Package: PrInDT (via r-universe)

September 11, 2024

Type Package

Title Prediction and Interpretation in Decision Trees for Classification and Regression

Version 1.0.1

Description Optimization of conditional inference trees from the package 'party' for classification and regression. For optimization, the model space is searched for the best tree on the full sample by means of repeated subsampling. Restrictions are allowed so that only trees are accepted which do not include pre-specified uninterpretable split results (cf. Weihs & Buschfeld, 2021a). The function PrInDT() represents the basic resampling loop for 2-class classification (cf. Weihs & Buschfeld, 2021a). The function RePrInDT() (repeated PrInDT()) allows for repeated applications of PrInDT() for different percentages of the observations of the large and the small classes (cf. Weihs & Buschfeld, 2021c). The function NesPrInDT() (nested PrInDT()) allows for an extra layer of subsampling for a specific factor variable (cf. Weihs & Buschfeld, 2021b). The functions PrInDTMulev() and PrInDTMulab() deal with multilevel and multilabel classification. In addition to these PrInDT() variants for classification, the function PrInDTreg() has been developed for regression problems. Finally, the function PostPrInDT() allows for a posterior analysis of the distribution of a specified variable in the terminal nodes of a given tree. References are: -- Weihs, C., Buschfeld, S. (2021a) ``Combining Prediction and Interpretation in Decision Trees (PrInDT) - a Linguistic Example'' <arXiv:2103.02336>; -- Weihs, C., Buschfeld, S. (2021b) ``NesPrInDT: Nested undersampling in PrInDT'' <arXiv:2103.14931>; -- Weihs, C., Buschfeld, S. (2021c) ``Repeated undersampling in PrInDT (RePrInDT): Variation in undersampling and prediction, and ranking of predictors in ensembles'' <arXiv:2108.05129>.

License GPL-2

Encoding UTF-8

Imports graphics, MASS, party, splitstackshape, stats, stringr, utils

RoxygenNote 7.2.3

Depends R (>= 2.10)

NeedsCompilation no

Maintainer Claus Weihs <claus.weihs@tu-dortmund.de>

LazyData true

Author Claus Weihs [aut, cre], Sarah Buschfeld [aut], Niklas Nitsch [ctb]

Repository CRAN

Date/Publication 2023-05-09 22:20:02 UTC

Contents

`data_land` *Landscape analysis*

Description

The use of language(s) on public signs and categorization criteria.

Usage

```
data_land
```

Format

A data frame with 149 observations and 28 columns

coder who coded the sign: Factor (3 levels "C","E","S") (anonymized)

researcher who took a photograph of the sign: Factor (2 levels "L","S") (anonymized)

sign where the sign was found: Factor (11 levels "digi","door","graf",...)

type.of.sign kind of sign: Factor (5 levels "com","commem","infra","reg","trans")

permanent was the sign permanent? Factor (2 levels "no","yes")

proper.noun kind of proper noun on the sign: Factor (9 levels "bn","bn+","cn",...)

no.languages number of languages on sign: Factor (4 levels "1","2","3","4+")

French French on sign? Factor (2 levels "0","1")

Dutch Dutch on sign? Factor (2 levels "0","1")

English English on sign? Factor (2 levels "0","1")

Italian Italian on sign? Factor (2 levels "0","1")

Spanish Spanish on sign? Factor (2 levels "0","1")

German German on sign? Factor (2 levels "0","1")

Indian,Mandarin,Portuguese,Libanese,Japanese,Danish,Hebrew,Catalan 8 infrequent languages: Factor (2 levels "0","1")

bn.unclear brand name unclear: Factor (2 levels "0","1")

multilingual.type type of multilingualism on sign: Factor (6 levels "0","1","2","3","4","uc")

location location of sign: Factor (2 levels "M","P") (anonymized)

Source

Sarah Buschfeld, TU Dortmund

`data_speaker` *Subject pronouns and a predictor with one very frequent level*

Description

Usage of subject pronouns and its predictors; speaker level "adult" very frequent.

Usage

```
data_speaker
```

Format

A data frame with 3370 observations and 6 columns

class subject pronoun realized? Factor (2 levels "zero","realized")

AGE age: Numerical (in months)

ETHN_GROUP ethnic group: Factor (3 levels "C","I","n_a") (anonymized)

MLU mean length of utterance: Factor (5 levels "1","2","3","adult","OL")

PRN_TYPE pronoun type: Factor (5 levels "dem","it_con","it_ex","it_ref","refer")

SPEAKER speaker: Factor (2 levels "adult","child")

Source

Sarah Buschfeld, TU Dortmund

`data_vowel` *Vowel length*

Description

Vowel length and categorization criteria.

Usage

```
data_vowel
```

Format

A data frame with 82 observations and 22 columns

Nickname nickname: Factor (43 levels "Nick1","Nick2",...,"Nick43") (anonymized)

LiBa linguistic background: Factor (2 levels "mono","multi")

MLU mean length of utterance: Factor (3 levels "1","2","3")

phone_label phone label: Factor (2 levels "fleece","kit")

lexeme lexeme: Factor (14 levels "bee","cheek","cheese","chicken", ...)

phone_left_1_duration duration of phone to the left of the vowel: Numerical (in msec)

phone_right_1_duration duration of phone to the right of the vowel: Numerical (in msec)

word_duration duration of word: Numerical (in msec)

vowel_minimum_pitch minimum pitch of vowel: Numerical (in Hertz)

vowel_maximum_pitch maximum pitch of vowel: Numerical (in Hertz)

vowel_intensity_mean mean intensity of vowel: Numerical (in decibel)

f1_fifty first formant F1 at midpoint of vowel (50%): Numerical (in Hertz)

f2_fifty second formant F2 at midpoint of vowel (50%): numercial (in Hertz)

target vowel length: Numerical (in msec)

cons_class_l class of consonant to the left of the vowel: Factor (6 levels "l","r","tsh",...)

cons_class_r class of consonant to the right of the vowel: Factor (7 levels "?"(glottal stop),"empty","nas",...)

ETH ethnic group: Factor (6 levels "C1a","C1b","C1c","C2a","C2b","C2c") (anonymized)

SEX gender: Factor (2 levels "female","male)

AGE age: Numerical (in months)

syllables number of syllables in lexeme: integer (1,2)

speed speed of speech: Numerical (word duration / syllables; in msec)

country country: Factor (2 levels "E","S") (anonymized)

Source

Sarah Buschfeld, TU Dortmund

`data_zero` *Subject pronouns*

Description

Usage of subject pronouns and its predictors.

Usage

```
data_zero
```

Format

A data frame with 1024 observations and 7 columns

real subject pronoun realized? Factor (2 levels "zero","realized")

AGE age: Numerical (in months)

LiBa linguistic background: Factor (3 levels "mono","multi", NA)

ETH ethnic group: Factor (6 levels "C1a","C1b","C1c","C2a","C2b","C2c") (anonymized)

SEX gender: Factor (2 levels "female","male")

MLU mean length of utterance: Factor (4 levels "1","2","3","OL")

PRN pronoun: Factor (7 levels "I","you_s","he","she","it","we","they")

Source

Sarah Buschfeld, TU Dortmund

`FindSubstr` *Check for forbidden split results in trees*

Description

Check whether one of the character strings in the vector 'ctestv' appears as a split result in the conditional inference tree 'ct'; ctestv is a vector of character strings of forbidden split results.
Example: ctestv <- rbind('variable1 == {value1, value2}','variable2 <= value3'), where character strings specified in 'value1', 'value2' are not allowed as results of a splitting operation in 'variable 1' in a tree. For restrictions of the type 'variable <= xxx', all split results in a tree are excluded with 'variable <= yyy' and yyy <= xxx.
Trees with split results specified in 'ctestv' are not accepted during optimization.
A concrete example is: 'ctestv <- rbind('ETH == {C2a, C1a}','AGE <= 20')' for variables 'ETH' and 'AGE' and values 'C2a','C1a', and '20';
For an application, please refer to, e.g., the functions `PrInDT` and `PrInDTreg`.
If no restrictions exist, the default = NA is used.

Usage

```
FindSubstr(ct, ctestv)
```

Arguments

`ct`	Tree to be checked
`ctestv`	Vector with character strings of excluded split results

Value

testt TRUE if any of the split results in 'ctestv' appears in 'ct'; FALSE otherwise

`NesPrInDT` *Nested `PrInDT` with additional undersampling of a factor with two unbalanced levels*

Description

Function for additional undersampling of the factor 'nesvar' with two unbalanced levels to avoid dominance of the level with higher frequency. The factor 'nesvar' is allowed not be part of the input data frame 'datain'. The data of this factor is given in the vector 'nesunder'. The observations in 'nesunder' have to represent the same cases as in 'datain' in the same ordering.
`PrInDT` is called 'repin' times with subsamples of the original data so that the level with the larger frequency in the vector 'nesunder' has approximately the same number of values as the level with the smaller frequency.
Only the arguments 'nesvar', 'nesunder', and 'repin' relate to the additional undersampling, all the other arguments relate to the standard `PrInDT` procedure.

As in `PrInDT`, the aim is to optimally model the relationship between the two-class factor variable ’classname’ and all other factor and numerical variables in the data frame ’datain’ by means of ’N’ repetitions of undersampling. The trees generated by `PrInDT` can be restricted by excluding unacceptable trees which include split results specified in the character strings of the vector ’ctestv’. The probability threshold ’thres’ for the prediction of the smaller class may be specified (default = 0.5).
Undersampling may be stratified in two ways by the feature ’strat’.
The results are evaluated on the full sample and on the subsamples of ’nesunder’.

Reference
Weihs, C., Buschfeld, S. 2021b. NesPrInDT: Nested undersampling in PrInDT. arXiv:2103.14931

Usage

```
NesPrInDT(datain, classname, ctestv=NA, N, plarge, psmall=1.0, conf.level=0.95,
       thres=0.5, stratvers=0, strat=NA, seedl=TRUE, nesvar, nesunder, repin)
```

Arguments

`datain`	Input data frame with class factor variable ’classname’ and the influential variables, which need to be factors or numericals (transform logicals and character variables to factors)
`classname`	Name of class variable (character)
`ctestv`	Vector of character strings of forbidden split results; see function `PrInDT` for details. If no restrictions exist, the default = NA is used.
`N`	Number of repetitions (integer > 0)
`plarge`	Undersampling percentage of larger class (numerical, > 0 and <= 1)
`psmall`	Undersampling percentage of smaller class (numerical, > 0 and <= 1); default = 1
`conf.level`	(1 - significance level) in function `ctree` (numerical, > 0 and <= 1); default = 0.95
`thres`	Probability threshold for prediction of smaller class; default = 0.5
`stratvers`	Version of stratification; = 0: none (default), = 1: stratification according to the percentages of the values of the factor variable ’strat’, > 1: stratification with minimum number ’stratvers’ of observations per value of ’strat’
`strat`	Name of one (!) stratification variable for undersampling (character); default = NA (no stratification)
`seedl`	Should the seed for random numbers be set (TRUE / FALSE)? default = TRUE
`nesvar`	Name of factor to be undersampled (character)
`nesunder`	Data of factor to be undersampled (integer)
`repin`	Number of repetitions (integer) for undersampling of ’nesvar’

Details

Standard output can be produced by means of `print(name)` or just `name` as well as `plot(name)` where 'name' is the output data frame of the function.
The plot function will produce a series of more than one plot. If you use R, you might want to specify `windows(record=TRUE)` before `plot(name)` to save the whole series of plots. In R-Studio this functionality is provided automatically.

Value

undba balanced accuracies on undersamples

imax indices of best trees on undersamples

undba3en balanced accuracies of ensembles of 3 best trees on undersamples

accF balanced accuracies on full sample

accE balanced accuracy on full sample of best ensemble of 3 trees from undersampling

maxt indices of best trees on full sample

treesb 3 best trees of all undersamples of 'nesunder'; refer to an individual tree as `treesb[[k]]`, k = 1, ..., 3*repin

Examples

```
# data input and preparation --> data frame with
#   class variable, factors, and numericals (no character variables)!!
data <- PrInDT::data_speaker
data <- na.omit(data)
nesvar <- "SPEAKER"
N <- 49  # no. of repetitions in inner loop
plarge <- 0.06 # sampling percentage for larger class in nesunder-subsample
psmall <- 1 # sampling percentage for smaller class in nesunder-subsample
nesunder <- data$SPEAKER
data[,nesvar] <- list(NULL)
outNes <- NesPrInDT(data,"class",ctestv=NA,N,plarge,psmall,conf.level=0.95,nesvar=nesvar,
  nesunder=nesunder,repin=5)
outNes
plot(outNes)
hist(outNes$undba,main=" ",xlab = "balanced accuracies of 3 best trees of all undersamples")
```

`PostPrInDT` *Posterior analysis of conditional inference trees: distribution of a specified variable in the terminal nodes.*

Description

The conditional inference tree 'ct' is analyzed according to the distribution of a variable 'var' in its terminal nodes.
In the case of a discrete variable 'var', the appearance of the different levels is considered for each terminal node.
In the case of a continuous variable 'var', means and standard deviations of 'var' or the target variable are considered for each terminal node.
In particular, this function can be used for the posterior analysis of a tree regarding the distribution of a variable not present in the tree.

Usage

```
PostPrInDT(datain, ct, target, var, vardata, vt)
```

Arguments

`datain`	input data frame with the observatios of all variables used in the model
`ct`	conditional inference tree to be analyzed
`target`	name of target variable of 'ct' (character)
`var`	name of variable of interest (character)
`vardata`	observations of 'var'
`vt`	type of variables: 'dd' for discrete target (classification) and discrete variable 'var', 'dc' for discrete target (classification) and continuous 'var', 'cd' for continuous target (regression) and discrete 'var', and 'cc' for continuous target (regression) and continuous 'var'.

Value

None: Relevant output is produced by the function.

Examples

```
data <- PrInDT::data_zero
data <- na.omit(data)
outAll <- PrInDTAll(data,"real")
PostPrInDT(data,outAll$treeAll,"real","ETH",data$ETH,vt="dd")
PostPrInDT(data,outAll$treeAll,"real","AGE",data$AGE,vt="dc")
datareg <- PrInDT::data_vowel
outregAll <- PrInDTregAll(datareg,"target")
PostPrInDT(datareg,outregAll$treeAll,"target","Nickname",datareg$Nickname,vt="cd")
PostPrInDT(datareg,outregAll$treeAll,"target","AGE",datareg$AGE,vt="cc")
```

`PrInDT` *The basic undersampling loop for classification*

Description

The function PrInDT uses ctrees (conditional inference trees from the package "party") for optimal modeling of the relationship between the two-class factor variable 'classname' and all other factor and numerical variables in the data frame 'datain' by means of 'N' repetitions of undersampling. The optimization citerion is the balanced accuracy on the full sample. The trees generated from undersampling can be restricted by not accepting trees including split results specified in the character strings of the vector 'ctestv'.
The undersampling percentages are 'percl' for the larger class and 'percs' for the smaller class (default = 1).
The probability threshold 'thres' for the prediction of the smaller class may be specified (default = 0.5).
Undersampling may be stratified in two ways by the feature 'strat'.

Usage

```
PrInDT(datain, classname, ctestv=NA, N, percl, percs=1, conf.level=0.95, thres=0.5,
       stratvers=0, strat=NA, seedl=TRUE)
```

Arguments

`datain`	Input data frame with class factor variable 'classname' and the influential variables, which need to be factors or numericals (transform logicals and character variables to factors)
`classname`	Name of class variable (character)
`ctestv`	Vector of character strings of forbidden split results; Example: ctestv <- rbind('variable1 == {value1, value2}','variable2 <= value3'), where character strings specified in 'value1', 'value2' are not allowed as results of a splitting operation in variable 1 in a tree. For restrictions of the type 'variable <= xxx', all split results in a tree are excluded with 'variable <= yyy' and yyy <= xxx. Trees with split results specified in 'ctestv' are not accepted during optimization. A concrete example is: 'ctestv <- rbind('ETH == {C2a, C1a}','AGE <= 20')' for variables 'ETH' and 'AGE' and values 'C2a','C1a', and '20'; If no restrictions exist, the default = NA is used.
`N`	Number (> 2) of repetitions (integer)
`percl`	Undersampling percentage of larger class (numerical, > 0 and <= 1)
`percs`	Undersampling percentage of smaller class (numerical, > 0 and <= 1); default = 1

`conf.level`	(1 - significance level) in function `ctree` (numerical, > 0 and <= 1);
	default = 0.95

`thres`	Probability threshold for prediction of smaller class (numerical, >= 0 and < 1); default = 0.5
`stratvers`	Version of stratification; = 0: none (default), = 1: stratification according to the percentages of the values of the factor variable 'strat', > 1: stratification with minimum number "stratvers" of observations per value of "strat"
`strat`	Name of one (!) stratification variable for undersampling (character); default = NA (no stratification)
`seedl`	Should the seed for random numbers be set (TRUE / FALSE)? default = TRUE

Details

For the optimzation of the trees, we employ a method we call Sumping (Subsampling umbrella of model parameters), a variant of Bumping (Bootstrap umbrella of model parameters) (Tibshirani & Knight, 1999) which use subsampling instead of bootstrapping. The aim of the optimization is to identify conditional inference trees with maximum predictive power on the full sample under interpretability restrictions.

References

– Tibshirani, R., Knight, K. 1999. Model Search and Inference By Bootstrap "bumping". Journal of Computational and Graphical Statistics, Vol. 8, No. 4 (Dec., 1999), pp. 671-686
– Weihs, C., Buschfeld, S. 2021a. Combining Prediction and Interpretation in Decision Trees (PrInDT) - a Linguistic Example. arXiv:2103.02336

Standard output can be produced by means of `print(name)` or just `name` as well as `plot(name)` where 'name' is the output data frame of the function.
The plot function will produce a series of more than one plot. If you use R, you might want to specify `windows(record=TRUE)` before `plot(name)` to save the whole series of plots. In R-Studio this functionality is provided automatically.

Value

tree1st best tree on full sample

tree2nd 2nd-best tree on full sample

tree3rd 3rd-best tree on full sample

treet1st best tree on test sample

treet2nd 2nd-best tree on test sample

treet3rd 3rd-best tree on test sample

ba1st accuracies: largeClass, smallClass, balanced of 'tree1st', both for full and test sample

ba2nd accuracies: largeClass, smallClass, balanced of 'tree2nd', both for full and test sample

ba3rd accuracies: largeClass, smallClass, balanced of 'tree3rd', both for full and test sample

baen accuracies: largeClass, smallClass, balanced of ensemble of all interpretable, 3 best acceptable, and all acceptable trees on full sample

bafull vector of balanced accuracies of all trees from undersampling

batest vector of test accuracies of all trees from undersampling

dataout transformed data set 'datain' for further analyses

treeAll tree based on all observations

baAll balanced accuracy of 'treeAll'

interpAll criterion of interpretability of 'treeall' (TRUE / FALSE)

confAll confusion matrix of 'treeAll'

Examples

```
datastrat <- PrInDT::data_zero
data <- na.omit(datastrat) # cleaned full data: no NAs
# interpretation restrictions (split exclusions)
ctestv <- rbind('ETH == {C2a, C1a}','MLU == {1, 3}') # split exclusions
N <- 41  # no. of repetitions
conf.level <- 0.99 # 1 - significance level (mincriterion) in ctree
percl <- 0.08  # undersampling percentage of the larger class
percs <- 0.95 # undersampling percentage of the smaller class
# calls of PrInDT
out <- PrInDT(data,"real",ctestv,N,percl,percs,conf.level) # unstratified
out # print best model and ensembles as well as all observations
plot(out)
out <- PrInDT(data,"real",ctestv,N,percl,percs,conf.level,stratvers=1,
              strat="SEX") # percentage stratification
out <- PrInDT(data,"real",ctestv,N,percl,percs,conf.level,stratvers=50,
              strat="SEX") # stratification with minimum no. of tokens
out <- PrInDT(data,"real",ctestv,N,percl,percs,conf.level,thres=0.4) # threshold = 0.4
```

`PrInDTAll` *Conditional inference tree (ctree) based on all observations*

Description

ctree based on all observations. Interpretability is checked (see 'ctestv'); probability threshold can be specified.

Reference: Weihs, C., Buschfeld, S. 2021a. Combining Prediction and Interpretation in Decision Trees (PrInDT) - a Linguistic Example. arXiv:2103.02336

Usage

```
PrInDTAll(datain, classname, ctestv=NA, conf.level=0.95, thres=0.5)
```

Arguments

`datain`	Input data frame with class factor variable 'classname' and the influential variables, which need to be factors or numericals (transform logicals and character variables to factors)
`classname`	Name of class variable (character)
`ctestv`	Vector of character strings of forbidden split results; see function `PrInDT` for details. If no restrictions exist, the default = NA is used.
`conf.level`	(1 - significance level) in function `ctree` (numerical, > 0 and <= 1); default = 0.95
`thres`	Probability threshold for prediction of smaller class (numerical, >= 0 and < 1); default = 0.5

Details

Standard output can be produced by means of `print(name)` or just `name` as well as `plot(name)` where 'name' is the output data frame of the function.

Value

treeall ctree based on all observations

baAll balanced accuracy of 'treeall'

interpAll criterion of interpretability of 'treeall' (TRUE / FALSE)

confAll confusion matrix of 'treeall'

Examples

```
datastrat <- PrInDT::data_zero
data <- na.omit(datastrat)
ctestv <- rbind('ETH == {C2a,C1a}','MLU == {1, 3}')
conf.level <- 0.99 # 1 - significance level (mincriterion) in ctree
outAll <- PrInDTAll(data,"real",ctestv,conf.level)
print(outAll) # print model based on all observations
plot(outAll) # plot model based on all observations
```

`PrInDTAllparts` *Conditional inference trees (ctrees) based on consecutive parts of the full sample*

Description

ctrees based on the full sample of the smaller class and consecutive parts of the larger class of the nesting variable 'nesvar'. The variable 'nesvar' has to be part of the data frame 'datain'.
Interpretability is checked (see 'ctestv'); probability threshold can be specified.

Reference
Weihs, C., Buschfeld, S. 2021b. NesPrInDT: Nested undersampling in PrInDT. arXiv:2103.14931

Usage

```
PrInDTAllparts(datain, classname, ctestv=NA, conf.level=0.95, thres=0.5,
       nesvar, divt)
```

Arguments

`datain`	Input data frame with class factor variable 'classname' and the influential variables, which need to be factors or numericals (transform logicals and character variables to factors)
`classname`	Name of class variable (character)
`ctestv`	Vector of character strings of forbidden split results; see function `PrInDT` for details. If no restrictions exist, the default = NA is used.
`conf.level`	(1 - significance level) in function `ctree` (numerical, > 0 and <= 1); default = 0.95
`thres`	Probability threshold for prediction of smaller class (numerical, >= 0 and < 1); default = 0.5
`nesvar`	Name of nesting variable (character)
`divt`	Number of parts of nesting variable nesvar for which models should be determined individually

Details

Standard output can be produced by means of `print(name)` or just `name` where 'name' is the output data frame of the function.

Value

baAll balanced accuracy of tree on full sample

nesvar name of nesting variable

divt number of consecutive parts of the sample

badiv balanced accuracy of trees on 'divt' consecutive parts of the sample

Examples

```
data <- PrInDT::data_speaker
data <- na.omit(data)
nesvar <- "SPEAKER"
outNesAll <- PrInDTAllparts(data,"class",ctestv=NA,conf.level=0.95,thres=0.5,nesvar,divt=8)
outNesAll
```

`PrInDTMulab` *Multiple label classification based on resampling by* `PrInDT`

Description

Multiple label classification based on resampling by `PrInDT`. We consider two ways of modeling (Binary relevance modeling, dependent binary modeling) and three ways of model evaluation: single assessment, joint assessment, and true prediction (see the Value section for more information).
Variables should be arranged in 'datain' according to indices specified in 'indind', 'indaddind', and 'inddep'.
Undersampling is repeated 'N' times.
Undersampling percentages 'percl' for the larger class and 'percs' for the smaller class can be specified, one each per dependent class variable.

Reference
Probst, P., Au, Q., Casalicchio, G., Stachl, C., and Bischl, B. 2017. Multilabel Classification with R Package mlr. arXiv:1703.08991v2

Usage

```
PrInDTMulab(datain, classnames, ctestv, conf.level=0.95, percl, percs=1,
       N, indind, indaddind, inddep)
```

Arguments

`datain`	Input data frame with class factor variable 'classname' and the influential variables, which need to be factors or numericals (transform logicals and character variables to factors)
`classnames`	names of class variables (character vector)
`ctestv`	Vector of character strings of forbidden split results; see function `PrInDT` for details. If no restrictions exist, the default = NA is used.
`conf.level`	(1 - significance level) in function `ctree` (numerical, > 0 and <= 1); default = 0.95
`percl`	list of undersampling percentages of larger class (numerical, > 0 and <= 1): one per dependent class variable
`percs`	list of undersampling percentage of smaller class (numerical, > 0 and <= 1); one per dependent class variable

`N`	no. of repetitions (integer > 0)
`indind`	indices of independent variables
`indaddind`	indices of additional independent variables used in the case of dependent binary relevance modeling
`inddep`	indices of dependent variables

Details

Standard output can be produced by means of `print(name)` or just `name` as well as `plot(name)` where 'name' is the output data frame of the function.
The plot function will produce a series of more than one plot. If you use R, you might want to specify `windows(record=TRUE)` before `plot(name)` to save the whole series of plots. In R-Studio this functionality is provided automatically.

Value

accbr model errors for Binary Relevance (single assessment) - only independent predictors are used for modeling one label at a time, the other labels are not used as predictors. As the performance measure for the resulting classification rules, the balanced accuracy of the best model from PrInDT is employed for each individual label.

errbin combined error for Binary Relevance (joint assessment) - the best prediction models for the different labels are combined to assess the combined prediction. The 01-accuracy counts a label combination as correct only if all labels are correctly predicted. The hamming accuracy corresponds to the proportion of labels whose value is correctly predicted.

accdbr model errors for Dependent Binary Relevance (Extended Model) (single assessment) - each label is trained by means of an extended model which not only includes the independent predictors but also the other labels. For these labels, the truly observed values are used for estimation and prediction. In the extended model, other labels, which are not treated as dependent variables, can also be used as additional predictors.

errext combined errors for Dependent Binary Relevance (Extended Model) (joint assessment)

errtrue combined errors for Dependent Binary Relevance (True Prediction) - in the prediction phase, the values of all modeled labels are first predicted by the independent predictors only and then the predicted labels are used in the estimated extended model in a 2nd step to ultimately predict the labels.

coldata column names of input data

inddep indices of dependent variables (labels to be modeled)

treebr list of trees from Binary Relevance modeling, one tree for each label; refer to an individual tree as `treebr[[i]]`, i = 1, ..., no. of labels

treedbr list of trees from Dependent Binary Relevance modeling, one for each label; refer to an individual tree as `treedbr[[i]]`, i = 1, ..., no. of labels

Examples

```
data <- PrInDT::data_land # load data
dataclean <- data[,c(1:7,23:24,11:13,22,8:10)]  # only relevant features
indind <- c(1:9) # original predictors
```

```
indaddind <- c(10:13) # additional predictors
inddep <- c(14:16) # dependent variables
dataclean <- na.omit(dataclean)
ctestv <- NA
N <- 21  # no. of repetitions
perc <- c(0.45,0.05,0.25)   # percentages of observations of larger class,
#                             1 per dependent class variable
perc2 <- c(0.75,0.95,0.75)  # percentages of observations of smaller class,
#                             1 per dependent class variable
##
# Call PrInDT: language by language
##
outmult <- PrInDTMulab(dataclean,colnames(dataclean)[inddep],ctestv=NA,conf.level=0.95,
                  percl=perc,percs=perc2,N,indind,indaddind,inddep)
print(outmult)
plot(outmult)
```

PrInDTMulabAll *Multiple label classification based on all observations*

Description

Multiple label classification based on all observations. We consider two ways of modeling (Binary relevance modeling, dependent binary modeling) and three ways of model evaluation: single assessment, joint assessment, and true prediction (see the Value section for more information).
Interpretability is checked (see ctestv).
Variables should be arranged in 'datain' according to indices specified in 'indind', 'indaddind', and 'inddep'.

Reference
Probst, P., Au, Q., Casalicchio, G., Stachl, C., and Bischl, B. 2017. Multilabel Classification with R Package mlr. arXiv:1703.08991v2

Usage

```
PrInDTMulabAll(datain, classnames, ctestv=NA, conf.level=0.95, indind, indaddind,
        inddep)
```

Arguments

`datain`	Input data frame with class factor variable 'classname' and the influential variables, which need to be factors or numericals (transform logicals and character variables to factors)
`classnames`	names of class variables (character vector)
`ctestv`	Vector of character strings of forbidden split results; see function PrInDT for details. If no restrictions exist, the default = NA is used.

`conf.level`	(1 - significance level) in function `ctree` (numerical, > 0 and <= 1); default = 0.95
`indind`	indices of independent variables
`indaddind`	indices of additional predictors used in the case of dependent binary relevance modeling
`inddep`	indices of dependent variables

Details

Standard output can be produced by means of `print(name)` or just `name` as well as `plot(name)` where 'name' is the output data frame of the function.
The plot function will produce a series of more than one plot. If you use R, you might want to specify `windows(record=TRUE)` before `plot(name)` to save the whole series of plots. In R-Studio this functionality is provided automatically.

Value

accabr model errors for Binary Relevance (single assessment) - only independent predictors are used for modeling one label at a time, the other labels are not used as predictors. The classification rules are trained on all observations. As the performance measure for the resulting classification rules, the balanced accuracy of the models for each individual label is employed.

errabin combined error for Binary Relevance (joint assessment) - the best prediction models for the different labels are combined to assess the combined prediction. The 01-accuracy counts a label combination as correct only if all labels are correctly predicted. The hamming accuracy corresponds to the proportion of labels whose value is correctly predicted.

accadbr model errors in Dependent Binary Relevance (Extended Model) (single assessment) - each label is trained by means of an extended model which not only includes the independent predictors but also the other labels. For these labels the truly observed values are used for estimation and prediction. In the extended model, further labels, which are not treated as dependent variables, can be used as additional predictors.

erraext combined errors for Dependent Binary Relevance (Extended Model) (joint assessment)

erratrue combined errors for Dependent Binary Relevance (True Prediction) - in the prediction phase, the values of all modeled labels are first predicted by the independent predictors only (see Binary Relevance) and then the predicted labels are used in the estimated extended model in a 2nd step to ultimately predict the labels.

coldata column names of input data

inddep indices of dependent variables (labels to be modeled)

treeabr list of trees from Binary Relevance modeling, one tree for each label; refer to an individual tree as `treeabr[[i]]`, i = 1, ..., no. of labels

treeadbr list of trees from Dependent Binary Relevance modeling, one for each label; refer to an individual tree as `treeadbr[[i]]`, i = 1, ..., no. of labels

Examples

```
data <- PrInDT::data_land # load data
dataclean <- data[,c(1:7,23:24,11:13,22,8:10)]  # only relevant features
```

```
indind <- c(1:9) # original predictors
indaddind <- c(10:13) # additional predictors
inddep <- c(14:16) # dependent variables
dataclean <- na.omit(dataclean)
ctestv <- NA
##
# Call PrInDTAll: language by language
##
outmultAll <- PrInDTMulabAll(dataclean,colnames(dataclean)[inddep],ctestv,conf.level=0.95,
                             indind,indaddind,inddep)
outmultAll
plot(outmultAll)
```

`PrInDTMulev` *PrInDT analysis for a classification problem with multiple classes.*

Description

PrInDT analysis for a classification problem with more than 2 classes. For each combination of one class vs. the other classes a 2-class `PrInDT` analysis is carried out.
The percentages for undersampling of the larger class ('percl' in `PrInDT`) are chosen so that the resulting sizes are comparable with the size of the smaller classes for which all their observations are used in undersampling ('percs' = 1 in `PrInDT`).
The class with the highest probability in the K (= number of classes) analyses is chosen for prediction.
Interpretability is checked (see 'ctestv').

Usage

```
PrInDTMulev(datain, classname, ctestv=NA, N, conf.level=0.95)
```

Arguments

`datain`	Input data frame with class factor variable 'classname' and the influential variables, which need to be factors or numericals (transform logicals and character variables to factors)
`classname`	Name of class variable (character)
`ctestv`	Vector of character strings of forbidden split results; see function `PrInDT` for details. If no restrictions exist, the default = NA is used.
`N`	Number of repetitions (integer > 0)
`conf.level`	(1 - significance level) in function `ctree` (numerical, > 0 and <= 1) (default = 0.95)

Details

Standard output can be produced by means of `print(name)` or just `name` as well as `plot(name)` where 'name' is the output data frame of the function.
The plot function will produce a series of more than one plot. If you use R, you might want to specify `windows(record=TRUE)` before `plot(name)` to save the whole series of plots. In R-Studio this functionality is provided automatically.

Value

class levels of class variable

trees trees for the levels of the class variable; refer to an individual tree as `trees[[k]]`, k = 1, ..., no. of levels

ba balanced accuracy of combined predictions

conf confusion matrix of combined predictions

ninterp no. of non-interpretable trees

Examples

```
datastrat <- PrInDT::data_zero
data <- na.omit(datastrat)
ctestv <- NA
data$rel[data$ETH %in% c("C1a","C1b","C1c") & data$real == "zero"] <- "zero1"
data$rel[data$ETH %in% c("C2a","C2b","C2c") & data$real == "zero"] <- "zero2"
data$rel[data$real == "realized"] <- "real"
data$rel <- as.factor(data$rel) # rel is new class variable
data$real <- NULL # remove old class variable
N <- 51
conf.level <- 0.99 # 1 - significance level (mincriterion) in ctree
out <- PrInDTMulev(data,"rel",ctestv,N,conf.level)
out # print best models based on subsamples
plot(out) # corresponding plots
```

`PrInDTMulevAll` *Conditional inference tree (ctree) for multiple classes on all observations*

Description

ctree for more than 2 classes on all observations. Interpretability is checked (see 'ctestv').

Usage

```
PrInDTMulevAll(datain, classname, ctestv=NA, conf.level=0.95)
```

Arguments

`datain`	Input data frame with class factor variable 'classname' and the influential variables, which need to be factors or numericals (transform logicals and character variables to factors)
`classname`	Name of class variable (character)
`ctestv`	Vector of character strings of forbidden split results; see function `PrInDT` for details. If no restrictions exist, the default = NA is used.
`conf.level`	(1 - significance level) in function `ctree` (numerical, > 0 and <= 1) (default = 0.95)

Details

Standard output can be produced by means of `print(name)` or just `name` as well as `plot(name)` where 'name' is the output data frame of the function.

Value

treeall ctree based on all observations

baAll balanced accuracy of 'treeall'

interpAll criterion of interpretability of 'treeall' (TRUE / FALSE)

confAll confusion matrix of 'treeall'

Examples

```
datastrat <- PrInDT::data_zero
data <- na.omit(datastrat)
ctestv <- rbind('ETH == {C2a,C1a}', 'MLU == {1, 3}')
data$rel[data$ETH %in% c("C1a","C1b","C1c") & data$real == "zero"] <- "zero1"
data$rel[data$ETH %in% c("C2a","C2b","C2c") & data$real == "zero"] <- "zero2"
data$rel[data$real == "realized"] <- "real"
data$rel <- as.factor(data$rel) # rel is new class variable
data$real <- NULL # remove old class variable
conf.level <- 0.99 # 1 - significance level (mincriterion) in ctree
outAll <- PrInDTMulevAll(data,"rel",ctestv,conf.level)
outAll # print model based on all observations
plot(outAll)
```

`PrInDTreg` *Regression tree resampling by the PrInDT method*

Description

Regression tree optimzation to identify the best interpretable tree; interpretability is checked (see 'ctestv').
The relationship between the target variable 'regname' and all other factor and numerical variables in the data frame 'datain' is optimally modeled by means of 'N' repetitions of subsampling.
The optimization criterion is the R2 of the model on the full sample.
Multiple subsampling percentages of observations and predictors can be specified (in 'pobs' and 'ppre', correspondingly).
The trees generated from undersampling can be restricted by rejecting unacceptable trees which include split results specified in the character strings of the vector 'ctestv'.

Usage

```
PrInDTreg(datain, regname, ctestv=NA, N, pobs, ppre, conf.level=0.95)
```

Arguments

`datain`	Input data frame with class factor variable 'classname' and the influential variables, which need to be factors or numericals (transform logicals and character variables to factors)
`regname`	name of regressand variable (character)
`ctestv`	Vector of character strings of forbidden split results; see function `PrInDT` for details. If no restrictions exist, the default = NA is used.
`N`	Number of repetitions (integer > 0)
`pobs`	Vector of resampling percentages of observations (numerical, > 0 and <= 1)
`ppre`	Vector of resampling percentages of predictor variables (numerical, > 0 and <= 1)
`conf.level`	(1 - significance level) in function `ctree` (numerical, > 0 and <= 1); default = 0.95

Details

For the optimzation of the trees, we employ a method we call Sumping (Subsampling umbrella of model parameters), a variant of Bumping (Bootstrap umbrella of model parameters) (Tibshirani & Knight, 1999) which use subsampling instead of bootstrapping. The aim of the optimization is to identify conditional inference trees with maximum predictive power on the full sample under interpretability restrictions.

Reference
Tibshirani, R., Knight, K. 1999. Model Search and Inference By Bootstrap "bumping". Journal of Computational and Graphical Statistics, Vol. 8, No. 4 (Dec., 1999), pp. 671-686

Standard output can be produced by means of `print(name)` or just `name` as well as `plot(name)` where 'name' is the output data frame of the function.
The plot function will produce a series of more than one plot. If you use R, you might want to specify `windows(record=TRUE)` before `plot(name)` to save the whole series of plots. In R-Studio this functionality is provided automatically.

Value

meanint Mean number of interpretable trees over the combinations of individual percentages in 'pobs' and 'ppre'

R2mean Mean R2 on test sets

ctmax best resampled regression tree according to R2 on the full data set

percmax Maximum R2 achieved for %observations

perfeamax Maximum R2 achieved for %predictors

maxR2 best R2 on the full data set for resampled regression trees (for 'ctmax')

interpmax interpretability of best tree 'ctmax'

ctmax2 second best resampled regression tree according to R2 on the full data set

percmax2 second best R2 achieved for %observations

perfeamax2 second best R2 achieved for %features

max2R2 second best R2 on the full data set for resampled regression trees (for 'ctmax2')

interp2max interpretability of second-best tree 'ctmax2'

Examples

```
data <- PrInDT::data_vowel
data <- na.omit(data)
ctestv <- 'vowel_maximum_pitch <= 320'
N <- 30 # no. of repetitions
pobs <- c(0.70,0.60) # percentages of observations
ppre <- c(0.90,0.70) # percentages of predictors
outreg <- PrInDTreg(data,"target",ctestv,N,pobs,ppre)
outreg
plot(outreg)
```

`PrInDTregAll` *Regression tree based on all observations*

Description

Regression tree based on the full sample; interpretability is checked (see 'ctestv').
The relationship between the target variable 'regname' and all other factor and numerical variables in the data frame 'datain' is modeled based on all observations.

Usage

```
PrInDTregAll(datain, regname, ctestv=NA, conf.level=0.95)
```

Arguments

`datain`	Input data frame with class factor variable 'classname' and the influential variables, which need to be factors or numericals (transform logicals and character variables to factors)
`regname`	name of regressand variable (character)
`ctestv`	Vector of character strings of forbidden split results; see function `PrInDT` for details. If no restrictions exist, the default = NA is used.
`conf.level`	(1 - significance level) in function `ctree` (numerical, > 0 and <= 1); default = 0.95

Details

Standard output can be produced by means of `print(name)` or just `name` as well as `plot(name)` where 'name' is the output data frame of the function.

Value

treeall tree based on all observations

R2All goodness of fit of 'treeall' based on all observations

interpAll criterion of interpretability of 'treeall' (TRUE / FALSE)

Examples

```
data <- PrInDT::data_vowel
data <- na.omit(data)
ctestv <- 'vowel_maximum_pitch <= 320'
outreg <- PrInDTregAll(data,"target",ctestv)
outreg
plot(outreg)
```

`RePrInDT` *Repeated* `PrInDT` *for specified percentage combinations*

Description

`PrInDT` is called repeatedly according to the percentages specified in the vectors 'plarge' and 'psmall'.
The relationship between the two-class factor variable 'classname' and all other factor and numerical variables in the data frame 'datain' is optimally modeled by means of 'N' repetitions of undersampling.
The trees generated from undersampling can be restricted by rejecting unacceptable trees which include split results specified in the character strings of the vector 'ctestv'.
The probability threshold 'thres' for the prediction of the smaller class may be specified (default =

0.5).
Undersampling may be stratified in two ways by the feature 'strat'.

Reference
Weihs, C., Buschfeld, S. 2021c. Repeated undersampling in PrInDT (RePrInDT): Variation in undersampling and prediction, and ranking of predictors in ensembles. arXiv:2108.05129

Usage

```
RePrInDT(datain, classname, ctestv=NA, N, plarge, psmall, conf.level=0.95,
       thres=0.5, stratvers=0, strat=NA, seedl=TRUE)
```

Arguments

`datain`	Input data frame with class factor variable 'classname' and the influential variables, which need to be factors or numericals (transform logicals and character variables to factors)
`classname`	Name of class variable (character)
`ctestv`	Vector of character strings of forbidden split results; see function `PrInDT` for details. If no restrictions exist, the default = NA is used.
`N`	Number of repetitions (integer > 0)
`plarge`	Vector of undersampling percentages of larger class (numerical, > 0 and <= 1)
`psmall`	Vector of undersampling percentages of smaller class (numerical, > 0 and <= 1)

`conf.level`	(1 - significance level) in function `ctree` (numerical, > 0 and <= 1);
	default = 0.95

`thres`	Probability threshold for prediction of smaller class (numerical, >= 0 and < 1); default = 0.5
`stratvers`	Version of stratification; = 0: none (default), = 1: stratification according to the percentages of the values of the factor variable 'strat', > 1: stratification with minimum number 'stratvers' of observations per value of 'strat'
`strat`	Name of one (!) stratification variable for undersampling (character); default = NA (no stratification)
`seedl`	Should the seed for random numbers be set (TRUE / FALSE)? default = TRUE

Details

Standard output can be produced by means of `print(name)` or just `name` as well as `plot(name)` where 'name' is the output data frame of the function.
The plot function will produce a series of more than one plot. If you use R, you might want to specify `windows(record=TRUE)` before `plot(name)` to save the whole series of plots. In R-Studio this functionality is provided automatically.

Value

treesb best trees for the different percentage combinations; refer to an individual tree as `treesb[[k]]`, k = 1, ..., length(plarge)*length(psmall)

acc1st accuracies of best trees on full sample

acc3en accuracies of ensemble of 3 best trees on full sample

simp_m mean of permutation losses for the predictors

Examples

```
datastrat <- PrInDT::data_zero
data <- na.omit(datastrat) # cleaned full data: no NAs
# interpretation restrictions (split exclusions)
ctestv <- rbind('ETH == {C2a, C1a}', 'MLU == {1, 3}')
N <- 51  # no. of repetitions
conf.level <- 0.99 # 1 - significance level (mincriterion) in ctree
psmall <- c(0.95,1)     # percentages of the small class
plarge <- c(0.09,0.1)  # percentages of the large class
outRe <- RePrInDT(data,"real",ctestv,N,plarge,psmall,conf.level) # might take 5 minutes
outRe
plot(outRe)
```

Index