Package 'RaSEn'

October 21, 2020

Type Package

Title Random Subspace Ensemble Classification

Version 1.1.0

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

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

Description We propose a flexible 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. And the choices of base classifiers are also various, for instance, linear discriminant analysis, quadratic discriminant analysis, k-nearest neighbor, 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. In addition, to relax the requirement of the number of random subspaces to be generated, we propose an iterative version of RaSE, which is shown to be effective under many sparse binary classification settings.

Imports MASS, caret, class, doParallel, e1071, foreach, nnet, randomForest, rpart, stats, ggplot2, gridExtra, formatR, FNN

License GPL-2

Encoding UTF-8

LazyData TRUE

RoxygenNote 7.1.0

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2020-10-21 04:40:02 UTC

R topics documented:

`predict.RaSE` *Predict the outcome of new observations based on the estimated RaSE classifier.*

Description

Predict the outcome of new observations based on the estimated RaSE classifier.

Usage

```
## S3 method for class 'RaSE'
predict(object, newx, ...)
```

Arguments

`object`	fitted `'RaSE'` object using `Rase`.
`newx`	a set of new observations. Each row of `newx` is a new observation.
`...`	additional arguments.

Value

The predicted labels for new observations.

See Also

`Rase`.

Examples

```
## Not run:
set.seed(0, kind = "L'Ecuyer-CMRG")
train.data <- RaModel(1, n = 100, p = 50)
test.data <- RaModel(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, cutoff = TRUE,
```

```
base = 'lda', cores = 2, criterion = 'ric', ranking = TRUE)
ypred <- predict(model.fit, xtest)

## 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(1, n = 100, p = 50)
test.data <- RaModel(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 = 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 6 models.*

Description

RaModel generates data from 6 models described in Tian, Y. and Feng, Y., 2020.

Usage

```
RaModel(Model.No, n, p, p0 = 1/2, sparse = TRUE)
```

Arguments

Model.No	model number, which can be 1, 2, 3, 4.
n	sample size
p	data dimension
p0	marginal probability of class 0. Default = 0.5. Only available when Model.No = 1, 2, 3.
sparse	a logistic object indicating model sparsity. Default = TRUE. Only available when Model.No = 1, 4. When it equals to FALSE, the data is generated from model 1' or 4' as described in Tian, Y. and Feng, Y., 2020.

Value

x	n * p matrix. n observations and p features.
y	n 0/1 observations.

Note

Models 1, 2 and 4 require $p \geq 5$. Models 1' and 3 requires $p \geq 50$. Model 4' requires $p \geq 30$.

References

Tian, Y. and Feng, Y., 2020. RaSE: Random subspace ensemble classification. arXiv preprint arXiv:2006.08855.

See Also

Rase

Examples

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

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.

See Also

Rase.

Examples

```
set.seed(0, kind = "L'Ecuyer-CMRG")
train.data <- RaModel(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)
```

Rase *Construct the random subspace ensemble classifier.*

Description

Rase is a novel model-free 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 = c("lda", "qda", "knn", "logistic", "tree", "svm", "randomforest", "gamma"),
  criterion = NULL,
  ranking = TRUE,
  k = c(3, 5, 7, 9, 11),
  cores = 1,
  seed = NULL,
  iteration = 0,
  cutoff = TRUE,
  cv = 10,
  scale = FALSE,
  C0 = 0.1,
  kl.k = 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 from the uniform distribution. Default = NULL, which is $min(\sqrt{n}0, \sqrt{n}1, p)$ when `base` = 'lda' and is $min(\sqrt{n}, p)$ otherwise.

`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'.

- 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. `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.

`criterion` the criterion to choose the best subspace for each weak learner. 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'.

- ric: minimizing ratio information criterion with parametric estimation (Tian, Y. and Feng, Y., 2020). Available when `base` = 'lda', 'qda', 'gamma' or 'logistic'.
- nric: minimizing ratio information criterion with non-parametric estimation (Tian, Y. and Feng, Y., 2020; Wang, Q., Kulkarni, S.R. and Verdú, S., 2009). 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.
- cv: minimizing k-fold cross-validation error. k equals to the value of `cv`. Default = 10. Not available when `base` = 'gamma'.
- 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', 'qda' 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'.
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 = 10. Only useful when criterion = 'cv'.
scale	whether to normalize the data. Logistic, default = FALSE.
C0	the threshold used to adjust the sampling probabilities of features when iteration > 0. Default = 0.1.
kl.k	the number of nearest neighbors used to estimate KL divergences when criterion = 'nric'. 2-dimensional vector. Default = NULL, in which case it will be set as $\sqrt{n}0, \sqrt{n}1$.
...	additional arguments.

Value

An object with S3 class 'RaSE'.

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.
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.
C0	the threshold used to adjust the sampling probabilities of features when iteration > 0.

References

Tian, Y. and Feng, Y., 2020. RaSE: Random subspace ensemble classification. arXiv preprint arXiv:2006.08855.

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.

Wang, Q., Kulkarni, S.R. and Verdú, S., 2009. Divergence estimation for multidimensional densities via $ k $-nearest-neighbor distances. IEEE Transactions on Information Theory, 55(5), pp.2392-2405.#' @examples

See Also

predict.RaSE, RaModel, print.RaSE, RaPlot.

Examples

```
set.seed(0, kind = "L'Ecuyer-CMRG")
train.data <- RaModel(1, n = 100, p = 50)
test.data <- RaModel(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 = 'ebic', gam = 0)
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)

## End(Not run)
```

Index