

# MiPP

March 24, 2012

---

`colon`                      *Gene expression data for colon cancer*

---

## Description

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

## Usage

```
data(colon)
```

## Format

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

## Source

Alon, U., Barkai, N., Notterman, D.A., Gish, K., Ybarra, S., Mack, D., Levine, A.J. (1999). Broad Patterns of Gene Expression Revealed by Clustering Analysis of Tumor and Normal Colon Tissues probed by Oligonucleotide Arrays, PNAS, 96(12), 6745–6750.

---

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

---

## Description

Fits cross-validation MiPP

---

`get.mipp`                      *Choosing a rule*

---

## Description

Choose a rule to compute MiPP

---

get.mipp.lda      *Fitting LDA to compute MiPP*

---

**Description**

Fits LDA to compute MiPP

---

get.mipp.logistic      *Fitting logistic model to compute MiPP*

---

**Description**

Fits logistic model to compute MiPP

---

get.mipp.qda      *Fitting QDA to compute MiPP*

---

**Description**

Fits QDA to compute MiPP

---

get.mipp.svm.linear  
                                 *Fitting SVM (linear) to compute MiPP*

---

**Description**

Fits SVM (linear) to compute MiPP

---

get.mipp.svm.rbf      *Fitting SVM (RBF) to compute MiPP*

---

**Description**

Fits SVM (RBF) to compute MiPP

---

leuk1

*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**

Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, P., Coller, H., Loh, M.L., Downing, J.R., Caliguri, M.A., Bloomfield, C.D., and Lander, E.S. (1999) Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286, 531-537.

---

leuk2

*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)

**Source**

Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, P., Coller, H., Loh, M.L., Downing, J.R., Caliguri, M.A., Bloomfield, C.D., and Lander, E.S. (1999) Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286, 531-537.

---

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**

Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, P., Coller, H., Loh, M.L., Downing, J.R., Caliguri, M.A., Bloomfield, C.D., and Lander, E.S. (1999) Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286, 531-537.

---

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**

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

**Value**

<code>model</code>	candidate genes (for each split if no indep set is available)
<code>model.eval</code>	Optimal sets of genes for each split when no indep set is available

**Author(s)**

Soukup M, Cho H, and Lee JK

**References**

Soukup M, Cho H, and Lee JK (2005). Robust classification modeling on microarray data using misclassification penalized posterior, *Bioinformatics*, 21 (Suppl): i423-i430.

Soukup M and Lee JK (2004). Developing optimal prediction models for cancer classification using gene expression data, *Journal of Bioinformatics and Computational Biology*, 1(4) 681-694

**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))

#Print candidate models
out$model

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

data(colon)

#Normalize data
x <- mipp.preproc(colon)
y <- factor(c("T", "N", "T", "N", "T", "N", "T", "N", "T", "N",
              "T", "N", "T", "N", "T", "N", "T", "N", "T", "N",
              "T", "N", "T", "N", "T", "T", "T", "T", "T", "T",
              "T", "T", "T", "T", "T", "T", "T", "T", "N", "T",
              "T", "N", "N", "T", "T", "T", "T", "N", "T", "N",
              "N", "T", "T", "N", "N", "T", "T", "T", "T", "N",
              "T", "N"))

#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=
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, thesholding, 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**

<code>x</code>	data matrix
<code>y</code>	class vector
<code>x.test</code>	test data matrix if available
<code>y.test</code>	test class vector if available
<code>probe.ID</code>	probe set IDs; if NULL, row numbers are assigned.
<code>rule</code>	classification rule: "lda","qda","logistic","svmlin","svmrbf"; the default is "lda".
<code>method.cut</code>	method for pre-selection; t-test is available.
<code>percent.cut</code>	proportion of pre-selected genes; the default is 0.01.
<code>model.sMiPP.margin</code>	smallest set of genes s.t. $sMiPP \leq (\max sMiPP - \text{model.sMiPP.margin})$ ; the default is 0.01.
<code>min.sMiPP</code>	Adding genes stops if max sMiPP is at least min.sMiPP; the default is 0.85.
<code>n.drops</code>	Adding genes stops if sMiPP decreases (n.drops) times, in addition to min.sMiPP criterion.; the default is 2.
<code>n.fold</code>	number of folds; default is 5.
<code>p.test</code>	partition percent of train and test samples when test samples are not available; the default is 1/3 for test set.
<code>n.split</code>	number of splits; the default is 20.
<code>n.split.eval</code>	numbr of splits for evaluation; the default is 100.
<code>n.seq</code>	Number of sequential gene model selection; the default is 3.
<code>cutoff.sMiPP</code>	Cutoff point of 5 percent sMiPP to select gene models
<code>remove.gene.each.model</code>	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**

<code>model</code>	candiadate genes (for each split if no indep set is available)
<code>model.eval</code>	Optimal sets of genes for each split when no indep set is available
<code>genes.selected</code>	a list of genes selected by sequential selection

**Author(s)**

Soukup M, Cho H, and Lee JK

**References**

- Soukup M, Cho H, and Lee JK (2005). Robust classification modeling on microarray data using misclassification penalized posterior, *Bioinformatics*, 21 (Suppl): i423-i430.
- Soukup M and Lee JK (2004). Developing optimal prediction models for cancer classification using gene expression data, *Journal of Bioinformatics and Computational Biology*, 1(4) 681-694



**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=0.05)

#Print candidate models
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)
y <- factor(c("T", "N", "T", "N", "T", "N", "T", "N", "T", "N", "T", "N",
             "T", "N", "T", "N", "T", "N", "T", "N", "T", "N", "T", "N",
             "T", "T", "T", "T", "T", "T", "T", "T", "N", "T",
             "T", "N", "N", "T", "T", "T", "T", "N", "T", "N",
             "N", "T", "T", "N", "N", "T", "T", "T", "T", "N",
             "T", "N"))

#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

---

```
quant.normal       Quantile normalization
```

---

**Description**

Performs quantile normalization

---

```
quant.normal2     Quantile normalization
```

---

**Description**

Performs quantile normalization

---

```
rbfkernel.decision.function
                    SVM (RBF) kernel to compute MiPP
```

---

**Description**

SVM (RBF) kernel to compute MiPP

# Index

## \*Topic **datasets**

- colon, 1
- leuk1, 3
- leuk2, 3
- leukemia, 4

## \*Topic **models**

- cv.mipp.rule, 1
- get.mipp, 1
- get.mipp.lda, 2
- get.mipp.logistic, 2
- get.mipp.qda, 2
- get.mipp.svm.linear, 2
- get.mipp.svm.rbf, 2
- linearkernel.decision.function,  
4
- mipp, 4
- mipp.preproc, 6
- mipp.rule, 7
- mipp.seq, 7
- pre.select, 10
- quant.normal, 10
- quant.normal2, 10
- rbfkernel.decision.function,  
10

colon, 1

cv.mipp.rule, 1

get.mipp, 1

get.mipp.lda, 2

get.mipp.logistic, 2

get.mipp.qda, 2

get.mipp.svm.linear, 2

get.mipp.svm.rbf, 2

leuk1, 3

leuk2, 3

leukemia, 4

leukimia (*leukemia*), 4

linearkernel.decision.function,  
4

mipp, 4, 7

mipp.preproc, 6

mipp.rule, 7

mipp.seq, 7

pre.select, 10

quant.normal, 10

quant.normal2, 10

rbfkernel.decision.function, 10