Package ‘RaSEn’

October 15, 2021

Type Package

Title Random Subspace Ensemble Classification and Variable Screening

Version 3.0.0

Author Ye Tian [aut, cre] and Yang Feng [aut]

Maintainer Ye Tian <ye.t@columbia.edu>

Description We propose a general ensemble classification framework, RaSE algorithm, for the sparse classification problem. In RaSE algorithm, for each weak learner, some random subspaces are generated and the optimal one is chosen to train the model on the basis of some criterion. To be adapted to the problem, a novel criterion, ratio information criterion (RIC) is put up with based on Kullback-Leibler divergence. Besides minimizing RIC, multiple criteria can be applied, for instance, minimizing extended Bayesian information criterion (eBIC), minimizing training error, minimizing the validation error, minimizing the cross-validation error, minimizing leave-one-out error. There are various choices of base classifier, for instance, linear discriminant analysis, quadratic discriminant analysis, k-nearest neighbour, logistic regression, decision trees, random forest, support vector machines. RaSE algorithm can also be applied to do feature ranking, providing us the importance of each feature based on the selected percentage in multiple subspaces. RaSE framework can be extended to the general prediction framework, including both classification and regression. We can use the selected percentages of variables for variable screening. The latest version added the variable screening function for both regression and classification problems.

Imports MASS, caret, class, doParallel, e1071, foreach, nnet, randomForest, rpart, stats, ggplot2, gridExtra, formatR, FNN, ranger, KernelKnn, utils, ModelMetrics, glmnet

License GPL-2

Encoding UTF-8

LazyData TRUE

LazyDataCompression bzip2

RoxygenNote 7.1.2

Suggests knitr, rmarkdown

VignetteBuilder knitr

Depends R (>= 3.1.0)

NeedsCompilation no

R topics documented:

colon . . . 2
predict.RaSE . . . 3
predict.super_RaSE . . . 4
print.RaSE . . . 5
print.super_RaSE . . . 6
RaModel . . . 7
RaPlot . . . 8
RaRank . . . 9
RaScreen . . . 10
Rase . . . 14
rat . . . 20

Index **21**

`colon` *Colon data set.*

Description

Alon et al.'s Colon cancer dataset containing information on 62 samples for 2000 genes. The samples belong to tumor and normal colon tissues.

Usage

```
colon
```

Format

A list with the predictor matrix `x` and binary 0/1 response vector `y`.

Source

The link to this data set: `http://genomics-pubs.princeton.edu/oncology/`

References

Alon, U., Barkai, N., Notterman, D.A., Gish, K., Ybarra, S., Mack, D. and Levine, A.J., 1999. *Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. Proceedings of the National Academy of Sciences, 96(12), pp.6745-6750.*

Tian, Y. and Feng, Y., 2021. *RaSE: A Variable Screening Framework via Random Subspace Ensembles. arXiv preprint arXiv:2102.03892.*

`predict.RaSE` *Predict the outcome of new observations based on the estimated RaSE classifier (Tian, Y. and Feng, Y., 2021).*

Description

Predict the outcome of new observations based on the estimated RaSE classifier (Tian, Y. and Feng, Y., 2021).

Usage

```
## S3 method for class 'RaSE'
predict(object, newx, type = c("vote", "prob", "raw-vote", "raw-prob"), ...)
```

Arguments

`object` fitted `'RaSE'` object using `Rase`.

`newx` a set of new observations. Each row of `newx` is a new observation.

`type` the type of prediction output. Can be 'vote', 'prob', 'raw-vote' or 'raw-prob'. Default = 'vote'.

- vote: output the predicted class (by voting and cut-off) of new observations. Avalilable for all base learner types.
- prob: output the predicted probabilities (posterior probability of each observation to be class 1) of new observations. It is the average probability over all base learners. Avalilable only when base leaner is not equal to 'svm' and 'tree'.
- raw-vote: output the predicted class of new observations for all base learners. It is a `n` by `B1` matrix. `n` is the test sample size and `B1` is the number of base learners used in RaSE. Avalilable for all base learner types.
- raw-prob: output the predicted probabilities (posterior probability of each observation to be class 1) of new observations for all base learners. It is a `n` by `B1` matrix. Avalilable only when base leaner is not equal to 'svm' and 'tree'.

`...` additional arguments.

Value

depends on the parameter `type`. See the list above.

References

Tian, Y. and Feng, Y., 2021. RaSE: Random subspace ensemble classification. Journal of Machine Learning Research, 22(45), pp.1-93.

See Also

`Rase`.

Examples

```
## Not run:
set.seed(0, kind = "L'Ecuyer-CMRG")
train.data <- RaModel("classification", 1, n = 100, p = 50)
test.data <- RaModel("classification", 1, n = 100, p = 50)
xtrain <- train.data$x
ytrain <- train.data$y
xtest <- test.data$x
ytest <- test.data$y

model.fit <- Rase(xtrain, ytrain, B1 = 100, B2 = 100, iteration = 0, base = 'lda',
cores = 2, criterion = 'ric', ranking = TRUE)
ypred <- predict(model.fit, xtest)
mean(ypred != ytest)

## End(Not run)
```

`predict.super_RaSE` *Predict the outcome of new observations based on the estimated super RaSE classifier (Zhu, J. and Feng, Y., 2021).*

Description

Predict the outcome of new observations based on the estimated super RaSE classifier (Zhu, J. and Feng, Y., 2021).

Usage

```
## S3 method for class 'super_RaSE'
predict(object, newx, type = c("vote", "prob", "raw-vote", "raw-prob"), ...)
```

Arguments

`object`	fitted `'super_RaSE'` object using `Rase`.
`newx`	a set of new observations. Each row of `newx` is a new observation.
`type`	the type of prediction output. Can be 'vote', 'prob', 'raw-vote' or 'raw-prob'. Default = 'vote'. • vote: output the predicted class (by voting and cut-off) of new observations. Avalilable for all base learner types. • prob: output the predicted probabilities (posterior probability of each observation to be class 1) of new observations. It is the average probability over all base learners. • raw-vote: output the predicted class of new observations for all base learners. It is a `n` by `B1` matrix. `n` is the test sample size and `B1` is the number of base learners used in RaSE. Avalilable for all base learner types. • raw-prob: output the predicted probabilities (posterior probability of each observation to be class 1) of new observations for all base learners. It is a `n` by `B1` matrix.
`...`	additional arguments.

Value

depends on the parameter `type`. See the list above.

References

Zhu, J. and Feng, Y., 2021. Super RaSE: Super Random Subspace Ensemble Classification. https://www.preprints.org/ma

See Also

Rase.

Examples

```
## Not run:
set.seed(0, kind = "L'Ecuyer-CMRG")
train.data <- RaModel("classification", 1, n = 100, p = 50)
test.data <- RaModel("classification", 1, n = 100, p = 50)
xtrain <- train.data$x
ytrain <- train.data$y
xtest <- test.data$x
ytest <- test.data$y

# fit a super RaSE classifier by sampling base learner from kNN, LDA and
# logistic regression in equal probability
fit <- Rase(xtrain = xtrain, ytrain = ytrain, B1 = 100, B2 = 100,
base = c("knn", "lda", "logistic"), super = list(type = "separate", base.update = T),
criterion = "cv", cv = 5, iteration = 1, cores = 2)
ypred <- predict(fit, xtest)
mean(ypred != ytest)

## End(Not run)
```

`print.RaSE` *Print a fitted RaSE object.*

Description

Similar to the usual print methods, this function summarizes results. from a fitted `'RaSE'` object.

Usage

```
## S3 method for class 'RaSE'
print(x, ...)
```

Arguments

`x`	fitted `'RaSE'` model object.
`...`	additional arguments.

Value

No value is returned.

See Also

Rase.

Examples

```
set.seed(0, kind = "L'Ecuyer-CMRG")
train.data <- RaModel("classification", 1, n = 100, p = 50)
xtrain <- train.data$x
ytrain <- train.data$y

# test RaSE classifier with LDA base classifier
fit <- Rase(xtrain, ytrain, B1 = 50, B2 = 50, iteration = 0, cutoff = TRUE,
base = 'lda', cores = 2, criterion = 'ric', ranking = TRUE)

# print the summarized results
print(fit)
```

---

print.super_RaSE *Print a fitted super_RaSE object.*

Description

Similar to the usual print methods, this function summarizes results. from a fitted 'super_RaSE' object.

Usage

```
## S3 method for class 'super_RaSE'
print(x, ...)
```

Arguments

x	fitted 'super_RaSE' model object.
...	additional arguments.

Value

No value is returned.

See Also

Rase.

Examples

```
set.seed(0, kind = "L'Ecuyer-CMRG")
train.data <- RaModel("classification", 1, n = 100, p = 50)
xtrain <- train.data$x
ytrain <- train.data$y

# test RaSE classifier with LDA base classifier
fit <- Rase(xtrain, ytrain, B1 = 50, B2 = 50, iteration = 0, cutoff = TRUE,
base = 'lda', cores = 2, criterion = 'ric', ranking = TRUE)
```

```
# print the summarized results
print(fit)
```

RaModel — *Generate data (x, y) from various models in two papers.*

Description

`RaModel` generates data from 4 models described in Tian, Y. and Feng, Y., 2021(b) and 8 models described in Tian, Y. and Feng, Y., 2021(a).

Usage

```
RaModel(model.type, model.no, n, p, p0 = 1/2, sparse = TRUE)
```

Arguments

`model.type`	indicator of the paper covering the model, which can be 'classification' (Tian, Y. and Feng, Y., 2021(b)) or 'screening' (Tian, Y. and Feng, Y., 2021(a)).
`model.no`	model number. It can be 1-4 when `model.type` = 'classification' and 1-8 when `model.type` = 'screening', respectively.
`n`	sample size
`p`	data dimension
`p0`	marginal probability of class 0. Default = 0.5. Only used when `model.type` = 'classification' and `model.no` = 1, 2, 3.
`sparse`	a logistic object indicating model sparsity. Default = TRUE. Only used when `model.type` = 'classification' and `model.no` = 1, 4.

Value

`x`	n * p matrix. n observations and p features.
`y`	n responses.

Note

When `model.type` = 'classification' and `sparse` = TRUE, models 1, 2, 4 require $p \geq 5$ and model 3 requires $p \geq 50$. When `model.type` = 'classification' and `sparse` = FALSE, models 1 and 4 require $p \geq 50$ and $p \geq 30$, respectively. When `model.type` = 'screening', models 1, 4, 5 and 7 require $p \geq 4$. Models 2 and 8 require $p \geq 5$. Model 3 requires $p \geq 22$. Model 5 requires $p \geq 2$.

References

Tian, Y. and Feng, Y., 2021(a). RaSE: A variable screening framework via random subspace ensembles. Journal of the American Statistical Association, (just-accepted), pp.1-30.

Tian, Y. and Feng, Y., 2021(b). RaSE: Random subspace ensemble classification. Journal of Machine Learning Research, 22(45), pp.1-93.

See Also

`Rase`, `RaScreen`.

Examples

```
train.data <- RaModel("classification", 1, n = 100, p = 50)
xtrain <- train.data$x
ytrain <- train.data$y

## Not run:
train.data <- RaModel("screening", 2, n = 100, p = 50)
xtrain <- train.data$x
ytrain <- train.data$y

## End(Not run)
```

RaPlot — *Visualize the feature ranking results of a fitted RaSE object.*

Description

This function plots the feature ranking results from a fitted `'RaSE'` object via `ggplot2`. In the figure, x-axis represents the feature number and y-axis represents the selected percentage of each feature in B1 subspaces.

Usage

```
RaPlot(
  object,
  main = NULL,
  xlab = "feature",
  ylab = "selected percentage",
  ...
)
```

Arguments

`object`	fitted `'RaSE'` model object.
`main`	title of the plot. Default = `NULL`, which makes the title following the orm 'RaSE-base' with subscript i (rounds of iterations), where base represents the type of base classifier. i is omitted when it is zero.
`xlab`	the label of x-axis. Default = 'feature'.
`ylab`	the label of y-axis. Default = 'selected percentage'.
`...`	additional arguments.

Value

a `'ggplot'` object.

References

Tian, Y. and Feng, Y., 2021. RaSE: Random subspace ensemble classification. Journal of Machine Learning Research, 22(45), pp.1-93.

See Also

Rase.

Examples

```
set.seed(0, kind = "L'Ecuyer-CMRG")
train.data <- RaModel("classification", 1, n = 100, p = 50)
xtrain <- train.data$x
ytrain <- train.data$y

# fit RaSE classifier with QDA base classifier
fit <- Rase(xtrain, ytrain, B1 = 50, B2 = 50, iteration = 1, base = 'qda',
cores = 2, criterion = 'ric')

# plot the selected percentage of each feature appearing in B1 subspaces
RaPlot(fit)
```

RaRank *Rank the features by selected percentages provided by the output from* RaScreen.

Description

Rank the features by selected percentages provided by the output from RaScreen.

Usage

```
RaRank(object, selected.num = "all positive", iteration = object$iteration)
```

Arguments

object output from RaScreen.

selected.num the number of selected variables. User can either choose from the following popular options or input an positive integer no larger than the dimension.

- 'all positive': the number of variables with positive selected percentage.
- 'D': floor(D), where D is the maximum of ramdom subspace size.
- '1.5D': floor(1.5D).
- '2D': floor(2D).
- '3D': floor(3D).
- 'n/logn': floor(n/logn), where n is the sample size.
- '1.5n/logn': floor(1.5n/logn).
- '2n/logn': floor(2n/logn).
- '3n/logn': floor(3n/logn).
- 'n-1': the sample size n - 1.
- 'p': the dimension p.

iteration indicates results from which iteration to use. It should be an positive integer. Default = the maximal interation round used by the output from RaScreen.

Value

Selected variables (indexes).

References

Tian, Y. and Feng, Y., 2021(a). RaSE: A variable screening framework via random subspace ensembles. Journal of the American Statistical Association, (just-accepted), pp.1-30.

Examples

```
## Not run:
set.seed(0, kind = "L'Ecuyer-CMRG")
train.data <- RaModel("screening", 1, n = 100, p = 100)
xtrain <- train.data$x
ytrain <- train.data$y

# test RaSE screening with linear regression model and BIC
fit <- RaScreen(xtrain, ytrain, B1 = 100, B2 = 50, iteration = 0, model = 'lm',
cores = 2, criterion = 'bic')

# Select floor(n/logn) variables
RaRank(fit, selected.num = "n/logn")

## End(Not run)
```

`RaScreen` *Variable screening via RaSE.*

Description

`RaSE` is a general framework for variable screening. In RaSE screening, to select each of the B1 subspaces, B2 random subspaces are generated and the optimal one is chosen according to some criterion. Then the selected proportions (equivalently, percentages) of variables in the B1 subspaces are used as importance measure to rank these variables.

Usage

```
RaScreen(
  xtrain,
  ytrain,
  xval = NULL,
  yval = NULL,
  B1 = 200,
  B2 = NULL,
  D = NULL,
  dist = NULL,
  model = NULL,
  criterion = NULL,
  k = 5,
  cores = 1,
  seed = NULL,
  iteration = 0,
```

```
    cv = 5,
    scale = FALSE,
    C0 = 0.1,
    kl.k = NULL,
    classification = NULL,
    ...
)
```

Arguments

- `xtrain` — n * p observation matrix. n observations, p features.
- `ytrain` — n 0/1 observatons.
- `xval` — observation matrix for validation. Default = `NULL`. Useful only when `criterion` = 'validation'.
- `yval` — 0/1 observation for validation. Default = `NULL`. Useful only when `criterion` = 'validation'.
- `B1` — the number of weak learners. Default = 200.
- `B2` — the number of subspace candidates generated for each weak learner. Default = `NULL`, which will set B2 = 20 $* floor(p/D)$.
- `D` — the maximal subspace size when generating random subspaces. Default = `NULL`. It means that `D` = $min(\sqrt{n}0, \sqrt{n}1, p)$ when `model` = 'qda', and `D` = $min(\sqrt{n}, p)$ otherwise.
- `dist` — the distribution for features when generating random subspaces. Default = `NULL`, which represents the hierarchical uniform distribution. First generate an integer d from $1, ..., D$ uniformly, then uniformly generate a subset with cardinality d.
- `model` — the model to use. Default = 'lda' when `classification` = TRUE and 'lm' when `classification` = FALSE.
  - lm: linear regression. Only available for regression.
  - lda: linear discriminant analysis. `lda` in `MASS` package. Only available for classification.
  - qda: quadratic discriminant analysis. `qda` in `MASS` package. Only available for classification.
  - knn: k-nearest neighbor. `knn`, `knn.cv` in `class` package, `knn3` in `caret` package and `knnreg` in `caret` package.
  - logistic: logistic regression. `glmnet` in `glmnet` package. Only available for classification.
  - tree: decision tree. `rpart` in `rpart` package. Only available for classification.
  - svm: support vector machine. If kernel is not identified by user, it will use RBF kernel. `svm` in `e1071` package.
  - randomforest: random forest. `randomForest` in `randomForest` package and `ranger` in `ranger` package.
  - kernelknn: k-nearest neighbor with different kernels. It relies on function `KernelKnn` in `KernelKnn` package. Arguments `method` and `weights_function` are required. Different choices of multiple arguments are available. See documentation of function `KernelKnn` for details.
- `criterion` — the criterion to choose the best subspace. Default = 'ric' when `model` = 'lda', 'qda'; default = 'bic' when `model` = 'lm' or 'logistic'; default = 'loo' when `model` = 'knn'; default = 'cv' and set cv = 5 when `model` = 'tree', 'svm', 'randomforest'.

- ric: minimizing ratio information criterion (RIC) with parametric estimation (Tian, Y. and Feng, Y., 2020). Available for binary classification and `model` = 'lda', 'qda', or 'logistic'.
- nric: minimizing ratio information criterion (RIC) with non-parametric estimation (Tian, Y. and Feng, Y., 2020;). Available for binary classification and `model` = 'lda', 'qda', or 'logistic'.
- training: minimizing training error/MSE. Not available when `model` = 'knn'.
- loo: minimizing leave-one-out error/MSE. Only available when `model` = 'knn'.
- validation: minimizing validation error/MSE based on the validation data.
- cv: minimizing k-fold cross-validation error/MSE. k equals to the value of `cv`. Default = 5.
- aic: minimizing Akaike information criterion (Akaike, H., 1973). Available when `base` = 'lm' or 'logistic'.
  AIC = -2 * log-likelihood + |S| * 2.
- bic: minimizing Bayesian information criterion (Schwarz, G., 1978). Available when `model` = 'lm' or 'logistic'.
  BIC = -2 * log-likelihood + |S| * log(n).
- ebic: minimizing extended Bayesian information criterion (Chen, J. and Chen, Z., 2008; 2012). `gam` value is needed. When `gam` = 0, it represents BIC. Available when `model` = 'lm' or 'logistic'.
  eBIC = -2 * log-likelihood + |S| * log(n) + 2 * |S| * gam * log(p).

`k`	the number of nearest neighbors considered when `model` = 'knn' or 'kernel'. Only useful when `model` = 'knn' or 'kernel'. `k` is required to be a positive integer. Default = 5.
`cores`	the number of cores used for parallel computing. Default = 1.
`seed`	the random seed assigned at the start of the algorithm, which can be a real number or `NULL`. Default = `NULL`, in which case no random seed will be set.
`iteration`	the number of iterations. Default = 0.
`cv`	the number of cross-validations used. Default = 5. Only useful when `criterion` = 'cv'.
`scale`	whether to normalize the data. Logistic, default = FALSE.
`C0`	a positive constant used when `iteration` > 1. See Tian, Y. and Feng, Y., 2021 for details. Default = 0.1.
`kl.k`	the number of nearest neighbors used to estimate RIC in a non-parametric way. Default = `NULL`, which means that $k0 = floor(\sqrt{n}0)$ and $k1 = floor(\sqrt{n}1)$. See Tian, Y. and Feng, Y., 2020 for details. Only available when `criterion` = 'nric'.
`classification`	the indicator of the problem type, which can be TRUE, FALSE or `NULL`. Default = `NULL`, which will automatically set `classification` = TRUE if the number of unique response value $\leq$ 10. Otherwise, it will be set as FALSE.
`...`	additional arguments.

Value

A list including the following items.

`model`	the model used in RaSE screening.
`criterion`	the criterion to choose the best subspace for each weak learner.

B1	the number of selected subspaces.
B2	the number of subspace candidates generated for each of B1 subspaces.
n	the sample size.
p	the dimension of data.
D	the maximal subspace size when generating random subspaces.
iteration	the number of iterations.
selected.perc	A list of length (iteration+1) recording the selected percentages of each feature in B1 subspaces. When it is of length 1, the result will be automatically transformed to a vector.
scale	a list of scaling parameters, including the scaling center and the scale parameter for each feature. Equals to NULL when the data is not scaled by RaScreen.

References

Tian, Y. and Feng, Y., 2021(a). RaSE: A variable screening framework via random subspace ensembles. Journal of the American Statistical Association, (just-accepted), pp.1-30.

Tian, Y. and Feng, Y., 2021(b). RaSE: Random subspace ensemble classification. Journal of Machine Learning Research, 22(45), pp.1-93.

Chen, J. and Chen, Z., 2008. Extended Bayesian information criteria for model selection with large model spaces. Biometrika, 95(3), pp.759-771.

Chen, J. and Chen, Z., 2012. Extended BIC for small-n-large-P sparse GLM. Statistica Sinica, pp.555-574.

Schwarz, G., 1978. Estimating the dimension of a model. The annals of statistics, 6(2), pp.461-464.

See Also

Rase, RaRank.

Examples

```
set.seed(0, kind = "L'Ecuyer-CMRG")
train.data <- RaModel("screening", 1, n = 100, p = 100)
xtrain <- train.data$x
ytrain <- train.data$y

# test RaSE screening with linear regression model and BIC
fit <- RaScreen(xtrain, ytrain, B1 = 100, B2 = 50, iteration = 0, model = 'lm',
cores = 2, criterion = 'bic')

# Select D variables
RaRank(fit, selected.num = "D")


## Not run:
# test RaSE screening with knn model and 5-fold cross-validation MSE
fit <- RaScreen(xtrain, ytrain, B1 = 100, B2 = 50, iteration = 0, model = 'knn',
cores = 2, criterion = 'cv', cv = 5)

# Select n/logn variables
RaRank(fit, selected.num = "n/logn")
```

```
# test RaSE screening with SVM and 5-fold cross-validation MSE
fit <- RaScreen(xtrain, ytrain, B1 = 100, B2 = 50, iteration = 0, model = 'svm',
cores = 2, criterion = 'cv', cv = 5)

# Select n/logn variables
RaRank(fit, selected.num = "n/logn")


# test RaSE screening with logistic regression model and eBIC (gam = 0.5). Set iteration number = 1
train.data <- RaModel("screening", 6, n = 100, p = 100)
xtrain <- train.data$x
ytrain <- train.data$y

fit <- RaScreen(xtrain, ytrain, B1 = 100, B2 = 100, iteration = 1, model = 'logistic',
cores = 2, criterion = 'ebic', gam = 0.5)

# Select n/logn variables from the selected percentage after one iteration round
RaRank(fit, selected.num = "n/logn", iteration = 1)

## End(Not run)
```

`Rase` *Construct the random subspace ensemble classifier.*

Description

RaSE is a general ensemble classification framework to solve the sparse classification problem. In RaSE algorithm, for each of the B1 weak learners, B2 random subspaces are generated and the optimal one is chosen to train the model on the basis of some criterion.

Usage

```
Rase(
  xtrain,
  ytrain,
  xval = NULL,
  yval = NULL,
  B1 = 200,
  B2 = 500,
  D = NULL,
  dist = NULL,
  base = NULL,
  super = list(type = c("separate"), base.update = TRUE),
  criterion = NULL,
  ranking = TRUE,
  k = c(3, 5, 7, 9, 11),
  cores = 1,
  seed = NULL,
  iteration = 0,
  cutoff = TRUE,
  cv = 5,
  scale = FALSE,
```

```
  C0 = 0.1,
  kl.k = NULL,
  lower.limits = NULL,
  upper.limits = NULL,
  weights = NULL,
  ...
)
```

Arguments

- `xtrain` n * p observation matrix. n observations, p features.
- `ytrain` n 0/1 observatons.
- `xval` observation matrix for validation. Default = `NULL`. Useful only when `criterion` = 'validation'.
- `yval` 0/1 observation for validation. Default = `NULL`. Useful only when `criterion` = 'validation'.
- `B1` the number of weak learners. Default = 200.
- `B2` the number of subspace candidates generated for each weak learner. Default = 500.
- `D` the maximal subspace size when generating random subspaces. Default = `NULL`, which is $min(\sqrt{n}0, \sqrt{n}1, p)$ when `base` = 'qda' and is $min(\sqrt{n}, p)$ otherwise. For classical RaSE with a single classifier type, `D` is a positive integer. For super RaSE with multiple classifier types, `D` is a vector indicating different D values used for each base classifier type (the corresponding classifier types should be noted in the names of the vector).
- `dist` the distribution for features when generating random subspaces. Default = `NULL`, which represents the uniform distribution. First generate an integer d from $1, ..., D$ uniformly, then uniformly generate a subset with cardinality d.
- `base` the type of base classifier. Default = 'lda'. Can be either a single string chosen from the following options or a string/probability vector. When it indicates a single type of base classifiers, the classical RaSE model (Tian, Y. and Feng, Y., 2021(b)) will be fitted. When it is a string vector which includes multiple base classifier types, a super RaSE model (Zhu, J. and Feng, Y., 2021) will be fitted, by samling base classifiers with equal probabilty. It can also be a probability vector with row names corresponding to the specific classifier type, in which case a super RaSE model will be trained by sampling base classifiers in the given sampling probability.
  - lda: linear discriminant analysis. `lda` in `MASS` package.
  - qda: quadratic discriminant analysis. `qda` in `MASS` package.
  - knn: k-nearest neighbor. `knn`, `knn.cv` in `class` package and `knn3` in `caret` package.
  - logistic: logistic regression. `glm` in `stats` package and `glmnet` in `glmnet` package.
  - tree: decision tree. `rpart` in `rpart` package.
  - svm: support vector machine. `svm` in `e1071` package.
  - randomforest: random forest. `randomForest` in `randomForest` package.
  - gamma: Bayesian classifier for multivariate gamma distribution with independent marginals.

`super` a list of control parameters for super RaSE (Zhu, J. and Feng, Y., 2021). Not used when base equals to a single string. Should be a list object with the following components:

- type: the type of super RaSE. Currently the only option is 'separate', meaning that subspace distributions are different for each type of base classifiers.
- base.update: indicates whether the sampling probability of base classifiers should be updated during iterations or not. Logistic, default = TRUE.

`criterion` the criterion to choose the best subspace for each weak learner. For the classical RaSE (when `base` includes a single classifier type), default = 'ric' when `base` = 'lda', 'qda', 'gamma'; default = 'ebic' and set `gam` = 0 when `base` = 'logistic'; default = 'loo' when `base` = 'knn'; default = 'training' when `base` = 'tree', 'svm', 'randomforest'. For the super RaSE (when `base` indicates multiple classifiers or the sampling probability of multiple classifiers), default = 'cv' with the number of folds `cv` = 5, and it can only be 'cv', 'training' or 'auc'.

- ric: minimizing ratio information criterion with parametric estimation (Tian, Y. and Feng, Y., 2021(b)). Available when `base` = 'lda', 'qda', 'gamma' or 'logistic'.
- nric: minimizing ratio information criterion with non-parametric estimation (Tian, Y. and Feng, Y., 2021(b)). Available when `base` = 'lda', 'qda', 'gamma' or 'logistic'.
- training: minimizing training error. Not available when `base` = 'knn'.
- loo: minimizing leave-one-out error. Only available when `base` = 'knn'.
- validation: minimizing validation error based on the validation data. Available for all base classifiers.
- auc: minimizing negative area under the ROC curve (AUC). Currently it is estimated on training data via function `auc` from package `ModelMetrics`. It is available for all classier choices.
- cv: minimizing k-fold cross-validation error. k equals to the value of `cv`. Default = 5. Not available when `base` = 'gamma'.
- aic: minimizing Akaike information criterion (Akaike, H., 1973). Available when `base` = 'lda' or 'logistic'.
  AIC = -2 * log-likelihood + |S| * 2.
- bic: minimizing Bayesian information criterion (Schwarz, G., 1978). Available when `base` = 'lda' or 'logistic'.
  BIC = -2 * log-likelihood + |S| * log(n).
- ebic: minimizing extended Bayesian information criterion (Chen, J. and Chen, Z., 2008; 2012). Need to assign value for `gam`. When `gam` = 0, it denotes the classical BIC. Available when `base` = 'lda' or 'logistic'.
  EBIC = -2 * log-likelihood + |S| * log(n) + 2 * |S| * gam * log(p).

`ranking` whether the function outputs the selected percentage of each feature in B1 subspaces. Logistic, default = TRUE.

`k` the number of nearest neightbors considered when `base` = 'knn'. Only useful when `base` = 'knn'. Default = (3, 5, 7, 9, 11).

`cores` the number of cores used for parallel computing. Default = 1.

`seed` the random seed assigned at the start of the algorithm, which can be a real number or `NULL`. Default = `NULL`, in which case no random seed will be set.

`iteration` the number of iterations. Default = 0.

`cutoff` whether to use the empirically optimal threshold. Logistic, default = TRUE. If it is FALSE, the threshold will be set as 0.5.

`cv`	the number of cross-validations used. Default = 5. Only useful when `criterion` = 'cv'.
`scale`	whether to normalize the data. Logistic, default = FALSE.
`C0`	a positive constant used when `iteration` > 1. Default = 0.1. See Tian, Y. and Feng, Y., 2021(b) for details.
`kl.k`	the number of nearest neighbors used to estimate RIC in a non-parametric way. Default = `NULL`, which means that $k0 = floor(\sqrt{n}0)$ and $k1 = floor(\sqrt{n}1)$. See Tian, Y. and Feng, Y., 2021(b) for details. Only available when `criterion` = 'nric'.
`lower.limits`	the vector of lower limits for each coefficient in logistic regression. Should be a vector of length equal to the number of variables (the column number of `xtrain`). Each of these must be non-positive. Default = `NULL`, meaning that lower limits are `-Inf` for all coefficients. Only available when `base` = 'logistic'. When it's activated, function `glmnet` will be used to fit logistic regression models, in which case the minimum subspace size is required to be larger than 1. The default subspace size distribution will be changed to uniform distribution on (2, ..., D).
`upper.limits`	the vector of upper limits for each coefficient in logistic regression. Should be a vector of length equal to the number of variables (the column number of `xtrain`). Each of these must be non-negative. Default = `NULL`, meaning that upper limits are `Inf` for all coefficients. Only available when `base` = 'logistic'. When it's activated, function `glmnet` will be used to fit logistic regression models, in which case the minimum subspace size is required to be larger than 1. The default subspace size distribution will be changed to uniform distribution on (2, ..., D).
`weights`	observation weights. Should be a vector of length equal to training sample size (the length of `ytrain`). It will be normailized inside the algorithm. Each component of weights must be non-negative. Default is `NULL`, representing equal weight for each observation. Only available when `base` = 'logistic'. When it's activated, function `glmnet` will be used to fit logistic regression models, in which case the minimum subspace size is required to be larger than 1. The default subspace size distribution will be changed to uniform distribution on (2, ..., D).
`...`	additional arguments.

Value

An object with S3 class `'RaSE'` if base indicates a single base classifier.

`marginal`	the marginal probability for each class.
`base`	the type of base classifier.
`criterion`	the criterion to choose the best subspace for each weak learner.
`B1`	the number of weak learners.
`B2`	the number of subspace candidates generated for each weak learner.
`D`	the maximal subspace size when generating random subspaces.
`iteration`	the number of iterations.
`fit.list`	sequence of B1 fitted base classifiers.
`cutoff`	the empirically optimal threshold.

- `subspace` — sequence of subspaces correponding to B1 weak learners.
- `ranking` — the selected percentage of each feature in B1 subspaces.
- `scale` — a list of scaling parameters, including the scaling center and the scale parameter for each feature. Equals to `NULL` when the data is not scaled in `RaSE` model fitting.

An object with S3 class `'super_RaSE'` if `base` includes multiple base classifiers or the sampling probability of multiple classifiers.

- `marginal` — the marginal probability for each class.
- `base` — the list of B1 base classifier types.
- `criterion` — the criterion to choose the best subspace for each weak learner.
- `B1` — the number of weak learners.
- `B2` — the number of subspace candidates generated for each weak learner.
- `D` — the maximal subspace size when generating random subspaces.
- `iteration` — the number of iterations.
- `fit.list` — sequence of B1 fitted base classifiers.
- `cutoff` — the empirically optimal threshold.
- `subspace` — sequence of subspaces correponding to B1 weak learners.
- `ranking.feature` — the selected percentage of each feature corresponding to each type of classifier.
- `ranking.base` — the selected percentage of each classifier type in the selected B1 learners.
- `scale` — a list of scaling parameters, including the scaling center and the scale parameter for each feature. Equals to `NULL` when the data is not scaled in `RaSE` model fitting.

Author(s)

Ye Tian (maintainer, `<ye.t@columbia.edu>`) and Yang Feng. The authors thank Yu Cao (Exeter Finance) and his team for many helpful suggestions and discussions.

References

Tian, Y. and Feng, Y., 2021(a). RaSE: A variable screening framework via random subspace ensembles. Journal of the American Statistical Association, (just-accepted), pp.1-30.

Tian, Y. and Feng, Y., 2021(b). RaSE: Random subspace ensemble classification. Journal of Machine Learning Research, 22(45), pp.1-93.

Zhu, J. and Feng, Y., 2021. Super RaSE: Super Random Subspace Ensemble Classification. https://www.preprints.org/ma

Chen, J. and Chen, Z., 2008. Extended Bayesian information criteria for model selection with large model spaces. Biometrika, 95(3), pp.759-771.

Chen, J. and Chen, Z., 2012. Extended BIC for small-n-large-P sparse GLM. Statistica Sinica, pp.555-574.

Akaike, H., 1973. Information theory and an extension of the maximum likelihood principle. In 2nd International Symposium on Information Theory, 1973 (pp. 267-281). Akademiai Kaido.

Schwarz, G., 1978. Estimating the dimension of a model. The annals of statistics, 6(2), pp.461-464.

See Also

`predict.RaSE`, `RaModel`, `print.RaSE`, `print.super_RaSE`, `RaPlot`, `RaScreen`.

Examples

```
set.seed(0, kind = "L'Ecuyer-CMRG")
train.data <- RaModel("classification", 1, n = 100, p = 50)
test.data <- RaModel("classification", 1, n = 100, p = 50)
xtrain <- train.data$x
ytrain <- train.data$y
xtest <- test.data$x
ytest <- test.data$y

# test RaSE classifier with LDA base classifier
fit <- Rase(xtrain, ytrain, B1 = 100, B2 = 50, iteration = 0, base = 'lda',
cores = 2, criterion = 'ric')
mean(predict(fit, xtest) != ytest)

## Not run:
# test RaSE classifier with LDA base classifier and 1 iteration round
fit <- Rase(xtrain, ytrain, B1 = 100, B2 = 50, iteration = 1, base = 'lda',
cores = 2, criterion = 'ric')
mean(predict(fit, xtest) != ytest)

# test RaSE classifier with QDA base classifier and 1 iteration round
fit <- Rase(xtrain, ytrain, B1 = 100, B2 = 50, iteration = 1, base = 'qda',
cores = 2, criterion = 'ric')
mean(predict(fit, xtest) != ytest)

# test RaSE classifier with kNN base classifier
fit <- Rase(xtrain, ytrain, B1 = 100, B2 = 50, iteration = 0, base = 'knn',
cores = 2, criterion = 'loo')
mean(predict(fit, xtest) != ytest)

# test RaSE classifier with logistic regression base classifier
fit <- Rase(xtrain, ytrain, B1 = 100, B2 = 50, iteration = 0, base = 'logistic',
cores = 2, criterion = 'bic')
mean(predict(fit, xtest) != ytest)

# test RaSE classifier with SVM base classifier
fit <- Rase(xtrain, ytrain, B1 = 100, B2 = 50, iteration = 0, base = 'svm',
cores = 2, criterion = 'training')
mean(predict(fit, xtest) != ytest)

# test RaSE classifier with random forest base classifier
fit <- Rase(xtrain, ytrain, B1 = 20, B2 = 10, iteration = 0, base = 'randomforest',
cores = 2, criterion = 'cv', cv = 3)
mean(predict(fit, xtest) != ytest)

# fit a super RaSE classifier by sampling base learner from kNN, LDA and logistic
# regression in equal probability
fit <- Rase(xtrain = xtrain, ytrain = ytrain, B1 = 100, B2 = 100,
base = c("knn", "lda", "logistic"), super = list(type = "separate", base.update = T),
criterion = "cv", cv = 5, iteration = 1, cores = 2)
mean(predict(fit, xtest) != ytest)

# fit a super RaSE classifier by sampling base learner from random forest, LDA and
# SVM with probability 0.2, 0.5 and 0.3
fit <- Rase(xtrain = xtrain, ytrain = ytrain, B1 = 100, B2 = 100,
base = c(randomforest = 0.2, lda = 0.5, svm = 0.3),
```

```
super = list(type = "separate", base.update = F),
criterion = "cv", cv = 5, iteration = 0, cores = 2)
mean(predict(fit, xtest) != ytest)

## End(Not run)
```

---

rat *Affymetrix rat genome 230 2.0 array data set.*

Description

Affymetrix rat genome 230 2.0 array annotation data (chip rat2302). For this data set, 120 twelve-week old male rats were selected for tissue harvesting from the eyes and for microarray analysis. The expression of gene TRIM32 is set as the response and the 18975 probes that are expressed in the eye tissue are considered as the predictors.

Usage

```
rat
```

Format

A list with the predictor matrix `x` and the response vector `y`.

Source

The link to this data set: https://bioconductor.org/packages/release/data/annotation/html/rat2302.db.html

References

Scheetz, T.E., Kim, K.Y.A., Swiderski, R.E., Philp, A.R., Braun, T.A., Knudtson, K.L., Dorrance, A.M., DiBona, G.F., Huang, J., Casavant, T.L. and Sheffield, V.C., 2006. *Regulation of gene expression in the mammalian eye and its relevance to eye disease. Proceedings of the National Academy of Sciences, 103(39), pp.14429-14434.*

Tian, Y. and Feng, Y., 2021. *RaSE: A Variable Screening Framework via Random Subspace Ensembles. arXiv preprint arXiv:2102.03892.*

Index

∗ **datasets**
colon, 2
rat, 20

auc, *16*

colon, 2

glm, *15*
glmnet, *11*, *15*, *17*

KernelKnn, *11*
knn, *11*, *15*
knn.cv, *11*, *15*
knn3, *11*, *15*
knnreg, *11*

lda, *11*, *15*

predict.RaSE, 3, *18*
predict.super_RaSE, 4
print.RaSE, 5, *18*
print.super_RaSE, 6, *18*

qda, *11*, *15*

RaModel, 7, *18*
randomForest, *11*, *15*
ranger, *11*
RaPlot, 8, *18*
RaRank, 9, *13*
RaScreen, *7*, 10, *18*
Rase, *3*, *5–7*, *9*, *13*, 14
rat, 20
rpart, *11*, *15*

svm, *11*, *15*