Package 'CompositionalML'

March 14, 2024

Type Package

Title Machine Learning with Compositional Data

Version 1.0

Date 2024-03-13

Author Michail Tsagris [aut, cre]

Maintainer Michail Tsagris <mtsagris@uoc.gr>

Depends R (>= 4.0)

Imports Boruta, Compositional, doParallel, e1071, foreach, graphics, ranger, Rfast, Rfast2, stats

Description Machine learning algorithms for predictor variables that are compositional data and the response variable is either continuous or categorical. Specifically, the Boruta variable selection algorithm, random forest, support vector machines and projection pursuit regression are included. Relevant papers include: Tsagris M.T., Preston S. and Wood A.T.A. (2011). ``A data-based power transformation for compositional data''. Fourth International International Workshop on Compositional Data Analysis. <doi:10.48550/arXiv.1106.1451> and Alenazi, A. (2023). ``A review of compositional data analysis and recent advances''. Communications in Statistics--Theory and Methods, 52(16): 5535--5567. <doi:10.1080/03610926.2021.2014890>.

License GPL (>= 2)

NeedsCompilation no

Repository CRAN

Date/Publication 2024-03-14 12:40:02 UTC

R topics documented:

CompositionalML-package

Machine Learning with Compositional Data

Description

Description: Machine learning algorithms for predictor variables that are compositional data and the response variable is either continuous or categorical. Specifically, the Boruta variable selection algorithm, random forest, support vector machines and projection pursuit regression are included. Relevant papers include: Tsagris M.T., Preston S. and Wood A.T.A. (2011). "A data-based power transformation for compositional data". Fourth International International Workshop on Compositional Data Analysis and Alenazi, A. (2023). "A review of compositional data analysis and recent advances". Communications in Statistics–Theory and Methods, 52(16): 5535–5567.

Details

Package:	CompositionalML
Type:	Package
Version:	1.0
Date:	2024-03-13
License:	GPL-2

Maintainers

Michail Tsagris <mtsagris@uoc.gr>

Author(s)

Michail Tsagris <mtsagris@uoc.gr>.

References

Alenazi A. (2023). A review of compositional data analysis and recent advances. Communications in Statistics–Theory and Methods, 52(16): 5535–5567.

Friedman J. H. and Stuetzle W. (1981). Projection pursuit regression. Journal of the American Statistical Association, 76, 817-823. doi: 10.2307/2287576.

Friedman Jerome, Trevor Hastie and Robert Tibshirani (2009). The elements of statistical learning, 2nd edition. Springer, Berlin.

Chang Chih-Chung and Lin Chih-Jen: LIBSVM: a library for Support Vector Machines https://www.csie.ntu.edu.tw/~cjlin/lib

Kursa M. B. and Rudnicki W. R. (2010). Feature Selection with the Boruta Package. Journal of Statistical Software, 36(11).

Tsagris M.T., Preston S. and Wood A.T.A. (2011). A data-based power transformation for compositional data. Fourth International International Workshop on Compositional Data Analysis.

Wright M. N. and Ziegler A. (2017). ranger: A fast implementation of random forests for high dimensional data in C++ and R. Journal of Statisrical Software 77:1-17. doi:10.18637/jss.v077.i01.

`alfa-PPR with compositional predictor variables`
α-PPR with compositional predictor variables

Description

α-PPR with compositional predictor variables.

Usage

```
alfa.ppr(xnew, y, x, a = seq(-1, 1, by = 0.1), nterms = 1:10)
```

Arguments

`xnew`	A matrix with the new compositional data whose group is to be predicted. Zeros are allowed, but you must be careful to choose strictly positive vcalues of α.
`y`	The response variable, a numerical vector.
`x`	A matrix with the compositional data.
`a`	A vector with a grid of values of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If a=0, the isometric log-ratio transformation is applied.
`nterms`	The number of terms to include in the model.

Details

This is the standard projection pursuit regression (PPR) applied to the α-transformed compositional predictors. See the built-in function "ppr" for more details.

Value

A list including:

`mod`	A list with the results of the PPR model for each value of α that includes the PPR output as provided by the function "ppr", for each value of "nterms".
`est`	A list with the predicted response values of "xnew" for each value of α and number of "nterms".

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <`mtsagris@uoc.gr`>.

References

Friedman J. H. and Stuetzle W. (1981). Projection pursuit regression. Journal of the American Statistical Association, 76, 817-823. doi: 10.2307/2287576.

Tsagris M.T., Preston S. and Wood A.T.A. (2011). A data-based power transformation for compositional data. In Proceedings of the 4th Compositional Data Analysis Workshop, Girona, Spain. https://arxiv.org/pdf/1106.1451.pdf

See Also

`alfappr.tune`

Examples

```
x <- as.matrix(iris[, 1:3])
x <- x/ rowSums(x)
y <- iris[, 4]
mod <- alfa.ppr(x, y, x, a = c(0, 0.5, 1), nterms = c(2, 3))
mod
```

`alpha-Boruta` *α-Boruta variable selection*

Description

α-Boruta variable selection.

Usage

```
alfa.boruta(y, x,  a = seq(-1, 1, by = 0.1), runs = 100 )
```

Arguments

`y` The response variable, it can either be a factor (for classification) or a numeric vector (for regression). Depending on the nature of the response variable, the function will proceed with the necessary task.

`x` A matrix with the compositional data.

`a` A vector with a grid of values of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If a=0, the isometric log-ratio transformation is applied.

`runs` Maximal number of importance source runs. You may increase it to avoid variables being characterised as "Tentative".

Details

For each value of α, the compositional data are transformed and then the Boruta variable selection algorithm is applied.

Value

A list with the results of the Boruta variable selection algortihm for each value of α as returned by the function "Boruta" of the package **Boruta**. The important elements are these (all the items returned by the function are of course included):

`finalDecision` A factor of three values: "Confirmed", "Rejected" or "Tentative"", for each variable, containing the final result of the variable selection.

`ImpHistory` A data frame of importances of variables gathered in each importance source run. Beside the importances, it contains maximal, mean and minimal importance of shadow variables in each run. Rejected attributes get -Inf importance. Set to NULL if holdHistory was given FALSE.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <`mtsagris@uoc.gr`>.

References

Kursa M. B. and Rudnicki W. R. (2010). Feature Selection with the Boruta Package. Journal of Statistical Software, 36(11).

Tsagris M.T., Preston S. and Wood A.T.A. (2011). A data-based power transformation for compositional data. In Proceedings of the 4th Compositional Data Analysis Workshop, Girona, Spain. https://arxiv.org/pdf/1106.1451.pdf

See Also

`alfa.rf`

Examples

```
x <- as.matrix(iris[, 1:4])
x <- x/ rowSums(x)
y <- iris[, 5]
mod <- alfa.boruta(y, x, a = c(0, 0.5))
mod
```

`alpha-RF` *α-RF*

Description

α-RF.

Usage

```
alfa.rf(xnew, y, x, a = seq(-1, 1, by = 0.1), size = c(1, 2, 3),
depth = c(0, 1), splits = 2:5, R = 500)
```

Arguments

`xnew`	A matrix with the new compositional data whose group is to be predicted. Zeros are allowed, but you must be careful to choose strictly positive vcalues of α.
`y`	The response variable, it can either be a factor (for classification) or a numeric vector (for regression). Depending on the nature of the response variable, the function will proceed with the necessary task.
`x`	A matrix with the compositional data.
`a`	A vector with a grid of values of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If a=0, the isometric log-ratio transformation is applied.
`size`	The minimal node size to split at.
`depth`	The maximal tree depth. A value of NULL or 0 corresponds to unlimited depth, 1 to tree stumps (1 split per tree).
`splits`	The number of random splits to consider for each candidate splitting variable.
`R`	The number of trees.

Details

For each value of α, the compositional data are transformed and then the random forest (RF) is applied for one or more combinations of the hyper-parameters.

Value

A list including:

`mod`	A list with the results of the RF model for each value of α that includes the RF output (a ranger class object) as provided by the function "ranger" of the package **ranger**, the configurations used and the predicted values of "xnew".

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <`mtsagris@uoc.gr`>.

References

Wright M. N. and Ziegler A. (2017). ranger: A fast implementation of random forests for high dimensional data in C++ and R. Journal of Statisrical Software 77:1-17. doi:10.18637/jss.v077.i01.

Tsagris M.T., Preston S. and Wood A.T.A. (2011). A data-based power transformation for compositional data. In Proceedings of the 4th Compositional Data Analysis Workshop, Girona, Spain. https://arxiv.org/pdf/1106.1451.pdf

See Also

`alfarf.tune`

Examples

```
x <- as.matrix(iris[, 1:4])
x <- x/ rowSums(x)
y <- iris[, 5]
mod <- alfa.rf(x, y, x, a = c(0, 0.5, 1), size = 3, depth = 1, splits = 2:3, R = 500)
mod
```

alpha-SVM *α-SVM*

Description

α-SVM.

Usage

```
alfa.svm(xnew, y, x, a = seq(-1, 1, by = 0.1), cost = seq(0.2, 2, by = 0.2), gamma = NULL)
```

Arguments

`xnew` A matrix with the new compositional data whose group is to be predicted. Zeros are allowed, but you must be careful to choose strictly positive vcalues of α.

`y` The response variable, it can either be a factor (for classification) or a numeric vector (for regression). Depending on the nature of the response variable, the function will proceed with the necessary task.

`x` A matrix with the compositional data.

`a` A vector with a grid of values of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If a=0, the isometric log-ratio transformation is applied.

`cost` A grid of values for the cost of constraints violation. The cost is the "C"-constant of the regularization term in the Lagrange formulation.

`gamma` A grid of values for the γ parameter of the Gaussian kernel. If no values are supplied the default grid is used, ten equidistant values from $1/D^2$ to $\sqrt{D}$,

Details

For each value of α, the compositional data are transformed and then the SVM is applied for one or more combinations of the cost and γ parameters.

Value

A list including:

`mod` A list with the results of the SVM model for each value of α that includes the SVM output (an svm class object) as provided by the function "svm" of the package **e1071**, the configurations used (cost and γ values) and the predicted values of "xnew".

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <`mtsagris@uoc.gr`>.

References

Chang Chih-Chung and Lin Chih-Jen: LIBSVM: a library for Support Vector Machines https://www.csie.ntu.edu.tw/~cjlin/lib

Tsagris M.T., Preston S. and Wood A.T.A. (2011). A data-based power transformation for compositional data. In Proceedings of the 4th Compositional Data Analysis Workshop, Girona, Spain. https://arxiv.org/pdf/1106.1451.pdf

See Also

`alfasvm.tune`

Examples

```
x <- as.matrix(iris[, 1:4])
x <- x/ rowSums(x)
y <- iris[, 5]
mod <- alfa.svm(x, y, x, a = c(0, 0.5, 1), cost = c(0.2, 0.4), gamma = c(0.1, 0.2) )
mod
```

`Tuning the parameters of the alfa-PPR`

Tuning the parameters of theα-PPR

Description

Tuning the parameters of theα-PPR.

Usage

```
alfappr.tune(y, x, a = seq(-1, 1, by = 0.1), nterms = 1:10, ncores = 1,
folds = NULL, nfolds = 10, seed = NULL, graph = FALSE)
```

Arguments

`y`	The response variable, a numerical vector.
`x`	A matrix with the compositional data.
`a`	A vector with a grid of values of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If a=0, the isometric log-ratio transformation is applied.
`nterms`	The number of terms to include in the model.

`ncores` The number of cores to use. If more than 1, parallel computing will take place. It is advisable to use it if you have many observations and or many variables, otherwise it will slow down the process.

`folds` If you have the list with the folds supply it here. You can also leave it NULL and it will create folds.

`nfolds` The number of folds in the cross validation.

`seed` You can specify your own seed number here or leave it NULL.

`graph` If graph is TRUE (default value) a plot will appear.

Details

K-fold cross-validation of the α-PPR with compositional predictor variables is performed to select the optimal value of α and the numer of terms in the PPR.

Value

If graph is true, a graph with the estimated performance for each value of α. A list including:

`per` A vector with the estimated performance for each value of α.

`performance` A vector with the optimal performance and the optimal number of terms.

`best_a` The value of α corresponding to the optimal performance.

`runtime` The time required by the cross-validation procedure.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <`mtsagris@uoc.gr`>.

References

Friedman J. H. and Stuetzle W. (1981). Projection pursuit regression. Journal of the American Statistical Association, 76, 817-823. doi: 10.2307/2287576.

Friedman Jerome, Trevor Hastie and Robert Tibshirani (2009). The elements of statistical learning, 2nd edition. Springer, Berlin.

Tsagris M.T., Preston S. and Wood A.T.A. (2011). A data-based power transformation for compositional data. In Proceedings of the 4th Compositional Data Analysis Workshop, Girona, Spain. https://arxiv.org/pdf/1106.1451.pdf

See Also

`alfa.ppr`

Examples

```
x <- as.matrix(iris[, 1:3])
x <- x/ rowSums(x)
y <- iris[, 4]
mod <- alfappr.tune(y, x, a = c(0, 0.5, 1), nterms = c(2, 3))
mod
```

`Tuning the parameters of the alpha-RF`

Tuning the parameters of the α-RF

Description

Tuning the parameters of the α-RF.

Usage

```
alfarf.tune(y, x, a = seq(-1, 1, by = 0.1), size = c(1, 2, 3),
depth = c(0, 1), splits = 2:5, R = 500, ncores = 1, folds = NULL,
nfolds = 10, stratified = TRUE, seed = NULL, graph = FALSE)
```

Arguments

`y` The response variable, it can either be a factor (for classification) or a numeric vector (for regression). Depending on the nature of the response variable, the function will proceed with the necessary task.

`x` A matrix with the compositional data.

`a` A vector with a grid of values of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If a=0, the isometric log-ratio transformation is applied.

`size` The minimal node size to split at.

`depth` The maximal tree depth. A value of NULL or 0 corresponds to unlimited depth, 1 to tree stumps (1 split per tree).

`splits` The number of random splits to consider for each candidate splitting variable.

`R` The number of trees.

`ncores` The number of cores to use. If more than 1, parallel computing will take place. It is advisable to use it if you have many observations and or many variables, otherwise it will slow down the process.

`folds` If you have the list with the folds supply it here. You can also leave it NULL and it will create folds.

`nfolds` The number of folds in the cross validation.

`stratified` Do you want the folds to be created in a stratified way? TRUE or FALSE.

`seed` You can specify your own seed number here or leave it NULL.

`graph` If graph is TRUE (default value) a plot will appear.

Details

K-fold cross-validation of the α-RF with compositional predictor variables is performed to select the optimal value of α and the optimal configutrations of the random forest (RF).

Value

If graph is true, a graph with the estimated performance for each value of α. A list including:

`per` A vector with the estimated performance for each value of α.

`performance` A vector with the optimal performance and the optimal combinations of the hyper-parameters of the RF.

`best_a` The value of α corresponding to the optimal performance.

`runtime` The time required by the cross-validation procedure.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <`mtsagris@uoc.gr`>.

References

Wright M. N. and Ziegler A. (2017). ranger: A fast implementation of random forests for high dimensional data in C++ and R. Journal of Statisrical Software 77:1-17. doi:10.18637/jss.v077.i01.

Friedman Jerome, Trevor Hastie and Robert Tibshirani (2009). The elements of statistical learning, 2nd edition. Springer, Berlin.

Tsagris M.T., Preston S. and Wood A.T.A. (2011). A data-based power transformation for compositional data. In Proceedings of the 4th Compositional Data Analysis Workshop, Girona, Spain. https://arxiv.org/pdf/1106.1451.pdf

See Also

`alfa.rf`

Examples

```
x <- as.matrix(iris[, 1:4])
x <- x/ rowSums(x)
y <- iris[, 5]
mod <- alfa.rf(x, y, x, a = c(0, 0.5, 1), size = 3, depth = 1, splits = 2:3, R = 500)
mod
```

`Tuning the parameters of the alpha-SVM`

Tuning the parameters of the α-SVM

Description

Tuning the parameters of the α-SVM.

Usage

```
alfasvm.tune(y, x, a = seq(-1, 1, by = 0.1), cost = seq(0.2, 2, by = 0.2), gamma = NULL,
ncores = 1, folds = NULL, nfolds = 10, stratified = TRUE, seed = NULL, graph = FALSE)
```

Arguments

`y` The response variable, it can either be a factor (for classification) or a numeric vector (for regression). Depending on the nature of the response variable, the function will proceed with the necessary task.

`x` A matrix with the compositional data.

`a` A vector with a grid of values of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If a=0, the isometric log-ratio transformation is applied.

`cost` A grid of values for the cost of constraints violation. The cost is the "C"-constant of the regularization term in the Lagrange formulation.

`gamma` A grid of values for the γ parameter of the Gaussian kernel. If no values are supplied the default grid is used, ten equidistant values from $1/D^2$ to $\sqrt{D}$,

`ncores` The number of cores to use. If more than 1, parallel computing will take place. It is advisable to use it if you have many observations and or many variables, otherwise it will slow down the process.

`folds` If you have the list with the folds supply it here. You can also leave it NULL and it will create folds.

`nfolds` The number of folds in the cross validation.

`stratified` Do you want the folds to be created in a stratified way? TRUE or FALSE.

`seed` You can specify your own seed number here or leave it NULL.

`graph` If graph is TRUE (default value) a plot will appear.

Details

K-fold cross validation is performed to select the optimal parameters for the SVM and also estimate the rate of accuracy. For continuous responses the estimated performance translates to the MSE, while for categorical responses (factors) this is the accuracy (percentage of crrect classification).

Value

If graph is true, a graph with the estimated performance for each value of α. A list including:

`per` A vector with the estimated performance for each value of α.

`performance` A vector with the optimal performance and the optimal combinations of cost and γ values.

`best_a` The value of α corresponding to the optimal performance.

`runtime` The time required by the cross-validation procedure.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

Chang Chih-Chung and Lin Chih-Jen: LIBSVM: a library for Support Vector Machines https://www.csie.ntu.edu.tw/~cjlin/lib

Friedman Jerome, Trevor Hastie and Robert Tibshirani (2009). The elements of statistical learning, 2nd edition. Springer, Berlin.

Tsagris M.T., Preston S. and Wood A.T.A. (2011). A data-based power transformation for compositional data. In Proceedings of the 4th Compositional Data Analysis Workshop, Girona, Spain. https://arxiv.org/pdf/1106.1451.pdf

See Also

alfa.svm

Examples

```
x <- as.matrix(iris[, 1:4])
x <- x/ rowSums(x)
y <- iris[, 5]
mod <- alfasvm.tune(y, x, a = c(0, 0.5, 1), cost = c(0.2, 0.4), gamma = c(0.1, 0.2) )
mod
```

Index