Package ‘MiPP’

May 30, 2024

Version 1.76.0
Date 2007-01-31
Title Misclassification Penalized Posterior Classification
Author HyungJun Cho <hj4cho@korea.ac.kr>,
    Sukwoo Kim <s4kim@korea.ac.kr>,
    Mat Soukup <soukup@fda.gov>, and
    Jae K. Lee <jaeklee@virginia.edu>
Maintainer Sukwoo Kim <s4kim@korea.ac.kr>
Depends R (>= 2.4)
Imports Biobase, e1071, MASS, stats
Description This package finds optimal sets of genes that separate samples into two or more classes.
License GPL (>= 2)
URL http://www.healthsystem.virginia.edu/internet/hes/biostat/bioinformatics/
biocViews Microarray, Classification
git_url https://git.bioconductor.org/packages/MiPP
git_branch RELEASE_3_19
git_last_commit 02aa5d3
git_last_commit_date 2024-04-30
Repository Bioconductor 3.19
Date/Publication 2024-05-29

Contents

colon .................................................. 2
cv.mipp.rule ........................................ 2
get.mipp ............................................. 3
get.mipp.lda ......................................... 3
get.mipp.logistic .................................... 3
get.mipp.qda ......................................... 3
get.mipp.svm.linear ................................. 3
**colon**  
*Gene expression data for colon cancer*

**Description**
This data set consists of gene expression of colon cancer study.

**Usage**
```r
data(colon)
```

**Format**
A matrix containing 2000 probe sets and 2 classes (T, F)

**Source**

---

**cv.mipp.rule**  
*Fitting cross-validation MiPP*

**Description**
Fits cross-validation MiPP
<table>
<thead>
<tr>
<th>Package</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>get.mipp</td>
<td><em>Choosing a rule</em></td>
</tr>
<tr>
<td></td>
<td>Choose a rule to compute MiPP</td>
</tr>
<tr>
<td>get.mipp.lda</td>
<td><em>Fitting LDA to compute MiPP</em></td>
</tr>
<tr>
<td></td>
<td>Fits LDA to compute MiPP</td>
</tr>
<tr>
<td>get.mipp.logistic</td>
<td><em>Fitting logistic model to compute MiPP</em></td>
</tr>
<tr>
<td></td>
<td>Fits logistic model to compute MiPP</td>
</tr>
<tr>
<td>get.mipp.qda</td>
<td><em>Fitting QDA to compute MiPP</em></td>
</tr>
<tr>
<td></td>
<td>Fits QDA to compute MiPP</td>
</tr>
<tr>
<td>get.mipp.svm.linear</td>
<td><em>Fitting SVM (linear) to compute MiPP</em></td>
</tr>
<tr>
<td></td>
<td>Fits SVM (linear) to compute MiPP</td>
</tr>
</tbody>
</table>
get.mipp.svm.rbf  

*Fitting SVM (RBF) to compute MiPP*

**Description**
Fits SVM (RBF) to compute MiPP

---

1leuk1  

*Gene expression data for leukemia*

**Description**
This data set consists of gene expression of leukemia study.

**Usage**
data(leukemia)

**Format**
A matrix containing 6817 probe sets and 38 samples (2 classes: AML, ALL)

**Source**

---

1leuk2  

*Gene expression data for leukemia*

**Description**
This data set consists of gene expression of leukemia study.

**Usage**
data(leukemia)

**Format**
A matrix containing 6817 probe sets and 34 samples (2 classes: AML, ALL)
leukemia

Source

leukemia Gene expression data for leukemia

Description
This data set consists of gene expression of leukemia study.

Usage
data(leukemia)

Format
A matrix containing 6817 probe sets and 2 classes (AML, ALL)

Source

linearkernel.decision.function
SVM (linear) kernel to compute MiPP

Description
SVM (linear) kernel to compute MiPP
mipp

MiPP-based Classification

Description
Finds optimal sets of genes for classification

Usage
mipp(x, y, x.test = NULL, y.test = NULL, probe.ID = NULL,
rule = "lda", method.cut = "t.test", percent.cut = 0.01,
model.sMiPP.margin = 0.01, min.sMiPP = 0.85, n.drops = 2,
n.fold = 5, p.test = 1/3, n.split = 20,
n.split.eval = 100)

Arguments

x  
data matrix
y  
class vector
x.test  
test data matrix if available
y.test  
test class vector if available
probe.ID  
probe set IDs; if NULL, row numbers are assigned.
rule  
classification rule: "lda", "qda", "logistic", "svmlin", "svmrbf"; the default is "lda".
method.cut  
method for pre-selection; t-test is available.
percent.cut  
proportion of pre-selected genes; the default is 0.01.
model.sMiPP.margin  
smallest set of genes s.t. sMiPP <= (max sMiPP-model.sMiPP.margin); the default is 0.01.
min.sMiPP  
Adding genes stops if max sMiPP is at least min.sMiPP; the default is 0.85.
n.drops  
Adding genes stops if sMiPP decreases (n.drops) times, in addition to min.sMiPP criterion.; the default is 2.
n.fold  
number of folds; default is 5.
p.test  
partition percent of train and test samples when test samples are not available; the default is 1/3 for test set.
n.split  
number of splits; the default is 20.
n.split.eval  
umbr of splits for evalutation; the default is 100.

Value
model  
candidade genes (for each split if no indep set is available
model.eval  
Optimal sets of genes for each split when no indep set is available
Author(s)
Soukup M, Cho H, and Lee JK

References

Examples
#Example 1: When an independent test set is available

data(leukemia)

# Normalize combined data
leukemia <- cbind(leuk1, leuk2)
leukemia <- mipp.preproc(leukemia, data.type="MAS4")

# Train set
x.train <- leukemia[,1:38]
y.train <- factor(c(rep("ALL",27),rep("AML",11)))

# Test set
x.test <- leukemia[,39:72]
y.test <- factor(c(rep("ALL",20),rep("AML",14)))

# Compute MiPP
out <- mipp(x=x.train, y=y.train, x.test=x.test, y.test=y.test, probe.ID = 1:nrow(x.train), n.fold=5, percent.cut=0.05, rule="lda")

# Print candidate models
out$model

#Example 2: When an independent test set is not available

data(colon)

# Normalize data
x <- mipp.preproc(colon)


```
#Deleting contaminated chips
x <- x[-c(51, 55, 45, 49, 56)]
y <- y[-c(51, 55, 45, 49, 56)]

#Compute MiPP
out <- mipp(x=x, y=y, probe.ID = 1:nrow(x), n.fold=5, p.test=1/3, n.split=5, n.split.eval=100, percent.cut= 0.1, rule="lda")

#Print candidate models for each split
out$model

#Print optimal models and independent evaluation for each split
out$model.eval
```

---

**mipp.preproc**

**Preprocessing**

**Description**

Performs IQR normalization, thresholding, and log2-transformation.

**Usage**

```
mipp.preproc(x, data.type = "MAS5")
```

**Arguments**

- `x`: data
- `data.type`: data type is MAS5, MAS4, or dChip

**See Also**

- `mipp`

**Examples**

```
library(MiPP)
data(colon)
colon.nor <- mipp.preproc(colon)
```
mipp.rule

Computing MiPP

Description

 Computes MiPP

mipp.seq

MiPP-based Classification

Description

 sequentially finds optimal sets of genes for classification

Usage

mipp.seq(x, y, x.test = NULL, y.test = NULL, probe.ID = NULL, 
rule = "lda", method.cut = "t.test", percent.cut = 0.01, 
model.sMiPP.margin = 0.01, min.sMiPP = 0.85, n.drops = 2, 
n.fold = 5, p.test = 1/3, n.split = 20, n.split.eval = 100, 
n.seq=3, cutoff.sMiPP=0.7, remove.gene.each.model="all")

Arguments

x          data matrix
y          class vector
x.test     test data matrix if available
y.test     test class vector if available
probe.ID   probe set IDs; if NULL, row numbers are assigned.
rule       classification rule: "lda","qda","logistic","svmlin","svmrbf"; the default is "lda".
method.cut method for pre-selection; t-test is available.
percent.cut proportion of pre-selected genes; the default is 0.01.
model.sMiPP.margin

smallest set of genes s.t. sMiPP <= (max sMiPP-model.sMiPP.margin); the default is 0.01.

min.sMiPP  Adding genes stops if max sMiPP is at least min.sMiPP; the default is 0.85.
n.drops   Adding genes stops if sMiPP decreases (n.drops) times, in addition to min.sMiPP criterion.; the default is 2.
n.fold    number of folds; default is 5.
p.test    partition percent of train and test samples when test samples are not available; the default is 1/3 for test set.
n.split number of splits; the default is 20.
n.split.eval number of splits for evaluation; the default is 100.
n.seq Number of sequential gene model selection; the default is 3.
cutoff.sMiPP Cutoff point of 5 percent sMiPP to select gene models
remove.gene.each.model Re-run after removing all genes in the selected models if “all” and the first gene for each of the selected models if “first”

Value
model candidate genes (for each split if no indep set is available
model.eval Optimal sets of genes for each split when no indep set is available
genes.selected a list of genes selected by sequential selection

Author(s)
Soukup M, Cho H, and Lee JK

References

Examples

##########
#Example 1: When an independent test set is available
data(leukemia)

#Normalize combined data
leukemia <- cbind(leuk1, leuk2)
leukemia <- mipp.preproc(leukemia, data.type="MAS4")

#Train set
x.train <- leukemia[,1:38]
y.train <- factor(c(rep("ALL",27),rep("AML",11)))

#Test set
x.test <- leukemia[,39:72]
y.test <- factor(c(rep("ALL",20),rep("AML",14)))

#Compute MiPP
out <- mipp.seq(x=x.train, y=y.train, x.test=x.test, y.test=y.test, n.fold=5, percent.cut=0.01, rule="lda", n.seq=3)

#Print candidate models
pre.select

out$model

#Print the genes selected
out$genes.selected

##########
#Example 2: When an independent test set is not available

data(colon)

#Normalize data
x <- mipp.preproc(colon)

#Deleting contaminated chips
x <- x[, -c(51, 55, 45, 49, 56)]
y <- y[ -c(51, 55, 45, 49, 56)]

#Compute MiPP
out <- mipp.seq(x = x, y = y, n.fold = 5, p.test = 1/3, n.split = 5, n.split.eval = 100,  
percent.cut = 0.05, rule = "lda", n.seq = 2)

#Print candidate models for each split
out$model

#Print optimal models and independent evaluation for each split
out$model.eval

#Print the genes selected
out$genes.selected

---

**pre.select**  

**Pre-selection**

**Description**

Pre-select genes
<table>
<thead>
<tr>
<th>Package</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>quant.normal</td>
<td>Quantile normalization</td>
</tr>
<tr>
<td>quant.normal2</td>
<td>Quantile normalization</td>
</tr>
<tr>
<td>rbfkernel.decision.function</td>
<td>SVM (RBF) kernel to compute MiPP</td>
</tr>
</tbody>
</table>

**Description**

- **quant.normal**
  - Performs quantile normalization

- **quant.normal2**
  - Performs quantile normalization

- **rbfkernel.decision.function**
  - SVM (RBF) kernel to compute MiPP
Index

* datasets
  colon, 2
  leuk1, 4
  leuk2, 4
  leukemia, 5
* models
  cv.mipp.rule, 2
  get.mipp, 3
  get.mipp.lda, 3
  get.mipp.logistic, 3
  get.mipp.qda, 3
  get.mipp.svm.linear, 3
  get.mipp.svm.rbf, 4
  linearkernel.decision.function, 5
  mipp, 6
  mipp.preproc, 8
  mipp.rule, 9
  mipp.seq, 9
  pre.select, 11
  quant.normal, 12
  quant.normal2, 12
  rbfkernel.decision.function, 12

colon, 2
cv.mipp.rule, 2

get.mipp, 3
get.mipp.lda, 3
get.mipp.logistic, 3
get.mipp.qda, 3
get.mipp.svm.linear, 3
get.mipp.svm.rbf, 4

leuk1, 4
leuk2, 4
leukemia, 5
leukemia(leukemia), 5
linearkernel.decision.function, 5

mipp, 6, 8