used during resampling procedure. Repeated cross validation or multiple boostrapping is performed if N >=2. One can choose 10 repeats for 'CV' and 100 repeats for 'BS'.
repeatB	The number of repeats N is used for generating test data prediction scores.
nfolds	The number of folds is defined for cross validation.
FSmethod	Feature selection methods. Available options are c(NULL, 'positive', 'wilcox.test', 'cor.test', 'chisq.test', 'posWilcox', or 'top10pCor').
cutP	The cutoff used for p value thresholding. Commonly used cutoffs are c(0.5, 0.1, 0.05, 0.01, etc). The default is 0.05.
fdr	Multiple testing correction method. Available options are c(NULL, 'fdr', 'BH', 'holm', etc). See also p.adjust. The default is NULL.
FScore	The number of cores used for feature selection, if parallel computing needed.
classifier	Machine learning classifiers.
predMode	The prediction mode. Available options are c('probability', 'classification', 'regression').
paramlist	A set of model parameters defined in an R list object.
innerCore	The number of cores used for computation.
outFileA	The file name of stage-2 training data with the '.rds' file extension. If it's provided, then the result will be saved in this file. The default is NULL.
outFileB	The file name of stage-2 training data with the '.rds' file extension. If it's provided, then the result will be saved in this file. The default is NULL.

Details

Stage-2 training data can be learned either using bootstrapping or cross validation resampling methods. Stage-2 test data is learned via independent test set prediction.

Value

The predicted stage-2 training data and also stage-2 test data, if 'testDataList' provided. If outFileA and outFileB are provided, then the results will be stored in the files.

Author(s)

Junfang Chen

Examples

```
## Load data
methylfile <- system.file('extdata', 'methylData.rds', package='BioMM')
methylData <- readRDS(methylfile)
## Annotation files for Mapping CpGs into chromosome
probeAnnoFile <- system.file('extdata', 'cpgAnno.rds', package='BioMM')
probeAnno <- readRDS(file=probeAnnoFile)
## Mapping CpGs into Chromosome
dataList <- omics2chrlist(data=methylData, probeAnno)
length(dataList)
library(ranger)
library(BiocParallel)
param1 <- MulticoreParam(workers = 1)
param2 <- MulticoreParam(workers = 20)
## Not Run
## stage2data <- BioMMreconData(trainDataList=dataList, testDataList=NULL,
##                              resample='CV', dataMode='allTrain',
##                              repeatA=1, repeatB=1, nfolds=10,
##                              FSmethod=NULL, cutP=0.1,
##                              fdr=NULL, FScore=param1,
##                              classifier='randForest',
##                              predMode='classification',
##                              paramlist=list(ntree=300, nthreads=20),
##                              innerCore=param2, outFileA=NULL, outFileB=NULL)
## print(dim(stage2data))
## print(head(stage2data[,1:5]))
```

BioMMstage1pca *Reconstruct stage-2 data by PCA*

Description

Stage-2 data reconstruction by regular or sparse constrained principal component analysis (PCA).

Usage

```
BioMMstage1pca(trainDataList, testDataList, typeMode = "regular",
  topPC = 1, innerCore = MulticoreParam(), outFileA = NULL,
  outFileB = NULL)
```

Arguments

trainDataList	The input training data list containing ordered collections of matrices.
testDataList	The input test data list containing ordered collections of matrices.
typeMode	The type of PCA prediction mode. Available options are c('regular', 'sparse'). (Default: regular)
topPC	The number of top PCs selected. The default is 1, i.e. the first PC.
innerCore	The number of cores used for computation.
outFileA	The file name of stage-2 training data with the '.rds' file extension. If it's provided, then the result will be saved in this file. The default is NULL.
outFileB	The file name of stage-2 training data with the '.rds' file extension. If it's provided, then the result will be saved in this file. The default is NULL.

Value

The predicted stage-2 training data and also stage-2 test data if 'testDataList' provided. If outFileA and outFileB are provided then the results will be stored in the files.

Author(s)

Junfang Chen

Examples

```
## Load data
methylfile <- system.file('extdata', 'methylData.rds', package='BioMM')
methylData <- readRDS(methylfile)
## Annotation files for Mapping CpGs into chromosome
probeAnnoFile <- system.file('extdata', 'cpgAnno.rds', package='BioMM')
probeAnno <- readRDS(file=probeAnnoFile)
## Mapping CpGs into Chromosome
dataList <- omics2chrlist(data=methylData, probeAnno)
length(dataList)
library(BiocParallel)
param <- MulticoreParam(workers = 10)
stage2data <- BioMMstage1pca(trainDataList=dataList, testDataList=NULL,
                             typeMode='regular', topPC=1,
                             innerCore=param, outFileA=NULL, outFileB=NULL)
print(dim(stage2data))
print(head(stage2data[,1:5]))
```

BioMMstage2pred *Prediction performance for stage-2 data using supervised machine learning*

Description

Prediction performance for reconstructed stage-2 data using supervised machine learning with feature selection methods.

Usage

```
BioMMstage2pred(trainData, testData, resample = "CV", dataMode,
  repeatA = 1, repeatB = 1, nfolds, FSmethod, cutP, fdr,
  FScore = MulticoreParam(), classifier, predMode, paramlist,
  innerCore = MulticoreParam(), outFileA = NULL, outFileB = NULL)
```

Arguments

trainData	The input training dataset (stage-2 data). The first column is the label or the output. For binary classes, 0 and 1 are used to indicate the class member.
testData	The input test dataset (stage-2 data). The first column is the label or the output. For binary classes, 0 and 1 are used to indicate the class member.
resample	The resampling methods. Valid options are 'CV' and 'BS'. 'CV' for cross validation and 'BS' for bootstrapping resampling. The default is 'CV'.
dataMode	The mode of data used. 'subTrain' or 'allTrain'.
repeatA	The number of repeats N is used during resampling prediction. The default is 1.
repeatB	The number of repeats N is used for test data prediction. The default is 1.
nfolds	The number of folds is defined for cross validation.
FSmethod	Feature selection methods. Available options are c(NULL, 'positive', 'wilcox.test', 'cor.test', 'chisq.test', 'posWilcox', or 'top10pCor').
cutP	The cutoff used for p value thresholding. Commonly used cutoffs are c(0.5, 0.1, 0.05, 0.01, etc). The default is 0.05.
fdr	Multiple testing correction method. Available options are c(NULL, 'fdr', 'BH', 'holm', etc). See also p.adjust. The default is NULL.
FScore	The number of cores used for feature selection if parallel computing needed.
classifier	Machine learning classifiers.
predMode	The prediction mode. Available options are c('probability', 'classification', 'regression').
paramlist	A set of model parameters defined in an R list object.
innerCore	The number of cores used for computation.
outFileA	The file name of prediction metrics based on resampling with the '.csv' file extension. If it's provided, then the result will be saved. The default is NULL.
outFileB	The file name of independent test prediction metrics with the '.csv' file extension. If it's provided, then the result will be saved. The default is NULL.

Details

Stage-2 prediction is performed typically using positively correlated features. Since negative associations likely reflect random effects in the underlying data

Value

The CV or BS prediction performance for stage-2 training data and test set prediction performance for stage-2 test data if the test set is given.

Author(s)

Junfang Chen

References

Perlich, C., & Swirszcz, G. (2011). On cross-validation and stacking: Building seemingly predictive models on random data. ACM SIGKDD Explorations Newsletter, 12(2), 11-15.

`classifiACC` *Compute the classification accuracy*

Description

Compute the classification accuracy for the binary classification problem.

Usage

```
classifiACC(dataY, predY)
```

Arguments

`dataY`	The observed outcome.
`predY`	The predicted outcome.

Value

The classification accuracy in terms of percentage.

Author(s)

Junfang Chen

Examples

```
## Load data
methylfile <- system.file('extdata', 'methylData.rds', package='BioMM')
methylData <- readRDS(methylfile)
dataY <- methylData[,1]
methylSub <- data.frame(label=dataY, methylData[,c(2:1001)])
library(ranger)
library(BiocParallel)
param1 <- MulticoreParam(workers = 1)
param2 <- MulticoreParam(workers = 10)
predY <- predByCV(methylSub, repeats=1, nfolds=10,
                  FSmethod=NULL, cutP=0.1,
                  fdr=NULL, FScore=param1,
                  classifier='randForest',
                  predMode='classification',
                  paramlist=list(ntree=300, nthreads=1),
                  innerCore=param2)
accuracy <- classifiACC(dataY=dataY, predY=predY)
print(accuracy)
```

`getDataAfterFS` *Return the data after feature selection*

Description

Get the new data set after performing feature selection on the input training and test data.

Usage

```
getDataAfterFS(trainData, testData, FSmethod, cutP = 0.1, fdr = NULL,
  FScore = MulticoreParam())
```

Arguments

`trainData`	The input training dataset. The first column is the label.
`testData`	The input test dataset. The first column is the label.
`FSmethod`	Feature selection methods. Available options are c(NULL, 'positive', 'wilcox.test', 'cor.test', 'chisq.test', 'posWilcox', or 'top10pCor'). 'positive' is the positively outcome-associated features using Pearson correlation method. 'posWilcox' is the positively outcome-associated features using Pearson correlation method together with 'wilcox.text' method. 'top10pCor' is the top 10 outcome-associcated features. This is useful when no features can be picked during stringent feature selection procedure.
`cutP`	The cutoff used for p value thresholding. It can be any value between 0 and 1. Commonly used cutoffs are c(0.5, 0.1, 0.05, 0.01, etc.). The default is 0.1.
`fdr`	Multiple testing correction method. Available options are c(NULL, 'fdr', 'BH', 'holm' etc). See also `p.adjust`. The default is NULL.
`FScore`	The number of cores used for some feature selection methods. The default is 10.

Details

Parallel computing is helpful if your input data is high dimensional. For 'cutP', a soft thresholding of 0.1 may be favorable than more stringent p value cutoff because the features with small effect size can be taken into consideration for downstream analysis. However, for high dimensional (e.g. p > 10,000) data, many false positive features may exist, thus, rigorous p value thresholding should be applied. 'chisq.test' is suggested for GWAS data due to the binary/discrete input and output.

Value

Both training and test data (if provided) with reduced number of features in the data are returned if feature selection method is applied. If no feature can be found during feature selection procedure, then the output is NULL.

Author(s)

Junfang Chen

Examples

```
## Load data
methylfile <- system.file('extdata', 'methylData.rds', package='BioMM')
methylData <- readRDS(methylfile)
trainIndex <- sample(nrow(methylData), 20)
trainData = methylData[trainIndex,]
testData = methylData[-trainIndex,]
## Feature selection
library(BiocParallel)
param <- MulticoreParam(workers = 2)
datalist <- getDataAfterFS(trainData, testData, FSmethod=NULL,
                           cutP=0.1, fdr=NULL, FScore=param)
trainDataSub <- datalist[[1]]
testDataSub <- datalist[[2]]
print(dim(trainData))
print(dim(trainDataSub))
```

`getMetrics` *Compute the evaluation metrics*

Description

Compute the evaluation metrics in the classification setting: P value based on chi-square test (pv), pearson correlation coefficient (cor), area under curve (AUC), classification accuracy (ACC) and the pseudo R square (R2).

Usage

```
getMetrics(dataY, predY)
```

Arguments

`dataY`	The observed outcome.
`predY`	The predicted outcome.

Details

If all samples are predicted into one class, then we assign R2=0, cor=0, and AUC=0.5.

Value

A set of metrics for model evaluation: pv, cor, AUC, ACC and R2.

Author(s)

Junfang Chen

Examples

```
## Load data
methylfile <- system.file('extdata', 'methylData.rds', package='BioMM')
methylData <- readRDS(methylfile)
dataY <- methylData[,1]
methylSub <- data.frame(label=dataY, methylData[,c(2:1001)])
library(ranger)
library(rms)
library(BiocParallel)
param1 <- MulticoreParam(workers = 1)
param2 <- MulticoreParam(workers = 10)
predY <- predByCV(methylSub, repeats=1, nfolds=10,
                  FSmethod=NULL, cutP=0.1,
                  fdr=NULL, FScore=param1,
                  classifier='randForest',
                  predMode='classification',
                  paramlist=list(ntree=300, nthreads=20),
                  innerCore=param2)
accuracy <- getMetrics(dataY=dataY, predY=predY)
print(accuracy)
```

---

omics2chrlist *Map individual probes into chromosome*

Description

Map a set of individual probes from different omics (i.e. SNPs, gene expression probes, CpGs etc.) into the corresponding chromosomes.

Usage

```
omics2chrlist(data, probeAnno)
```

Arguments

data	The input dataset (either data.frame or matrix). Rows are the samples, columns are the probes/genes, except that the first column is the label.
probeAnno	The annotation data stored in a data.frame for probe mapping. It must have at least two columns named 'ID' and 'chr'.

Value

A list of matrices with chromosome IDs as the associated list member names. For each matrix, rows are the samples and columns are the probe names, except that the first column is named 'label'.

Examples

```
## Load data from DNA methylation
methylfile <- system.file('extdata', 'methylData.rds', package='BioMM')
methylData <- readRDS(methylfile)
## Annotation files for Mapping CpGs into chromosome
```

```
probeAnnoFile <- system.file('extdata', 'cpgAnno.rds', package='BioMM')
probeAnno <- readRDS(file=probeAnnoFile)
## Mapping CpGs into Chromosome
dataList <- omics2chrlist(data=methylData, probeAnno)
length(dataList)
```

`omics2genelist` *Map individual probes into gene*

Description

Map a set of individual probes from different omics (i.e. SNPs and CpGs etc.) into genes.

Usage

```
omics2genelist(data, featureAnno, restrictUp = 500, restrictDown = 5)
```

Arguments

`data`	The input dataset (either data.frame or matrix). Rows are the samples, columns are the probes, except that the first column is the label.
`featureAnno`	The annotation data stored in a data.frame for probe mapping. It must have at least two columns named 'ID' and 'entrezID'.
`restrictUp`	The upper-bound of the number of probes in each gene. The default is 500.
`restrictDown`	The lower-bound of the number of probes in each gene. The default is 5.

Details

This function is not applicable for transcriptomic data. The data types including DNA methylation and GWAS often have several probes within one gene, therefore gene-based stratification is feasible

Value

A list of matrices with gene IDs as the associated list member names. For each matrix, rows are the samples and columns are the probe names, except that the first column is named 'label'.

Examples

```
## Load data from DNA methylation
methylfile <- system.file('extdata', 'methylData.rds', package='BioMM')
methylData <- readRDS(methylfile)
## Annotation files for Mapping CpGs into genes
featureAnnoFile <- system.file('extdata', 'cpgAnno.rds', package='BioMM')
featureAnno <- readRDS(file=featureAnnoFile)
## Mapping CpGs into gene list
## Not run
## dataList <- omics2genelist(data=methylData, featureAnno,
##                            restrictUp=100, restrictDown=5)
## length(dataList)
```

omics2pathlist *Map individual probes into pathway*

Description

Map a set of individual probes from different omics (i.e. SNPs, gene expression probes, CpGs etc.) into pathway such as Gene Ontology (GO) categories and KEGG.

Usage

```
omics2pathlist(data, pathlistDB, featureAnno = NULL, restrictUp = 200,
  restrictDown = 10, minPathSize = 2)
```

Arguments

data	The input dataset (either data.frame or matrix). Rows are the samples, columns are the probes/genes, except that the first column is the label. If it's transcriptomic data, gene ID is the 'entrezID'.
pathlistDB	A list of pathways with pathway IDs and their corresponding genes ('entrezID' is used).
featureAnno	The annotation data stored in a data.frame for probe mapping. It must have at least two columns named 'ID' and 'entrezID'. If it's NULL, then the input probe is from transcriptomic data.
restrictUp	The upper-bound of the number of genes in each pathway. The default is 200.
restrictDown	The lower-bound of the number of genes in each pathway. The default is 10.
minPathSize	The minimal required number of probes in each pathway after mapping the input data to pathlistDB.

Details

If gene expression data is the input, then featureAnno is NULL,since the gene IDs are already defined as column names of the data. Since online database is updated from time to time, it is adivsed to make sure that the study database (e.g. pathlistDB) is frozen at particular time for reproducing the results. The number of genes in each pathway can be restricted for downstream analysis because too small pathways are sparsely distributed, and too large pathways are often computationally intensive, and likely nonspecific.

Value

A list of matrices with pathway IDs as the associated list member names. For each matrix, rows are the samples and columns are the probe names, except that the first column is named 'label'.

Examples

```
## Load data from DNA methylation
methylfile <- system.file('extdata', 'methylData.rds', package='BioMM')
methylData <- readRDS(methylfile)
## Annotation files for Mapping CpGs into pathways
pathlistDBfile <- system.file('extdata', 'goDB.rds', package='BioMM')
featureAnnoFile <- system.file('extdata', 'cpgAnno.rds', package='BioMM')
```

```
pathlistDB <- readRDS(file=pathlistDBfile)
featureAnno <- readRDS(file=featureAnnoFile)
## To reduce runtime
pathlistDB <- pathlistDB[1:20]
## Mapping CpGs into pathway list
dataList <- omics2pathlist(data=methylData,
                                pathlistDB, featureAnno,
                                restrictUp=100, restrictDown=20,
                                minPathSize=10)
length(dataList)
```

`plotRankedFeature` *Plot top outcome-associated features*

Description

Plot top ranked outcome-associated features from stage-2 data. The ranking criteria are based on metrics such as Nagelkerke pseudo R-square.

Usage

```
plotRankedFeature(data, posF = TRUE, topF = 10, blocklist,
  stratify = c("gene", "pathway", "chromosome"), rankMetric = c("cor",
  "AUC", "ACC", "R2", "size"), colorMetric = c("cor", "AUC", "ACC", "R2",
  "size"), core = MulticoreParam(), fileName = NULL)
```

Arguments

`data`	The input stage-2 data (either data.frame or matrix). Rows are the samples, columns are gene IDs, or pathway names or chromosome IDs, except that the first column is the label (the outcome).
`posF`	A logical value indicating if only positively outcome-associated features should be used. (Default: TRUE)
`topF`	The top ranked number of features at stage-2 (`topF` >= 2). (Default: 10)
`blocklist`	A list of matrices with block IDs as the associated list member names. The block IDs identical to the stage-2 feature names. The block can be gene, pathway or chromosome. For each matrix, rows are the samples and columns are the probe names, except that the first column is named 'label'. See also `omics2genelist`; `omics2pathlist`; `omics2chrlist`
`stratify`	A string. The applied stratification method to generate `blocklist`. Valid options are c('gene', 'pathway', 'chromosome').
`rankMetric`	A string representing the metrics used for ranking. Valid options are c('cor', 'AUC', 'ACC', 'R2', 'size'). 'size' is the block size.
`colorMetric`	A string representing the metric used to color the plot. Valid options are c('cor', 'AUC', 'ACC', 'R2', 'size'). 'size' is the block size.
`core`	The number of cores used for computation. (Default: 10)
`fileName`	The plot file name. (Default: 'plottopF.png')

Details

If the argument posF is TRUE, and no positively outcome-associated features are present in stage-2 data , then an error is reported. In addition, if topF is bigger than the number of positively outcome-associated features, an error is returned.

Value

An output image file.

References

Perlich, C., & Swirszcz, G. (2011). On cross-validation and stacking: Building seemingly predictive models on random data. ACM SIGKDD Explorations Newsletter, 12(2), 11-15.

See Also

omics2genelist; omics2pathlist; omics2chrlist

---

plotVarExplained *Plot data summary statistics*

Description

Plot data summary statistics in terms of the proportion of variance explained.

Usage

```
plotVarExplained(data, posF = TRUE, stratify = c("gene", "pathway",
  "chromosome"), core = MulticoreParam(), fileName = NULL)
```

Arguments

data	The input dataset (either data.frame or matrix). Rows are the samples, columns are the probes/genes, except that the first column is the label (the outcome).
posF	A logical value indicating if only positively outcome-associated features should be used. (Default: TRUE)
stratify	A string. The applied stratification method to generate blocklist. Valid options are c('gene', 'pathway', 'chromosome').
core	The number of cores used for computation. (Default: 1)
fileName	The file name specified for the plot. If it is not NULL, then the plot will be generated. The plot will project the data on the first two components. (Default: 'R2explained.png')

Value

An output image file with '.png' format.

References

Yu, Guangchuang, et al. 'clusterProfiler: an R package for comparing biological themes among gene clusters.' Omics: a journal of integrative biology 16.5 (2012): 284-287.

Perlich, C., & Swirszcz, G. (2011). On cross-validation and stacking: Building seemingly predictive models on random data. ACM SIGKDD Explorations Newsletter, 12(2), 11-15.

`predByBS` *Bootstrap resampling prediction via supervised machine learning with feature selection*

Description

Prediction via supervised machine learning using bootstrap resampling along with feature selection methods.

Usage

```
predByBS(trainData, testData, dataMode, repeats, FSmethod, cutP, fdr,
  FScore = MulticoreParam(), classifier, predMode, paramlist,
  innerCore = MulticoreParam())
```

Arguments

`trainData`	The input training dataset. The first column is the label or the output. For binary classes, 0 and 1 are used to indicate the class member.
`testData`	The input test dataset. The first column is the label or the output. For binary classes, 0 and 1 are used to indicate the class member.
`dataMode`	The input training data mode for model training. It is used only if 'testData' is present. It can be a subset of the whole training data or the entire training data. 'subTrain' is the given for subsetting and 'allTrain' for the entire training dataset.
`repeats`	The number of repeats used for boostrapping.
`FSmethod`	Feature selection methods. Available options are c(NULL, 'positive', 'wilcox.test', 'cor.test', 'chisq.test', 'posWilcox', or 'top10pCor').
`cutP`	The cutoff used for p value thresholding. Commonly used cutoffs are c(0.5, 0.1, 0.05, 0.01, etc). The default is 0.05.
`fdr`	Multiple testing correction method. Available options are c(NULL, 'fdr', 'BH', 'holm', etc). See also `p.adjust`. The default is NULL.
`FScore`	The number of cores used for feature selection if parallel computing needed.
`classifier`	Machine learning classifiers.
`predMode`	The prediction mode. Available options are c('probability', 'classification', 'regression').
`paramlist`	A set of model parameters defined in an R list object.
`innerCore`	The number of cores used for computation.

Value

The predicted output for the test data.

Examples

```
## Load data
methylfile <- system.file('extdata', 'methylData.rds', package='BioMM')
methylData <- readRDS(methylfile)
dataY <- methylData[,1]
## select a subset of genome-wide methylation data at random
methylSub <- data.frame(label=dataY, methylData[,c(2:2001)])
trainIndex <- sample(nrow(methylSub), 30)
trainData = methylSub[trainIndex,]
testData = methylSub[-trainIndex,]
library(ranger)
library(BiocParallel)
param1 <- MulticoreParam(workers = 1)
param2 <- MulticoreParam(workers = 20)
predY <- predByBS(trainData, testData,
                  dataMode='allTrain', repeats=50,
                  FSmethod=NULL, cutP=0.1,
                  fdr=NULL, FScore=param1,
                  classifier='randForest',
                  predMode='classification',
                  paramlist=list(ntree=300, nthreads=10),
                  innerCore=param2)
testY <- testData[,1]
accuracy <- classifiACC(dataY=testY, predY=predY)
print(accuracy)
```

`predByCV` *Cross validation prediction by supervised machine learning and feature selection*

Description

Prediction by supervised machine learning models using cross validation along with feature selection methods.

Usage

```
predByCV(data, repeats, nfolds, FSmethod, cutP, fdr,
  FScore = MulticoreParam(), classifier, predMode, paramlist,
  innerCore = MulticoreParam())
```

Arguments

`data`	The input dataset. The first column is the label or the output. For binary classes, 0 and 1 are used to indicate the class member.
`repeats`	The number of repeats used for cross validation. Repeated cross validation is performed if N >=2.
`nfolds`	The number of folds is defined for cross validation.
`FSmethod`	Feature selection methods. Available options are c(NULL, 'positive', 'wilcox.test', 'cor.test', 'chisq.test', 'posWilcox', or 'top10pCor').

cutP	The cutoff used for p value thresholding. Commonly used cutoffs are c(0.5, 0.1, 0.05, 0.01, etc). The default is 0.05.
fdr	Multiple testing correction method. Available options are c(NULL, 'fdr', 'BH', 'holm', etc). See also p.adjust. The default is NULL.
FScore	The number of cores used for feature selection if parallel computing needed.
classifier	Machine learning classifiers.
predMode	The prediction mode. Available options are c('probability', 'classification', 'regression').
paramlist	A set of model parameters defined in an R list object.
innerCore	The number of cores used for computation.

Value

The predicted cross validation output.

Examples

```
## Load data
methylfile <- system.file('extdata', 'methylData.rds', package='BioMM')
methylData <- readRDS(methylfile)
dataY <- methylData[,1]
## select a subset of genome-wide methylation data at random
methylSub <- data.frame(label=dataY, methylData[,c(2:2001)])
library(ranger)
library(BiocParallel)
param1 <- MulticoreParam(workers = 1)
param2 <- MulticoreParam(workers = 20)
predY <- predByCV(methylSub, repeats=1, nfolds=10,
                  FSmethod=NULL, cutP=0.1,
                  fdr=NULL, FScore=param1,
                  classifier='randForest',
                  predMode='classification',
                  paramlist=list(ntree=300, nthreads=1),
                  innerCore=param2)
dataY <- methylData[,1]
accuracy <- classifiACC(dataY=dataY, predY=predY)
print(accuracy)
```

---

predByFS *Prediction by supervised machine learning along with feature selection*

Description

Prediction by supervised machine learning along with feature selection.

Usage

```
predByFS(trainData, testData, FSmethod, cutP, fdr,
  FScore = MulticoreParam(), classifier, predMode, paramlist)
```

Arguments

trainData	The input training dataset. The first column is the label or the output. For binary classes, 0 and 1 are used to indicate the class member.
testData	The input test dataset. The first column is the label or the output. For binary classes, 0 and 1 are used to indicate the class member.
FSmethod	Feature selection methods. Available options are c(NULL, 'positive', 'wilcox.test', 'cor.test', 'chisq.test', 'posWilcox', or 'top10pCor').
cutP	The cutoff used for p value thresholding. Commonly used cutoffs are c(0.5, 0.1, 0.05, etc.). The default is 0.05.
fdr	Multiple testing correction method. Available options are c(NULL, 'fdr', 'BH', 'holm', etc.). See also p.adjust. The default is NULL.
FScore	The number of cores used for feature selection.
classifier	Machine learning classifiers. Available options are c('randForest', 'SVM', 'glmnet').
predMode	The prediction mode. Available options are c('probability', 'classification', 'regression').
paramlist	A set of model parameters defined in an R list object.

Details

If no feature selected or just one selected feature, then top 10

Value

The predicted output for the test data.

Author(s)

Junfang Chen

See Also

getDataAfterFS

Examples

```
## Load data
methylfile <- system.file('extdata', 'methylData.rds', package='BioMM')
methylData <- readRDS(methylfile)
dataY <- methylData[,1]
## select a subset of genome-wide methylation data at random
methylSub <- data.frame(label=dataY, methylData[,c(2:501)])
trainIndex <- sample(nrow(methylSub), 30)
trainData = methylSub[trainIndex,]
testData = methylSub[-trainIndex,]
library(ranger)
library(BiocParallel)
param <- MulticoreParam(workers = 10)
predY <- predByFS(trainData, testData,
                  FSmethod='cor.test', cutP=0.1,
                  fdr=NULL, FScore=param,
```

```
                  classifier='randForest',
                  predMode='classification',
                  paramlist=list(ntree=300, nthreads=20))
testY <- testData[,1]
accuracy <- classifiACC(dataY=testY, predY=predY)
print(accuracy)
```

Index