Package 'classyfire'

July 15, 2014

Type Package

Title Optimised multivariate classification models

Version 0.1-0

Date 2014-07-11

Author Eleni Chatzimichali <hatzimihali.eleni@gmail.com>, Conrad Bessant <c.bessant@qmul.ac.uk>

Maintainer Eleni Chatzimichali <hatzimihali.eleni@gmail.com>

Description The classyfire package provides optimised multivariate classification analyses by incorporating parallel programming in order to significantly minimise the computational complexity and execution times as well as improve the overall performance of the classifiers and classification ensembles. The package introduces a new heuristic methodology for speeding up the optimisation process of the SVM hyperparameters via bootstrapping, chiefly focusing on the cases of SVMs with the RBF kernel. In this novel approach, a fast and robust approximation algorithm for constrained nonlinear optimisation, the Box complex algorithm, was used to replace the widely applied yet computationally intensive grid-search.

License GPL (>= 2)

Depends e1071 (>= 1.6-3), boot, neldermead (>= 1.0-9), snowfall (>= 1.84-6), ggplot2

R topics documented:

`avgTestAcc` *Average test accuracy for a classification ensemble*

Description

The avgTestAcc function returns the average test accuracy (%CC) of an ensemble of classifiers as generated by the function `boxEnsRBF`.

Usage

```
avgTestAcc(ens)
```

Arguments

`ens`	The R object (classification ensemble) as generated by `boxEnsRBF`

Value

Returns a numerical value, equal to the average overall test accuracy (%CC) of the ensemble.

Author(s)

Eleni Chatzimichali <hatzimihali.eleni@gmail.com>, Conrad Bessant <c.bessant@qmul.ac.uk>

References

http://www.classyfire.org

See Also

`getTestAcc`, `confMatr`

Examples

```
data(iris)

irisClass = iris[,5]
irisData = iris[,-5]

ens <- boxEnsRBF(irisData, irisClass, bootNum = 10, ensNum = 20, parallel = TRUE,
                 cpus = 4, type = "SOCK")

# Return the average test accuracy for the ensemble
avgTestAcc(ens)
```

avgTrainAcc *Average train accuracy for a classification ensemble*

Description

The avgTrainAcc function returns the average train accuracy of an ensemble of classifiers as generated by the function boxEnsRBF.

Usage

```
avgTrainAcc(ens)
```

Arguments

ens	The R object as generated by boxEnsRBF

Value

Returns a numeric value, equal to the average overall train accuracy of the ensemble.

Author(s)

Eleni Chatzimichali <hatzimihali.eleni@gmail.com>, Conrad Bessant <c.bessant@qmul.ac.uk>

References

http://www.classyfire.org

See Also

getTrainAcc, avgTrainAcc, confMatr

Examples

```
data(iris)

irisClass = iris[,5]
irisData = iris[,-5]

ens <- boxEnsRBF(irisData, irisClass, bootNum = 10, ensNum = 20, parallel = TRUE,
                 cpus = 4, type = "SOCK")

# Return the average train accuracy for the ensemble
avgTrainAcc(ens)
```

boxEnsRBF *Optimisation of the radial (RBF) SVM tuning process via bootstrapping*

Description

The boxEnsRBF function introduces a new heuristic methodology for speeding up the optimisation process of the RBF SVM hyperparameters via bootstrapping. In this novel approach, a fast and robust approximation algorithm for constrained nonlinear optimisation, the Box complex algorithm, is used to replace the widely applied yet computationally intensive grid-search. The Box complex algorithm in addition to parallel programming is used as a means of significantly minimising the computational complexity and overall execution time of the analysis as well as improving the overall performance of the classifiers.

Usage

```
boxEnsRBF(inputData, inputClass, ...)

## Default S3 method:
boxEnsRBF(inputData, inputClass, bootNum = 100, ensNum = 100,
     parallel=TRUE, cpus = NULL, type = "SOCK", socketHosts = NULL)
```

Arguments

inputData	The input data matrix as provided by the user (mandatory field).
inputClass	The input class vector as provided by the user (mandatory field).
bootNum	The number of bootstrap iterations in the optimisation process. By default, the value is set to 100.
ensNum	The number of classifiers that form the classification ensemble. By default, the value is set to 100.
parallel	Boolean value that determines parallel or sequential execution.
cpus	Numeric value that provides the number of CPUs requested for the cluster.
type	The type of cluster. It can take the values 'SOCK', 'MPI', 'PVM' or 'NWS'. By default, type is equal to 'SOCK'
socketHosts	Host list for socket clusters. Only needed for socketmode (SOCK) and if using more than one machines (if using only your local machine (localhost) no list is needed).
...	The remaining optional fields.

Details

For a given input dataset D, a random fraction of samples is removed and kept aside as an independent test set during the training process of the model. This selection of samples forms the dataset Dtest. This test set typically comprises a third of the original samples. Using a stratified holdout approach, the test set consists of the same balance of sample classes as the initial dataset D. The remaining samples that are not selected, form the training set Dtrain. Since the test set is kept aside during the whole training process, the risk of overfitting is minimised.
In the case of bootstrapping, a bootstrap training set DbootTrain is created by randomly picking n samples with replacement from the training dataset Dtrain. The total size of DbootTrain is equal to

the size of Dtrain. Since bootstrapping is based on sampling with replacement, any given sample could be present multiple times within the same bootstrap training set. The remaining samples not found in the bootstrap training set make up the bootstrap test set DbootTest. To avoid reliance on one specific bootstrapping split, bootstrapping is repeated at least 100 times until a clear winning parameter combination emerges.
Ultimately, the optimal parameters are used to train a new classifier with the full Dtrain dataset and test it on the independent test set Dtest, which has been left aside during the entire optimisation process. Even though the approach described thus far generates an excellent classifier, the random selection of test samples in the initial split may have been fortunate. For a more accurate and reliable overview, the whole process is repeated a minimum of 100 times until a stable average classification rate emerges. The output of this repetition consists of at least 100 individual classification models built using the optimum parameter settings. Rather than isolating a single classifier, all individual classification models are fused into a classification ensemble.

Value

The boxEnsRBF function returns an object in the form of an R list. The attributes of the list can be accessed by executing the attributes command. More specifically, the list of attributes includes:

`testAcc`	The test accuracy (%CC) (numerical value) of a single classifier in the ensemble.
`trainAcc`	The train accuracy (numerical value) of a single classifier in the ensemble.
`optGamma`	Optimal gamma value
`optCost`	Optimal cost value for the RBF SVM
`runTime`	The execution time for a single classifier within the ensemble.
`confMatr`	The confusion matrix of a single classifier.
`propTable`	Similar to the confusion matrix, the per class (%) accuracies.
`predClass`	The vector of classes for the test class as predicted by the SVM model.
`testClass`	The vector of true classes of the test class.
`missNames`	In case that the names of the samples (rows) are supplied in the input data matrix, the missNames attribute returns the names of the missclassified samples.
`accNames`	In case that the names of the samples (rows) are supplied in the input data matrix, the accNames attribute returns the names of the correctly classified samples.
`trainClass`	The vector of classes used in the training of the SVM.
`testData`	The data matrix (individual test set) used for testing.
`trainData`	The data matrix (individual test set) used for training.
`testSamples`	The randomly selected samples (rows) that were used in this instance to create the
`bootObj`	The matrix of bootstrapped samples based on the provided bootNum.
`svmModel`	The created SVM model as an R object.

Note

The boxEnsRBF function does not force an upper limit for the bootNum, ensNum and cpus parameters to the users. However, it is advisable not to use extremely high values for these parameters.

Author(s)

Eleni Chatzimichali <hatzimihali.eleni@gmail.com>, Conrad Bessant <c.bessant@qmul.ac.uk>

References

http://www.classyfire.org

See Also

runPerm

Examples

```
data(iris)

irisClass = iris[,5]
irisData = iris[,-5]

# Construct a classification ensemble with 20 classifiers and 10 bootstrap iterations during optimisation

ens <- boxEnsRBF(irisData, irisClass, bootNum = 10, ensNum = 20, parallel = TRUE,
                 cpus = 4, type = "SOCK")

# Return the size of the classification ensemble
length(ens)

# Return the list of attributes available for the ensemble
attributes(ens[[1]])

# Return the test accuracy of one individual classifier in the ensemble

round(ens[[1]]$testAcc, digits=2)

# Return the train accuracy of one individual classifier in the ensemble
round(ens[[1]]$trainAcc, digits=2)

# Return all the test accuracies and the average test accuracy in the ensemble
getTestAcc(ens)
avgTestAcc(ens)

# Return all the train accuracies and the average train accuracy in the ensemble
getTrainAcc(ens)
avgTestAcc(ens)
```

---

confMatr *Confusion matrix*

Description

Returns the confusion matrix for a given iteration - a given classifier within the ensemble.

Usage

```
confMatr(iter, ens)
```

Arguments

iter	The number of the given iteration
ens	The R object as generated by boxEnsRBF

Author(s)

Eleni Chatzimichali <hatzimihali.eleni@gmail.com>, Conrad Bessant <c.bessant@qmul.ac.uk>

References

http://www.classyfire.org

See Also

overallClassPred

Examples

```
data(iris)

irisClass = iris[,5]
irisData = iris[,-5]

ens <- boxEnsRBF(irisData, irisClass, bootNum = 10, ensNum = 20, parallel = TRUE,
                 cpus = 4, type = "SOCK")

confMatr(1, ens)
```

---

fiveNum *Five number summary*

Description

The fiveNum returns the "five number summary", a descriptive statistic that consists of the minimum, first (lower) quartile, median, third (upper) quartile and maximum value of a given distribution.

Usage

```
fiveNum(permObj)
```

Arguments

permObj	The permutation object as generated by runPerm

Value

Returns an R object in the form of a list that contains the five number summary. The names of the returned statistics can be viewed by using the function "attributes".

Author(s)

Eleni Chatzimichali <hatzimihali.eleni@gmail.com>, Conrad Bessant <c.bessant@qmul.ac.uk>

References

http://www.classyfire.org

See Also

fivenum

Examples

```
data(iris)

irisClass = iris[,5]
irisData = iris[,-5]

permObj <- runPerm(irisData, irisClass, bootNum = 10, ensNum = 20, permNum = 5, parallel = TRUE,
                   cpus = 4, type = "SOCK")

fiveNum(permObj)
fiveNum(permObj)$median
fiveNum(permObj)$minimum
fiveNum(permObj)$maximum
fiveNum(permObj)$upperQ
fiveNum(permObj)$lowerQ
```

---

getTestAcc *Test Accuracies (%CC) in a classification ensemble*

Description

The testAcc function returns a vector of the test accuracies for all the classifiers within the ensemble as generated by boxEnsRBF.

Usage

```
getTestAcc(ens)
```

Arguments

ens	The R object as generated by boxEnsRBF

Author(s)

Eleni Chatzimichali <hatzimihali.eleni@gmail.com>, Conrad Bessant <c.bessant@qmul.ac.uk>

References

http://www.classyfire.org

See Also

avgTestAcc, confMatr

Examples

```
data(iris)

irisClass = iris[,5]
irisData = iris[,-5]

ens <- boxEnsRBF(irisData, irisClass, bootNum = 10, ensNum = 20, parallel = TRUE,
                 cpus = 4, type = "SOCK")

# Return the test accuracies for every classifier in the ensemble

getTestAcc(ens)

# Return the train accuracies for every classifier in the ensemble

getTrainAcc(ens)
```

---

getTrainAcc *Train Accuracies in a classification ensemble*

Description

The testAcc function returns a vector of the test accuracies for all the classifiers within the ensemble as generated by boxEnsRBF.

Usage

```
getTrainAcc(ens)
```

Arguments

ens	The R object as generated by boxEnsRBF

Author(s)

Eleni Chatzimichali <hatzimihali.eleni@gmail.com>, Conrad Bessant <c.bessant@qmul.ac.uk>

References

http://www.classyfire.org

See Also

avgTrainAcc, confMatr

Examples

```
data(iris)

irisClass = iris[,5]
irisData = iris[,-5]

ens <- boxEnsRBF(irisData, irisClass, bootNum = 10, ensNum = 20, parallel = TRUE,
```

```
                cpus = 4, type = "SOCK")

# Return the test accuracies for every classifier in the ensemble

getTestAcc(ens)

# Return the train accuracies for every classifier in the ensemble

getTrainAcc(ens)
```

---

ggbarTest *Scatter Plot of Test Accuracies*

Description

The ggbarTest function generates a scatter plot of all the test accuracies within an ensemble, as constructed by the boxEnsRBF function.

Usage

```
ggbarTest(ensObj, xlabel = NULL, ylabel = NULL, showText = FALSE, xlims = NULL, ylims = NULL, ...)
```

Arguments

ensObj	The R object as generated by boxEnsRBF
xlabel	A sub title for the x axis (optional field).
ylabel	A sub title for the y axis (optional field).
showText	Boolean value, by default set to FALSE. If showText=TRUE, then the values of all test accuracies in the ensemble are displayed in the plot.
xlims	A vector of numeric values that specifies the minimum and maximum values in the x axis.
ylims	A vector of numeric values that specifies the minimum and maximum values in the y axis.
...	Further arguments and graphical parameters passed to the ggplot.

Author(s)

Eleni Chatzimichali <hatzimihali.eleni@gmail.com>, Conrad Bessant <c.bessant@qmul.ac.uk>

References

http://www.classyfire.org

See Also

ggplotTrend

Examples

```
data(iris)

irisClass = iris[,5]
irisData = iris[,-5]

ens <- boxEnsRBF(irisData, irisClass, bootNum = 10, ensNum = 20, parallel = TRUE,
                 cpus = 4, type = "SOCK")

ggbarTest(ens)
ggbarTest(ens, showText = TRUE)
ggbarTest(ens, showText = TRUE, ylims=c(80, 100))
```

---

ggClassBar *Per class accuracies of correctly classified samples*

Description

The ggClassBar function generates a barplot of the per class correctly classified samples.

Usage

```
ggClassBar(ensObj, showText = FALSE, ...)
```

Arguments

ensObj	The R object as generated by boxEnsRBF
showText	Boolean value, by default set to FALSE. If showText=TRUE, then the per class accuracies (%) for all classifiers in the ensemble are displayed in the plot.
...	Further arguments and graphical parameters passed to the ggplot.

Author(s)

Eleni Chatzimichali <hatzimihali.eleni@gmail.com>, Conrad Bessant <c.bessant@qmul.ac.uk>

References

http://www.classyfire.org

See Also

ggClassPred

Examples

```
data(iris)

irisClass = iris[,5]
irisData = iris[,-5]

ens <- boxEnsRBF(irisData, irisClass, bootNum = 10, ensNum = 20, parallel = TRUE,
                 cpus = 4, type = "SOCK")
```

```
ggClassBar(ens)
ggClassBar(ens, showText = TRUE)
```

---

ggClassPred *Barplot of the per class accuracies.*

Description

The ggClasPred function generates a barplot with the per class accuracies (%) for all the correctly classified and misclassified samples in the classification ensemble.

Usage

```
ggClassPred(ensObj, position = "stack", showText = FALSE, xlabel = NULL, ylabel = NULL, ...)
```

Arguments

ensObj	The R object as generated by boxEnsRBF
position	The position may be equal to either "stack" or "dodge".
showText	Boolean value, by default set to FALSE. If showText=TRUE, then the per class accuracies (%) for all classifiers in the ensemble are displayed in the plot.
xlabel	A sub title for the x axis (optional field).
ylabel	A sub title for the y axis (optional field).
...	Further arguments and graphical parameters passed to the ggplot.

Author(s)

Eleni Chatzimichali <hatzimihali.eleni@gmail.com>, Conrad Bessant <c.bessant@qmul.ac.uk>

References

http://www.classyfire.org

See Also

ggClassBar

Examples

```
data(iris)

irisClass = iris[,5]
irisData = iris[,-5]

ens <- boxEnsRBF(irisData, irisClass, bootNum = 10, ensNum = 20, parallel = TRUE,
                 cpus = 4, type = "SOCK")

ggClassPred(ens, position = "dodge", showText = TRUE)
ggClassPred(ens, position = "stack")
```

ggEnsHist *Ensemble Histograms*

Description

The ggEnsHist function generates a histogram of the ensemble results as generated by boxEnsRBF.

Usage

```
ggEnsHist(ensObj, density = FALSE, percentiles = FALSE, mean = FALSE, median = FALSE)
```

Arguments

ensObj	The ensemble object as generated by boxEnsRBF
density	Boolean value, by default equal to FALSE. If density = FALSE, the histogram depicts frequencies, the counts component of the result. Instead, for density = TRUE, probability densities are plotted.
percentiles	Boolean value, by default equal to FALSE. If oercentiles = TRUE, the upper and lower percentiles of the distribution are depicted in the plot.
mean	Boolean value, by default equal to FALSE. If mean = TRUE, the mean of the distribution is depicted in the plot.
median	Boolean value, by default equal to FALSE. If median = TRUE, the median of the distribution is depicted in the plot.

Author(s)

Eleni Chatzimichali <hatzimihali.eleni@gmail.com>, Conrad Bessant <c.bessant@qmul.ac.uk>

References

http://www.classyfire.org

See Also

ggEnsHist

Examples

```
data(iris)

irisClass = iris[,5]
irisData = iris[,-5]

ens <- boxEnsRBF(irisData, irisClass, bootNum = 10, ensNum = 20, parallel = TRUE,
                 cpus = 4, type = "SOCK")

ggEnsHist(ens)
ggEnsHist(ens, density = TRUE)
ggEnsHist(ens, density = TRUE, percentiles=TRUE)
ggEnsHist(ens, density = TRUE, percentiles=TRUE, mean=TRUE)
ggEnsHist(ens, density = TRUE, percentiles=TRUE, median=TRUE)
```

ggPermHist *Permutation Histograms*

Description

The ggPermHist function generates a histogram of the permutation results as generated by runPerm.

Usage

```
ggPermHist(permObj, density = FALSE, percentiles = FALSE, mean = FALSE, median = FALSE, ...)
```

Arguments

permObj	The permutation object as generated by runPerm
density	Boolean value, by default equal to FALSE. If density = FALSE, the histogram depicts frequencies, the counts component of the result. Instead, for density = TRUE, probability densities are plotted.
percentiles	Boolean value, by default equal to FALSE. If oercentiles = TRUE, the upper and lower percentiles of the distribution are depicted in the plot.
mean	Boolean value, by default equal to FALSE. If mean = TRUE, the mean of the distribution is depicted in the plot.
median	Boolean value, by default equal to FALSE. If median = TRUE, the median of the distribution is depicted in the plot.
...	Further arguments and graphical parameters passed to the ggplot.

Author(s)

Eleni Chatzimichali <hatzimihali.eleni@gmail.com>, Conrad Bessant <c.bessant@qmul.ac.uk>

References

http://www.classyfire.org

See Also

ggEnsHist

Examples

```
data(iris)

irisClass = iris[,5]
irisData = iris[,-5]

permObj <- runPerm(irisData, irisClass, bootNum = 10, ensNum = 20, permNum = 5, parallel = TRUE,
                   cpus = 4, type = "SOCK")

ggPermHist(permObj)
ggPermHist(permObj, density=TRUE)
ggPermHist(permObj, density=TRUE, percentiles = TRUE, mean = TRUE)
ggPermHist(permObj, density=TRUE, percentiles = TRUE, median = TRUE)
```

ggplotTrend *Trend of the test accuracies*

Description

The ggplotTrend function displays the average test accuracies for every new classifier added to the ensemble, as constructed by the boxEnsRBF function.

Usage

```
ggplotTrend(ensObj, xlabel = NULL, ylabel = NULL, showText = FALSE, xlims = NULL, ylims = NULL, ...
```

Arguments

ensObj	The R object as generated by boxEnsRBF
xlabel	A sub title for the x axis (optional field).
ylabel	A sub title for the y axis (optional field).
showText	Boolean value, by default set to FALSE. If showText=TRUE, then the values of all test accuracies in the ensemble are displayed in the plot.
xlims	A vector of numeric values that specifies the minimum and maximum values in the x axis.
ylims	A vector of numeric values that specifies the minimum and maximum values in the y axis.
...	Further arguments and graphical parameters passed to the ggplot.

Author(s)

Eleni Chatzimichali <hatzimihali.eleni@gmail.com>, Conrad Bessant <c.bessant@qmul.ac.uk>

References

http://www.classyfire.org

See Also

ggbarTest

Examples

```
data(iris)

irisClass = iris[,5]
irisData = iris[,-5]

ens <- boxEnsRBF(irisData, irisClass, bootNum = 10, ensNum = 20, parallel = TRUE,
                 cpus = 4, type = "SOCK")

ggplotTrend(ens)
ggplotTrend(ens, showText  = TRUE)
ggplotTrend(ens, showText  = TRUE, ylims=c(90, 100))
```

optimParams *Optimal Paramaters*

Description

The function returns the optimal hyperparameters for the constructed classifier or classification ensemble.

Usage

```
optimParams(ens)
```

Arguments

ens	The R object as generated by boxEnsRBF

Value

Returns a vector or matrix with the optimal hyperparameters as found by the optimisation process.

Author(s)

Eleni Chatzimichali <hatzimihali.eleni@gmail.com>, Conrad Bessant <c.bessant@qmul.ac.uk>

References

http://www.classyfire.org

Examples

```
data(iris)

irisClass = iris[,5]
irisData = iris[,-5]

ens <- boxEnsRBF(irisData, irisClass, bootNum = 10, ensNum = 20, parallel = TRUE,
                 cpus = 4, type = "SOCK")

optimParam <- optimParams(ens)
optimParam
```

overallClassPred *Overal Class Predictions*

Description

The function overallClassPred returns the percentages of correctly classified samples as well as misclassified samples per given class, displayed in the form of a confusion matrix.

Usage

```
overallClassPred(ens)
```

Arguments

ens	The R object as generated by boxEnsRBF

Author(s)

Eleni Chatzimichali <hatzimihali.eleni@gmail.com>, Conrad Bessant <c.bessant@qmul.ac.uk>

References

http://www.classyfire.org

See Also

confMatr

Examples

```
data(iris)

irisClass = iris[,5]
irisData = iris[,-5]

ens <- boxEnsRBF(irisData, irisClass, bootNum = 10, ensNum = 20, parallel = TRUE,
                 cpus = 4, type = "SOCK")

overallClassPred(ens)
```

---

runPerm *Permutation tests*

Description

The function runPermutation performs permutation tests for an ensemble of RBF SVM models validated with bootstrapping

Usage

```
runPerm(inputData, inputClass, bootNum = 100, ensNum = 100, permNum = 100, parallel = TRUE, cpus =
```

Arguments

inputData	The input data matrix as provided by the user (mandatory field).
inputClass	The input class vector as provided by the user (mandatory field).
bootNum	The number of bootstrap iterations during the optimisation process. By default, the value is set to 100.
ensNum	The number of classifiers that constitute the ensemble for each permutation. By default, the value is set to 100.
permNum	The number of permutations to be executed. By default, the value is set to 100.
parallel	Boolean value that determines parallel or sequential execution.
cpus	Numeric value that provides the number of CPUs requested for the cluster.

type	The type of cluster. It can take the values 'SOCK', 'MPI', 'PVM' or 'NWS'. By default, type is equal to 'SOCK'
socketHosts	Host list for socket clusters. Only needed for socketmode (SOCK) and if using more than one machines (if using only your local machine (localhost) no list is needed).
progressBar	Boolean value that determines whether a progress bar should be displayed.

Details

Permutation testing is a widely-applied process used in order to provide an indication of the statistical significance of the classification results. In a permutation test, the entries of the original class vector (inputClass) are randomly shuffled, while the class distribution is preserved. This approach destroys all the sample membership information since the samples of a permuted dataset correspond to randomly assigned classes. The whole model building process as described boxEnsRBF is once more repeated for the "false" (permuted) classes. In general, permutation testing is performed at least 100 times until a stable distribution of results is obtained.

Author(s)

Eleni Chatzimichali <hatzimihali.eleni@gmail.com>, Conrad Bessant <c.bessant@qmul.ac.uk>

Examples

```
data(iris)

irisClass = iris[,5]
irisData = iris[,-5]

ens <- boxEnsRBF(irisData, irisClass, bootNum = 10, ensNum = 20, parallel = TRUE,
                 cpus = 4, type = "SOCK")

# Return the list of attributes available for the ensemble
attributes(ens[[1]])

# Return the test accuracy of one individual classifier in the ensemble

round(ens[[1]]$testAcc, digits=2)


# Execute 5 permutation rounds; in each permutation test, an ensemble of 20 classifiers is constructed,
# each running 10 bootstrap iterations during the optimization process

permObj <- runPerm(irisData, irisClass, bootNum = 10, ensNum = 20, permNum = 5, parallel = TRUE,
                   cpus = 4, type = "SOCK")

length(permObj)
attributes(permObj)

permObj$avgTestAcc
permObj$execTime
length(permObj$permList)
```

Index