Package 'PRIM'

May 30, 2017

Type Package

Title Patient Rule Induction Method (PRIM)

Version 1.0

Date 2017-05-23

Author Armin Ott [aut, cre]

Maintainer Armin Ott <armin-ott@freenet.de>

Imports survival

Description This Package implements the Patient Rule Induction Method as sugessted by Friedman and Fisher (1999).

License GPL (>= 3)

LazyData TRUE

RoxygenNote 5.0.1

NeedsCompilation no

R topics documented:

define_fixbox *Defining a "fixbox"-Object*

Description

Function for determining on one box out of a "peel"-object.

Usage

```
define_fixbox(prim, step)
```

Arguments

prim	object of class "peel" from which the box should be defined.
step	number of the step (i.e. row in the argument box of the "peel"-object.) of the requested box.

Value

define_fixbox returns a object of class "fixbox", which is the basically same as a "peel"-object, but only contains one box. It is a list containing at least the following elements:

f	target function of the selected box.
beta	support of the selected box.
box	a data.frame with one row defining the borders of the box. The columns with "min." and "max." describe the lower and upper boundaries of the at least ordinal covariates. Therefore the value taken is the last one that is **not** included in the box. For the nominal variables there are columns for every category they can take. If the category is removed from the box the value FALSE is taken. The names of these columns are structured like: <variable name>.<category> For each variable with missing values (only if use_NAs = TRUE) there is also a column taking the value FALSE if the NAs of this variable are removed from the box. The names of these columns are structured like: <variable name>.NA
box_metric, box_nom, box_na	easier to handle definitions of the box for other functions
subset	logical vector indicating the subset (i.e. the observations that lie in the box).
data_orig	original dataset that was used for the peeling.

inbox *Defining Subset lying in a Box*

Description

Help function that returns the subset of the whole data set which is contained in one box (`fixbox`). This function is needed in other functions like `PRIM_peel_bs`.

Usage

```
inbox(X, fixbox_metric = NULL, fixbox_nom = NULL, fixbox_na = NULL)
```

Arguments

`X`	`data.frame` from which the subset should be taken.
`fixbox_metric`	one row of a `box_metric` from a "`peel`"-object.
`fixbox_nom`	one element of a `box_nom` list from a "`peel`"-object.
`fixbox_na`	one element of a `box_na` list from a "`peel`"-object.

Details

The arguments `fixbox_metric`, `fixbox_nom`, `fixbox_na` can also come from a "`fixbox`"-object.

Value

`inbox` returns a logical vector indicating the observations that lie in the box.

inter_diss *Interbox Dissimilarity*

Description

This Function calculates the interbox dissimilarity between two boxes defined on one data set.

Usage

```
inter_diss(fixbox1, fixbox2, data)
```

Arguments

`fixbox1`	first object of class "`fixbox`".
`fixbox2`	second object of class "`fixbox`".
`data`	original data used to calculate the supports needed for the interbox dissimilarity. If this argument is missing `data_orig` from `fixbox1` is used as `data`.

Details

The interbox dissimilarity is a diagnostic tool of PRIM measuring the dissimilarity between two boxes B_k and B_l. It is defined as the difference between the smallest box B_{kl} that covers both boxes and the support of their union.

The interbox dissimilarity can assume values between 0 and 1. While nested boxes have a dissimilarity of 0, it gets bigger the more different the two boxes are.

---

plot.peel *Plotting a Trajectory*

Description

This funcion plots the trajectory of a "peel"-object. In the trajectory the target functions evaluated on the data in the box are plotted against the supports beta.

Usage

```
## S3 method for class 'peel'
plot(x, ...)
```

Arguments

x	object of class "peel".
...	further arguments of the plot-function.

PRIM *Combined Function for the Patient Rule Induction Method (PRIM)*

Description

This function is a automated implementation of PRIM as suggested by Friedman and Fisher (1999). It includes multiple peeling (PRIM_peel_bs), pasting (PRIM_paste) and the covering stretegy to find more than one box.

Usage

```
PRIM(formula, data, f_min, beta_min = 0.2, max_boxes = Inf,
  peel_alpha = seq(0.01, 0.4, 0.03), B = 0, target = mean,
  alter_crit = TRUE, use_NAs = TRUE, seed, print_position = FALSE,
  paste_alpha = 0.01, max_steps = 50, stop_by_dec = TRUE)
```

Arguments

formula	an object of class "formula" with a response but no interaction terms. It indicates the response over which the target function should be maximized and the covariates that are used for the later box definitions.
data	an object of class data.frame containing the variables named in the formula.
f_min	minimum target the final box must have. From all boxes, that fulfill this criterion, the one with the biggest support is taken after the peeling. If this argument is missing the box with the biggest target having at least a support of beta_min is taken.
beta_min	minimum support that one box must have. This proportion always refers to the whole data set.
max_boxes	maximum number of boxes to be found.
peel_alpha	vector of a sequence of different alpha-fractions used for the peelings.
B	number of bootstrap samples on which the peeling is applied to for each alpha. For B = 0 no bootstraps are created.
target	target-function to be maximized. In most cases the mean is a useful target, although other functions like e.g. the median are also possible here.
alter_crit	logical. If TRUE the alternative criterion is used for peeling. I.e. "target/beta" is maximized during peeling instead of "target", so that large subboxes are not prefered to be peeled off. This is important especially in case of nominal covariates.
use_NAs	logical. If TRUE observations with missing values are included in the analysis.
seed	seed to be set before the first iteration. Only useful for B > 0.
print_position	logical. If TRUE the current position of the algorithm is printed out.
paste_alpha	alpha-fraction that is pasted to the box at each pasting step
max_steps	maximum number of pasting steps the function should make.
stop_by_dec	logical. If TRUE the pasting stops if the target at one step is lower than the target of the last step.

Details

This function repeats the peeling and pasting algorithm for the same settings of the metaparameters until a stop ctiterion is reached. After each iteration the observations already included in a box are removed from the data, on which the next box is built. This strategy is called covering. This iteration stops if either max_boxes is reached or if the target function of the "best" box is lower than the overall target.

In each iteration step this function does a multiple peeling characterized by the sequenz alpha_peel and B. From the peeling output the box defined by beta_min and f_min is chosen. After that the pasting function seeks for boxes with bigger supports and bigger targets and takes the one with the highest target function within the box.

The function can also cope with survival outcomes (Surv-object). Therefore the hazard rate is used as target function as suggested in Ott and Hapfelmeier (2017). The value of the input parameter target is ignored in this case.

Value

PRIM returns an object of class "prim", which is a list containing the following components:

f	vector of the target functions evaluated on each box. The last element is the target of all observations not lying in a box.
beta	vector of the supports of each box. The last element is the fraction of observations not lying in a box.
box	a data.frame defining the borders of the boxes. Each row belongs to one box. The columns with "min." and "max." describe the lower and upper boundaries of the at least ordinal covariates. Therefore the value taken is the last one that is **not** included in the current box. For the nominal variables there are columns for every category they can take. If the category is removed from the box the value FALSE is taken. The names of these columns are structured like: <variable name>.<category> For each variable with missing values (only if use_NAs = TRUE) there is also a column taking the value FALSE if the NAs of this variable are removed from the current box. The names of these columns are structured like: <variable name>.NA
box_metric, box_nom, box_na	easier to handle definitions of the boxes for other functions
subsets	list of logical vectors indicating the subsets (i.e. the observations that lie in each box)
fixboxes	list of all fixbox'es defining the final boxes.
data_orig	original dataset that is used.

References

Friedman, J. H. and Fisher, N. I., 'Bump hunting in high-dimensional data', Statistics and Computing 9 (2) (1999), 123-143

Ott, A. and Hapfelmeier, A., 'Nonparametric Subgroup Identification by PRIM and CART: A Simulation and Application Study', Computational and Mathematical Methods in Medicine, vol. 2017 (2017), 17 pages, Article ID 5271091

See Also

PRIM_peel_bs, PRIM_paste, define_fixbox

Examples

```
# generating random data:
set.seed(123)
n <- 500
x1 <- runif(n = n, min = -1)
x2 <- runif(n = n, min = -1)
x3 <- runif(n = n, min = -1)
cat <- as.factor(sample(c("a","b","c", "d"), size = n, replace = TRUE))
wsk <- (1-sqrt(x1^2+x2^2)/sqrt(2))
y <- as.logical(rbinom(n = n, prob = wsk, size = 1))
dat <- cbind.data.frame(y, x1, x2, x3, cat)
#plot(dat$x1, dat$x2, col=dat$y+1, pch=16)
remove(x1, x2, x3, y, wsk, cat, n)

# apply the PRIM function to find the best boxes with a support of at least 0.1:
```

```
p <- PRIM(y~., data=dat, beta_min = 0.1, max_boxes = 3)
p
```

PRIM_paste *Pasting-Function*

Description

This function is an implementation of the Pasting-Algorithm as suggested by Friedman and Fisher (1999). In each iteration the fraction alpha is pasted to one edge of the current box.

Usage

```
PRIM_paste(fixbox, paste_alpha = 0.01, max_steps = 50, stop_by_dec = TRUE)
```

Arguments

fixbox	an object of class fixbox, which was defined after the peeling function and now should be used for pasting.
paste_alpha	alpha-fraction that is pasted to the box at each iteration.
max_steps	maximum number of pasting steps the function should make.
stop_by_dec	logical. If TRUE the pasting stops if the target at one step is lower than the target of the last step.

Details

The outcome of this function is also a "peel"-object, because it has basically the same structure as the outcome of the peeling functions. The only difference is, that pasting goes from small supports to bigger ones, while by peeling its the other way round.

Value

PRIM_paste returns an object of class "peel", which is a list containing at least the following components:

f	vector of the target functions evaluated on the box at each pasting step.
beta	vector of the supports beta of the boxes at each pasting step.
box	a data.frame defining the borders of the boxes. Each row belongs to one pasting step. The columns with "min." and "max." describe the lower and upper boundaries of the at least ordinal covariates. Therefore the value taken is the last one that is **not** included in the current box. For the nominal variables there are columns for every category they can take. If the category is removed from the box the value FALSE is taken. The names of these columns are structured like: <variable name>.<category> For each variable with missing values (only if use_NAs = TRUE) there is also a column taking the value FALSE if the NAs of this variable are removed from the current box. The names of these columns are structured like: <variable name>.NA

box_metric, box_nom, box_na
: easier to handle definitions of the boxes for other functions

subsets
: list of logical vectors indicating the subsets at each pasting step (i.e. the observations that lie in the box)

data_orig
: original dataset that is used (extracted from fixbox).

References

Friedman, J. H. and Fisher, N. I., 'Bump hunting in high-dimensional data', Statistics and Computing 9 (2) (1999), 123-143

Ott, A. and Hapfelmeier, A., 'Nonparametric Subgroup Identification by PRIM and CART: A Simulation and Application Study', Computational and Mathematical Methods in Medicine, vol. 2017 (2017), 17 pages, Article ID 5271091

Examples

```
# generating random data:
set.seed(123)
n <- 500
x1 <- runif(n = n, min = -1)
x2 <- runif(n = n, min = -1)
x3 <- runif(n = n, min = -1)
cat <- as.factor(sample(c("a","b","c", "d"), size = n, replace = TRUE))
wsk <- (1-sqrt(x1^2+x2^2)/sqrt(2))
y <- as.logical(rbinom(n = n, prob = wsk, size = 1))
dat <- cbind.data.frame(y, x1, x2, x3, cat)
#plot(dat$x1, dat$x2, col=dat$y+1, pch=16)
remove(x1, x2, x3, y, wsk, cat, n)

# apply the PRIM_peel function:
prim <- PRIM_peel(y ~ ., data = dat, beta_min = .01, peel_alpha = .1)
plot(prim)
abline(h=prim$f[17], v=prim$beta[17]) # box decided to paste
fix <- define_fixbox(prim, 17) # define fixbox

# apply the PRIM_paste function:
paste <- PRIM_paste(fix, stop_by_dec = FALSE)
head(cbind(paste$box, paste$f, paste$beta))
```

PRIM_peel *Peeling-Function*

Description

This function is an implementation of the (singular) Peeling-Algorithm as suggested by Friedman and Fisher (1999). In each iteration the fraction alpha is peeled from one edge of the current box.

Usage

```
PRIM_peel(formula, data, peel_alpha = 0.05, beta_min = 0.01,
  target = mean, alter_crit = TRUE, use_NAs = TRUE)
```

Arguments

formula	an object of class "formula" with a response but no interaction terms. It indicates the response over which the target function should be maximized and the covariates that are used for the later box definitions. If this argument is missing, the argument data is used as the model.frame.
data	an object of class data.frame containing the variables named in the formula.
peel_alpha	alpha-fraction used for the peeling.
beta_min	minimum support that one Box should have (stop-criterion).
target	target-function to be maximized. In most cases the mean is a useful target, although other functions like e.g. the median are also possible here.
alter_crit	logical. If TRUE the alternative criterion is used for peeling. I.e. "target/beta" is maximized during peeling instead of "target", so that large subboxes are not prefered to be peeled off. This is important especially in case of nominal covariates.
use_NAs	logical. If TRUE observations with missing values are included in the analysis.

Details

The outcome of the formula can either be numeric, logical or a survival object (see Surv). If it is a survival object the target is set to the number of events per amount of time.

This function is the main part of the multiple Version PRIM_peel_bs.

Value

PRIM_peel returns an object of class "peel", which is a list containing at least the following components:

f	vector of the target functions evaluated on the box at each peeling step.
beta	vector of the supports beta of the boxes at each peeling step.
box	a data.frame defining the borders of the boxes. Each row belongs to one peeling step. The columns with "min." and "max." describe the lower and upper boundaries of the at least ordinal covariates. Therefore the value taken is the last one that is **not** included in the current box. For the nominal variables there are columns for every category they can take. If the category is removed from the box the value FALSE is taken. The names of these columns are structured like: <variable name>.<category> For each variable with missing values (only if use_NAs = TRUE) there is also a column taking the value FALSE if the NAs of this variable are removed from the current box. The names of these columns are structured like: <variable name>.NA
box_metric, box_nom, box_na	easier to handle definitions of the boxes for other functions
subsets	list of logical vectors indicating the subsets at each peeling step (i.e. the observations that lie in the box)
data_orig	original dataset that is used for the peeling.

References

Friedman, J. H. and Fisher, N. I., 'Bump hunting in high-dimensional data', Statistics and Computing 9 (2) (1999), 123-143

Ott, A. and Hapfelmeier, A., 'Nonparametric Subgroup Identification by PRIM and CART: A Simulation and Application Study', Computational and Mathematical Methods in Medicine, vol. 2017 (2017), 17 pages, Article ID 5271091

Examples

```
# generating random data:
set.seed(123)
n <- 500
x1 <- runif(n = n, min = -1)
x2 <- runif(n = n, min = -1)
x3 <- runif(n = n, min = -1)
cat <- as.factor(sample(c("a","b","c", "d"), size = n, replace = TRUE))
wsk <- (1-sqrt(x1^2+x2^2)/sqrt(2))
y <- as.logical(rbinom(n = n, prob = wsk, size = 1))
dat <- cbind.data.frame(y, x1, x2, x3, cat)
#plot(dat$x1, dat$x2, col=dat$y+1, pch=16)
remove(x1, x2, x3, y, wsk, cat, n)

# apply the PRIM_peel function:
prim <- PRIM_peel(formula=y ~ ., data=dat, beta_min = .01, peel_alpha = .1)
plot(prim) # trajectory
prim$box # box definitions
```

`PRIM_peel_bs` *Multiple Peeling-Function*

Description

This function is an implementation of the multiple Peeling-Algorithm as suggested by Friedman and Fisher (1999). The singular peeling function `PRIM_peel` is repeated for different alpha's and bootstrap samples out of the original data.

Usage

```
PRIM_peel_bs(formula, data, peel_alpha = seq(0.01, 0.4, 0.03), B = 0,
  beta_min = 0.01, target = mean, alter_crit = TRUE, use_NAs = TRUE,
  seed, print_position = TRUE)
```

Arguments

`formula`	an object of class "`formula`" with a response but no interaction terms. It indicates the response over which the target function should be maximized and the covariates that are used for the later box definitions.
`data`	an object of class `data.frame` containing the variables named in the formula.
`peel_alpha`	vector of a sequence of different alpha-fractions used for the peelings.

B	number of bootstrap samples on which the peeling is applied to for each alpha. For B = 0 no bootstraps are created.
beta_min	minimum support that one Box should have (stop-criterion).
target	target-function to be maximized. In most cases the mean is a useful target, although other functions like e.g. the median are also possible here.
alter_crit	logical. If TRUE the alternative criterion is used for peeling. I.e. "target/beta" is maximized during peeling instead of "target", so that large subboxes are not prefered to be peeled off. This is important especially in case of nominal covariates.
use_NAs	logical. If TRUE observations with missing values are included in the analysis.
seed	seed to be set before the first iteration. Only useful for B > 0.
print_position	logical. If TRUE the current position of the algorithm is printed out.

Details

The outcome of the formula can either be numeric, logical or a survival object (see Surv). If it is a survival object the target is set to the number of events per amount of time.

The output of this function can become very large because all outputs of the singular peel function PRIM_peel are put together in one output. Therefore it is usefull to remove all the dominated boxes (see remove_dominated).

Value

PRIM_peel_bs returns an object of class "peel", which is a list containing at least the following components:

f	vector of the target functions evaluated on the box at each peeling step.
beta	vector of the supports beta of the boxes at each peeling step.
box	a data.frame defining the borders of the boxes. Each row belongs to one peeling step. The columns with "min." and "max." describe the lower and upper boundaries of the at least ordinal covariates. Therefore the value taken is the last one that is **not** included in the current box. For the nominal variables there are columns for every category they can take. If the category is removed from the box the value FALSE is taken. The names of these columns are structured like: <variable name>.<category> For each variable with missing values (only if use_NAs = TRUE) there is also a column taking the value FALSE if the NAs of this variable are removed from the current box. The names of these columns are structured like: <variable name>.NA
box_metric, box_nom, box_na	easier to handle definitions of the boxes for other functions
subsets	list of logical vectors indicating the subsets at each peeling step (i.e. the observations that lie in the box)
data_orig	original dataset that is used for the peeling.

References

Friedman, J. H. and Fisher, N. I., 'Bump hunting in high-dimensional data', Statistics and Computing 9 (2) (1999), 123-143

Ott, A. and Hapfelmeier, A., 'Nonparametric Subgroup Identification by PRIM and CART: A Simulation and Application Study', Computational and Mathematical Methods in Medicine, vol. 2017 (2017), 17 pages, Article ID 5271091

See Also

remove_dominated, PRIM_peel, PRIM_paste, PRIM

Examples

```
# generating random data:
set.seed(123)
n <- 500
x1 <- runif(n = n, min = -1)
x2 <- runif(n = n, min = -1)
x3 <- runif(n = n, min = -1)
cat <- as.factor(sample(c("a","b","c", "d"), size = n, replace = TRUE))
wsk <- (1-sqrt(x1^2+x2^2)/sqrt(2))
y <- as.logical(rbinom(n = n, prob = wsk, size = 1))
dat <- cbind.data.frame(y, x1, x2, x3, cat)
#plot(dat$x1, dat$x2, col=dat$y+1, pch=16)
remove(x1, x2, x3, y, wsk, cat, n)

# apply the PRIM_peel_bs function:
prim <- PRIM_peel_bs(formula=y ~ ., data=dat, beta_min = .01)
plot(prim) # multiple trajectory
head(prim$box) # box definitions
```

---

print.PRIM *Printing a "PRIM"-Object*

Description

This function prints a PRIM-object. It is a method for the generic function print of class "PRIM".

Usage

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

Arguments

x	object of class "PRIM".
...	further arguments of the print-function.

---

relfreq *Relative Frequency Plots*

Description

This Function creates the relative frequency plots of a box defined by define_fixbox.

Usage

```
relfreq(fixbox, plot_together = TRUE, ...)
```

Arguments

fixbox	object of class "fixbox" to be used to create the relative frequency plots.
plot_together	logical. If TRUE all plots are drawn together in one figure by par(mfrow = ...).
...	further arguments of the functions plot and barplot.

Details

Relative frequency plots are a diagnostic tool for a result of PRIM (fixbox). These are histograms or barplots illustrating ratios of distributions r_{jk} for each variable x_j. This ratio is defined by the distribution of x_j in one box B_k divided by the distribution of this variable over the whole data set.

See Also

define_fixbox, PRIM

Examples

```
# generating random data:
set.seed(123)
n <- 500
x1 <- runif(n = n, min = -1)
x2 <- runif(n = n, min = -1)
x3 <- runif(n = n, min = -1)
cat <- as.factor(sample(c("a","b","c", "d"), size = n, replace = TRUE))
wsk <- (1-sqrt(x1^2+x2^2)/sqrt(2))
y <- as.logical(rbinom(n = n, prob = wsk, size = 1))
dat <- cbind.data.frame(y, x1, x2, x3, cat)
plot(dat$x1, dat$x2, col=dat$y+1, pch=16)
remove(x1, x2, x3, y, wsk, cat, n)

# apply the PRIM function to find the best box with a support of at least 0.1:
p <- PRIM(y~., data=dat, beta_min = 0.1, max_boxes = 1, print_position = FALSE)

# relative frequency plots:
relfreq(p$fixboxes[[1]])
```

---

remove_dominated *Remove Dominated Boxes from a Peeling Output*

Description

This function removes all boxes from one "peel"-object that are dominated by another box. I.e. there is at least one other box with a better output and a bigger support beta.

Usage

```
remove_dominated(prim, sort = TRUE, dup.rm = TRUE)
```

Arguments

prim	object of class "peel".
sort	logical. If TRUE the boxes in the output are sorted decreasing by their box supports.
dup.rm	logical. Tf TRUE duplicated boxes in the output are removed. Boxes are duplicated if another box has exactly the same target and support.

Details

Dominated boxes mainly occur in outputs of the multiple peeling function (PRIM_peel_bs). So this function is practically only useful for "peel"-objects beeing the results of such a function. Without the dominated boxes the object gets much smaller by keeping all the relevant boxes. Also the trajectory (see plot.peel) without the not useful dominated boxes looks much clearer.

Value

This function returns another "peel"-object having the same structure as the input object, just without the dominated boxes.

See Also

PRIM_peel_bs, plot.peel

---

var_select *Sequential Relevance of Box-defining Variables*

Description

This function creates a table of the sequential relevance of the variables used in the definition of a box defined by define_fixbox.

Usage

```
var_select(fixbox)
```

Arguments

fixbox	object of class "fixbox" to be used to create the table of the sequential variable relevances.

Details

This function does a sequential removal off all variables defining one box. In each iteration every variable left in the box definition is tried to be completley left out of the definition. From all variables this one is removed, which causes the least decrease of the target function evaluated on all observations lying in the box.

Value

`var_select` returns a list with the following components:

`tab`	table showing the sequential relevances of the variables used in the definition of the `fixbox`.
`fixboxes`	`list` of the boxes defined at every iteration step (by removing of one variable).

See Also

`define_fixbox`, `PRIM`

Examples

```
# generating random data:
set.seed(123)
n <- 500
x1 <- runif(n = n, min = -1)
x2 <- runif(n = n, min = -1)
x3 <- runif(n = n, min = -1)
cat <- as.factor(sample(c("a","b","c", "d"), size = n, replace = TRUE))
wsk <- (1-sqrt(x1^2+x2^2)/sqrt(2))
y <- as.logical(rbinom(n = n, prob = wsk, size = 1))
dat <- cbind.data.frame(y, x1, x2, x3, cat)
plot(dat$x1, dat$x2, col=dat$y+1, pch=16)
remove(x1, x2, x3, y, wsk, cat, n)

# apply the PRIM function to find the best box with a support of at least 0.1:
p <- PRIM(y~., data=dat, beta_min = 0.1, max_boxes = 1, print_position = FALSE)

# sequential variable relevances:
var_select(p$fixboxes[[1]])$tab
```

Index