Package 'RFtools'

November 17, 2017

Title Miscellaneous Tools For Random Forest Models

Version 0.0.1

Description Functions to analyze Random Forest results or for tuning the models. Smoothing classes, creating null classes, multivariate structure test, flexibility test, post-hoc test for false positive discovery, robustness test.

URL `https://github.com/pmartinezarbizu/RFtools`

Maintainer Pedro Martínez Arbizu `<pmartinez@senckenberg.de>`

Depends R (>= 3.0.0)

License GPL (>= 3)

Encoding UTF-8

LazyData true

Imports randomForest, foreach, doParallel, MASS

RoxygenNote 6.0.1

NeedsCompilation no

Author Pedro Martínez Arbizu [aut, cre], Sven Rossel [aut]

R topics documented:

`add.null.class` *Add a Null Class*

Description

Create nMC observations of a null class.

Usage

```
add.null.class(target, predictors, nMC = 50)
```

Arguments

`target`	Column with names of classes.
`predictors`	The columns containig the predictor variables.
`nMC`	Number returned observations of the null class (default = 50)

Details

A null class has no multivariate structure. It is created by shuffling the rows across the whole dataset and then shuffling the column labels. By doing so we generate nMC in silico observations for a class containing only random variable values but preserving the variable space.

Value

Dataframe with original observatios and nMC observations of a null class. First column named 'class' contains the target classes, predictor names are mantained.

Author(s)

Pedro Martinez Arbizu & Sven Rossel

See Also

`smooth.data`

Examples

```
data(iris)
iris.null <- add.null.class(iris$Species,iris[,1:4])
summary(iris.null)
```

maldi *Example data maldi.RData for RFtools*

Description

This dataset is the intensity matrix generated by MALDI-TOF MS used in Rossel and Martínez Arbizu (submitted) The Data shows MALDI-TOF MS spectra of 173 harpacticoid copepod specimens belonging to 16 different species from the North Sea benthic environment resulting in 176 different mass spectra. Three specimens were cut into to parts before measurement, accordingly resulting in two mass spectra per specimen. Data was generated using a Microflex LT/SH System (Bruker Daltonics) with a laser intensity between 50 to 60 Mass spectra were measured with flexControl 3.4 software (factory delivered Bruker Daltonics software). Data was processed in R using packages MALDIquant and MALDIquantForeign, trimming mass spectra from 3000 to 15000 m/z. Spectra were smoothed (Savitzky & Golay), baseline removed (SNIP baseline estimation method) and spectra normalized using TIC method. Noise was estimated using a signal to noise ratio of 7. Subsequently peaks were binned once as implemented in MALDIquant with a tolerance of 0.002 in a strict approach. The resulting intensity matrix was hellinger transformed (square root of relative intensities) and is provided herewith. Data was generated at Senckenberg am Meer Proteome Laboratory, German Center for Marine Biodiversity Research (DZMB), Wilhelmshaven, Germany

Usage

```
data(maldi)
```

Format

A data frame with 176 rows (specimens), 543 variables(protein masses) and a grouping factor (species)

Details

- code Individual specimen code
- species Name of the harpacticoid species
- Columns 3 to 544: m/z bins of MALDI-TOF spectrum showing hellinger transformed intensity

Author(s)

Sven Rossel <sven.rossel@senckenberg.de>

`MVSF.test` *Multivariate Structure and Flexibility Test*

Description

Test the multivariate structure of your data against an empirical null model and the flexibility of your model to correctly classify new observations created by smoothing the classes.

Usage

```
MVSF.test(target, predictors, nMC = 999)
```

Arguments

`target`	Vector or target Column with names of classes.
`predictors`	Dataframe or matrix with predictor variables.
`nMC`	Number of smoothed observations returned per class (default = 999)

Details

Multivariate Test: The observed OOB is compared against an empirical null model with no multivariate structure. The null model is created by shuffling the lables of the classes nMC times and keeping the OOB of each class under the null model. A Pseudo P-value is returned as P = r/nMC-1, with r = the number of times that the null model OOB >= trained OOB, and nMC = number of Monte Carlo simulations.

Flexibility test: Create nMC of each class observations smoothing the classes and predict using the trained model. Return error rate as n/nMC, with n = total number of missclassifications and nMC = number of Monte Carlo simulations. prop.test is used for testing the null that trained OOB error = error of the smoothed observations

A plot method is available

Value

An object of class MVSF. Dataframe with results of multivariate test and flexibility test.

`oob.err`	OOB of trained model
`null.err`	mean OOB of null model
`P.null`	Pseudo P value, probability that trained OOB can happen by random
`q05.null`	0.5 quantile of the ecdf of the null error
`q95.null`	0.95 quantile of the ecdf of the null error
`smooth.err`	prediction error rate of the smoothed classes
`p.smooth`	p value of prop.test, probability that the trained error and the prediction error of smoothed classes are the same

Author(s)

Pedro Martinez Arbizu & Sven Rossel

See Also

`plot.MVSF` `robust.test`

Examples

```
data(iris)
MVSF.iris <- MVSF.test(iris$Species,iris[,1:4],nMC=99)
print(MVSF.iris)
plot(MVSF.iris)
```

`plot.MVSF` *Plot Results of MVSF Tests*

Description

Plot results of Multivariate Structure and Flexibility Test.

Usage

```
## S3 method for class 'MVSF'
plot(x, pnull = 0.05, psmooth = 0.05)
```

Arguments

`x`	Object created by MVSF.test()
`pnull`	Significance level for the multivariate structure test
`psmooth`	Significance level for the flexibility test

Details

For each class, the trained OOB, the prediction error of smoothed class and the error under null model is plotted. For the null error the 0.5-0.95 quantile interval is plotted as error line. Filled symbols indicate significant difference under alfa level defined by pnull and psmooth.

Author(s)

Pedro Martinez Arbizu

Examples

```
data(iris)
MVSF.iris <- MVSF.test(iris$Species,iris[,1:4])
print(MVSF.iris)
plot(MVSF.iris)
```

`plot.RFPH` *Plot Post-hoc test*

Description

Plots the results of rf.post.hoc.

Usage

```
## S3 method for class 'RFPH'
plot(x, Species = "all", q = 0.05)
```

Arguments

`x`	Object of class RFPH.
`Species`	Name of class to be plotted or 'all' for all classes
`q`	Significance level.

Details

If name of class to be plotted is specify, the plot returns the density function of the probability of assignment 'POA' to the correct class in the trainig data and the POA of the predicted data as circles symbols. Circle is filled 'red' if not significantly different at q level, otherwise 'white'. The POA at significance level q is plotted as dashed grey line.

If Species = 'all' (default), then all classes are plotted. Instead of density function for trained POA, the median (grey symbol) and the range between quantiles q and 1-q are plotted.

Value

Plot

Author(s)

Pedro Martinez Arbizu & Sven Rossel

See Also

`rf.post.hoc`

Examples

```
data(iris)
library(randomForest)
irSe <- iris[iris$Species == 'setosa',]
ir <- iris[iris$Species != 'setosa',]
ir$Species <- factor(ir$Species)
irNc <- add.null.class(ir$Species,ir[,1:4])
rf <- randomForest(class ~ ., data = irNc)
```

```
posth <- rf.post.hoc(rf,irSe)

plot(posth)
plot(posth,'versicolor',0.001)

#example with maldi data
data(maldi)
library(randomForest)
unique(maldi$species)
maldi_train <- maldi[maldi$species != 'Tachidius discipes',]
maldi_test <- maldi[maldi$species == 'Tachidius discipes',]
#exclude Tachidius discipes from factors
maldi_train$species <- factor(maldi_train$species)
rf <- randomForest(species ~ ., data = maldi_train[-1])
test <- rf.post.hoc(rf,maldi_test)
plot(test)
plot(test,'Cletodes limicola')
```

`rf.post.hoc` *RF Post-Hoc Test*

Description

Test for false positives in RF prediction.

Usage

```
rf.post.hoc(rf, newdata)
```

Arguments

`rf`	Object of class randomForest.
`newdata`	Dataframe to predict using RF model.

Details

RF Post-Hoc test: Take an object created with randomForest and a new dataset to predict. From the votes matrix of RF object take only correct assigments and calculate for each predicted class the beta distribution of the probability of assignment (POA) to the correct class using fitdistr() from MASS. In case fitdistr fails in estimating the parameters of beta, the non-parametric empirical cumulative distribution function (ecdf) is used instead. The for each observation in newdata, the probability that the POA to winner class (p.assign) belongs to the beta (or ecdf) of that class in the training data set is calculated (p.post.hoc). Use p.post.hoc to reject the hypothesis that the predicted POA belongs to the trained POA. A low p.post.hoc probability (e.g. <= 0.05) indicates a possible missclassification of that observation.

Methods plot, print and summary are available

Value

An object of class RF.ph List with results of Post-Hoc test .

- ecdf The empirical cumulative distribution fuction of the beta distributed RF probability of assignment to correct class.
- post.hoc Dataframe with results of post-hoc test.
  - winner Dataframe with results of post-hoc test.
  - p.assign Dataframe with results of post-hoc test.
  - p.post.hoc Dataframe with results of post-hoc test.
  - sig Significance code: 0 ***, 0.001 **, 0.01 *, 0.05 ., > ns

Author(s)

Pedro Martinez Arbizu & Sven Rossel

See Also

`add.null.class` `smooth.data` `robust.test`

Examples

```
data(iris)
library(randomForest)
irSe <- iris[iris$Species == 'setosa',]
ir <- iris[iris$Species != 'setosa',]
ir$Species <- factor(ir$Species)
irNc <- add.null.class(ir$Species,ir[,1:4])
rf <- randomForest(class ~ ., data = irNc)
table(predict(rf,irSe))
posth <- rf.post.hoc(rf,irSe)
summary(posth)
```

`robust.test` *RF Robustness Test*

Description

Test how prone is your RF model to misclassification of classes, that were not included in the training dataset after applying the post-hoc test.

Usage

```
robust.test(target, predictors, nMC = 999, q = 0.05)
```

Arguments

target	Vector or target Column with names of classes.
predictors	Dataframe or matrix with predictor variables.
nMC	Number of smoothed observations returned per class (default = 999)
q	Number, quantile to use for the post-hoc test, default 0.05

Details

RF Robustness Test: Models are calculated leaving one class out. Observations of the class that is excluded from the training dataset are used to predict with the trained model. It is evident that these observations cannot belong to any of the target classes. After prediction, the post-hoc test for false positive discovery is applied to the results. A residual false positive rate is calculated. Row lables display the class that was excluded from the model. Column names display the predicted class.

Value

An object of class RBT. List with results of RF robustness test.

assignment	Table with the class assigments per model
false.pos	Table with the false positives, after post-hoc test, per model
false.pos.rate	Table with the false positive rate, after post-hoc test, per model

Author(s)

Pedro Martinez Arbizu

See Also

add.null.class smooth.data

Examples

```
data(iris)
rbt.iris <- robust.test(iris$Species,iris[,1:4])
print(rbt.iris)
```

`smooth.data` *Smooth Classes*

Description

Create a new dataset with smoothed classes and nMC observations per class.

Usage

```
smooth.data(target, predictors, nMC = 100)
```

Arguments

`target`	Vector or target Column with names of classes.
`predictors`	Dataframe or matrix with predictor variables.
`nMC`	Number of smoothed observations returned per class (default = 100)

Details

For each class, the function will shuffle the rows of each predictor separately and extract nMC rows for each class. By doing so we generate nMC in silico observations for each class, but mantaining the range (observed variability) for each predictor.

Value

Dataframe with nMC smoothed observations. First column named 'class' contains the target classes, predictor names are mantained.

Author(s)

Pedro Martinez Arbizu & Sven Rossel

See Also

`add.null.class`

Examples

```
data(iris)
iris.sm <- smooth.data(iris$Species,iris[,1:4])
summary(iris.sm)
```