Package 'subtee'

February 25, 2021

Type Package

Title Subgroup Treatment Effect Estimation in Clinical Trials

Version 0.3-7.1

Date 2021-02-24

Author Nicolas Ballarini [aut, cre],
Bjoern Bornkamp [aut],
Marius Thomas [aut, cre],
Baldur Magnusson [ctb]

Maintainer Nicolas Ballarini <nicoballarini@gmail.com>

Description Naive and adjusted treatment effect estimation for subgroups. Model averaging (Bornkamp et.al, 2016 <doi:10.1002/pst.1796>) and bagging (Rosenkranz, 2016 <doi:10.1002/bimj.201500147>) are proposed to address the problem of selection bias in treatment effect estimates for subgroups. The package can be used for all commonly encountered type of outcomes in clinical trials (continuous, binary, survival, count). Additional functions are provided to build the subgroup variables to be used and to plot the results using forest plots.

Imports MASS, ggplot2, survival, matrixStats

Suggests knitr, rmarkdown, parallel, testthat

VignetteBuilder knitr

Depends R (>= 2.10)

License GPL-2

NeedsCompilation no

Repository CRAN

Date/Publication 2021-02-25 11:10:03 UTC

R topics documented:

---

bagged *Bootstrap estimates for interaction terms in exploratory subgroup analyses*

Description

Perform exploratory subgroup analysis using bootstrap bias adjustment as described in Rosenkranz (2016). The function fits a GLM or a Cox model in the data and then performs bootstrap samples to correct for bias.

Usage

```
bagged(resp, trt, subgr, covars = NULL, data,
       fitfunc = c("lm", "glm", "glm.nb", "survreg", "coxph", "rlm"),
       event, exposure,
       level = 0.1,
       B = 2000, mc.cores = 1, stratified = TRUE,
       select.by = c("BIC", "AIC"), quietly = FALSE, ...)
```

Arguments

resp	Character giving the name of the response variable. The variable can be either defined in the global environment or in the data-set data specified below. For interactive use it is also possible to use unquoted names (i.e. bagged(resp,...) instead of bagged("resp",...)), avoid this for non-interactive use of the function.
trt	Character giving the name of the treatment variable. The variable can be either defined in the global environment or in the data-set data specified below. Note that the treatment variable itself needs to be defined as a numeric variable, with control coded as 0, and treatment coded as 1. For interactive use it is also possible to use unquoted names (as for the resp argument. see above).
subgr	Character vector giving the variable names in data to use as subgroup identifiers. Note that the subgroup variables in data need to be numeric 0-1 variables.
covars	Formula, specifying additional covariates to be included in the models (need to be available in data).
data	Data frame containing the variables referenced in resp, trt, subgr and covars (and possibly event and exposure).

`fitfunc`	Model fitting functions. Currrently one of `'lm'`, `'glm'`, `'glm.nb'`, `'survreg'`, `'coxph'` or `'rlm'`.
`event`	Character giving the name of the event variable. Has to be specified when using fit functions `'survreg'` and `'coxph'`. The variable can be either defined in the global environment or in the data-set `data`.
`exposure`	Character giving the name of the exposure variable, needed for negative binomial regression, when using fit functions `'glm.nb'`. This is typically the time each patient is exposed to the drug. The fitted model uses the call `glm.nb(.~.+offset(log(exposure))`. The variable needs to be defined either in the global environment or in the data-set `data`.
`level`	Significance level for confidence intervals will be calculated for treatment effect estimates.
`B`	A numeric input. The number of bootstrap samples to perform.
`mc.cores`	A numeric input. This argument is passed to the `mclapply` function to perform computations in parallel. If `mc.cores = 1`, then `lapply` is used.
`stratified`	Should the bootstrap resampling be done stratifying by treatment group? (default: TRUE).
`select.by`	Should the model selection be done using BIC or AIC? (default: BIC).
`quietly`	A logical. By default (quietly = FALSE), `bagged` prints messages when a subgroup is not selected in any bootstrap sample or when the variance for the bootstrap estimate of one or more subgroups could not be calculated. If TRUE, these messages are not printed.
`...`	Other arguments passed to the model fitting function.

Details

In the generalized linear model case, P generalized linear models are fitted such that

$$M_p : h(\mu_{pi}) = \alpha_p + \beta_p z_i + (\gamma_p + \delta_p z_i) s_{pi} + \sum_{k=1}^{K} \tau_k x_{ik}$$

where h is the link function, $\mu_{pi} = E_p[Y_i]$ is the expectation of the response Y_i under model M_p and x_{ik} are additional covariates we control for. For survival data, a proportional hazards model can be used:

$$M_p : \lambda_{pi}(t) = \lambda_{p0}(t) \exp\left\{ \beta_p z_i + (\gamma_p + \delta_p z_i) s_{pi} + \sum_{k=1}^{K} \tau_k x_{ik} \right\}$$

The focus of estimation is the difference in the treatment effect between a subgroup and its complement, the treatment by subgroup interaction δ_p.

Consider now B bootstrap samples from the original data. Let $(Y^*_{b1}, ..., Y^*_{bN})$ be a bootstrap sample from the original data. Let $(z^*_{b1}, ..., z^*_{bN})$, $(s^*_{b1}, ..., s^*_{bN})$, and $(x^*_{b1k}, ..., x^*_{bNk})$ be the corresponding treatment indicators, group indicators, and covariates in the bootstrap samples, respectively. For each $p = 1, ..., P$ and $b = 1, ..., B$ we fit the model:

$$h(E_p[Y^*_{bi}]) = \alpha^*_{bp} + \beta^*_{bp} z^*_{bi} + (\gamma^*_{bp} + \delta^*_{bp} z^*_{bi}) s^*_{bpi} + \sum_{k=1}^{K} \tau_k x^*_{ik}$$

An estimator of δ_p given that subgroup S_p provided the best fit can be calculated as

$$\overline{\delta}_p^* = \frac{\sum_{b=1}^{B} u_{bp}\hat{\delta}_{bp}^*}{\sum_{b=1}^{B} u_{bp}}$$

where $\hat{\delta}_{bp}^*$ is the usual estimator of δ_{bp} and $u_{bp} = 1$ if the subgroup p provides the best fit for bootstrap sample b and 0 otherwise.

An bias-reduced estimator of δ_p can be obtained as:

$$\breve{\delta}_p^* = 2\hat{\delta}_p - \overline{\delta}_p^*$$

A bias-reduced estimator with decreased variability is obtained by replacing the maximum likelihood estimator by the bagging estimator $\hat{\delta}_{bp}^*$:

$$\hat{\delta}_p^* = \frac{1}{B}\sum_{b=1}^{B} \hat{\delta}_{bp}^*$$

so that the bias reduced estimator is

$$\tilde{\delta}_p^* = 2\hat{\delta}_p^* - \overline{\delta}_p^*$$

Value

An object of class subtee. A list containing a dataframe (model_fits) with the estimates using the original data, and a dataframe (bagged_results) with the bootstrap estimates with their percent of selection. The latter contains the following columns: 'percent_selected': the relative proportion for selection of each subgroup, 'bagg': the (uncorrected) bagged estimate $\hat{\delta}_p^*$ 'boot_red': the bias reduced bootstrap estimate $\breve{\delta}_p^*$ 'bagg_red': the bias reduced bootstrap estimate with decreased variability by bagging $\tilde{\delta}_p^*$ and the respective standard deviations of the estimates.

References

Rosenkranz, G.(2016) "Exploratory subgroup analysis in clinical trials by model selection", Biometrical Journal, 58, 1007-1259. doi: 10.1002/bimj.201500147

See Also

glm, coxph

Examples

```
## Not run:
## toy example calls using the simulated datnorm data-set without
## treatment and subgroup effect, see ?datnorm for details
data(datnorm)
head(datnorm)

## first need to create candidate subgroups (if not already defined in data-set)
## here generate candidate subgroups manually (need to be numeric 0-1 variables)
groups <- data.frame(labvalL.5=as.numeric(datnorm$labvalue < 0.5),
```

```
                    regUS=as.numeric(datnorm$region == "US"),
                    hgtL175=as.numeric(datnorm$height < 175))
fitdat <- cbind(datnorm, groups) # bind subgroup variables to main data
## subgroups of interest
subgr <- c("labvalL.5", "regUS", "hgtL175")
res <- bagged(resp = "y", trt = "treat", subgr = subgr, data = fitdat,
                      covars = ~ x1 + x2, fitfunc = "lm")
res


## generate candidate subgroups using the subbuild function
## semi-automatically i.e. some groups specified directly (height and
## smoker), for region and labvalue subbuild generates subgroups (see
## ?subbuild)
cand.groups <- subbuild(datnorm, height < 175, smoker == 1, region, labvalue)
head(cand.groups)
fitdat <- cbind(datnorm, cand.groups)
subgr <- colnames(cand.groups)
res <- bagged(resp = "y", trt = "treat", subgr = subgr, data = fitdat,
                      covars = ~ x1 + x2, fitfunc = "lm")
res

## toy example call for binary data on simulated datbin data-set
data(datbin)
cand.groups <- subbuild(datbin, height < 175, smoker == 1, region, labvalue)
fitdat <- cbind(datbin, cand.groups)
subgr <- colnames(cand.groups)
res <- bagged(resp = "y", trt = "treat", subgr = subgr, data = fitdat,
                      covars = ~ x1 + x2, fitfunc = "glm",
                      family = binomial(link = "logit"))
## scale of the treatment effect estimate: difference on log-odds scale
res

## toy example call for parametric and semi-parametric survival data on
## datsurv data-set
data(datsurv)
cand.groups <- subbuild(datsurv, height < 175, smoker == 1, region, labvalue)
fitdat <- cbind(datsurv, cand.groups)
subgr <- colnames(cand.groups)
res.survreg <- bagged(resp = "y", trt = "treat", subgr = subgr, data = fitdat,
                              covars = ~ x1 + x2,
                            fitfunc = "survreg", event = "event", dist = "exponential")
## parametric survival model (here exponential distribution)
## scale of treatment effect estimate: log scale (see ?survreg for details)
res.survreg

# Decreased B for a reduction in computational time
res.cox <- bagged(resp = "y", trt = "treat", subgr = subgr, data = fitdat,
                  fitfunc = "coxph", event = "event", B = 20) # B=2000 should be used
## scale of treatment effect estimate: difference in log-hazard rate
res.cox
## toy example call overdispersed count data on datcount data-set
data(datcount)
```

```
cand.groups <- subbuild(datcount, height < 175, smoker == 1, region, labvalue)
fitdat <- cbind(datcount, cand.groups)
subgr <- colnames(cand.groups)
# Decreased B for a reduction in computational time
res <- bagged(resp = "y", trt = "treat", subgr = subgr, data = fitdat,
              fitfunc = "glm.nb", exposure = "exposure", B = 20) # B=2000 should be used
## scale of treatment effect estimate: difference on log scale
res
## End(Not run)
```

`confint.subtee` *Confidence intervals for treatment effect estimates*

Description

Computes confidence intervals for subtee objects. This allows the recalculation of confidence intervals at a desired levels without fitting the models again, which is particularly useful for the results of the bagged function.

Usage

```
## S3 method for class 'subtee'
confint(object, parm, level = 0.95, ...)
```

Arguments

`object`	An object of class subtee, usually a result of a call to modav, unadj or bagged.
`parm`	Not used.
`level`	the confidence level required. Note that this is 1 - the significance level used in modav, unadj or bagged.
`...`	Not used.

Value

An object of class subtee

See Also

plot.subtee, summary.subtee

Examples

```
data(datnorm)
cand.groups <- subbuild(datnorm, height, labvalue, region, smoker)
fitd <- cbind(datnorm, cand.groups)
subgr <- colnames(cand.groups)
res <- modav(resp = "y", trt = "treat", subgr = subgr, data = fitd,
             covars = ~ x1 + x2, fitfunc = "lm")
confint(res, level = 0.80)
```

`get_prca_data` *Downloads the prca dataset to use in the package's examples (Internet connection is required).*

Description

Fetchs the Advanced prostate cancer data from Patrick Royston and Willi Sauerbrei' book Multivariable Model-building. Thw data comes from from a clinical trial with prostate carcinoma patients described in David P. Byar and Sylvan B. Green (1980).

Usage

```
get_prca_data()
```

Value

A data frame with 475 rows and 15 variables: PATNR, AGE, WT, SBP, DBP, SZ, AP, HG, SG, PF, HX, BM, STAGE, EKG, RX, SURVTIME, CENS, X_ST, X_D, X_T, X_T0

References

Royston, P., & Sauerbrei, W. (2008). Multivariable model-building: a pragmatic approach to regression anaylsis based on fractional polynomials for modelling continuous variables (Vol. 777). John Wiley & Sons.

Byar, D. P., & Green, S. B. (1980). The choice of treatment for cancer patients based on covariate information. Bulletin du cancer, 67(4), 477.

See Also

`bagged`, `unadj`, `modav`

Examples

```
prca = get_prca_data()
head(prca)
```

`modav` *Treatment effect estimation using model averaging based on marginal models.*

Description

Fits separate (marginal) models for each candidate subgroup, i.e. including the subgroup as a main effect and interaction with treatment for each model. These models are used predict the treatment effect for the subgroup of interest by predicting the effect for all patients in the subgroup and then averaging. These subgroup effects are then calculated for all models and then averaged according to posterior model weights. Details of the procedure are explained in Bornkamp et al. (2017) and Thomas and Bornkamp (2017).

Usage

```
modav(resp, trt, subgr, covars = NULL, data,
      fitfunc = c("lm", "glm", "glm.nb", "survreg", "coxph", "rlm"),
      event, exposure,
      level=0.1, prior = 1, nullprior = 0, ...)
```

Arguments

resp	Character giving the name of the response variable. The variable can be either defined in the global environment or in the data-set data specified below. For interactive use it is also possible to use unquoted names (i.e. modav(resp,...) instead of modav("resp",...)), avoid this for non-interactive use of the function.
trt	Character giving the name of the treatment variable. The variable can be either defined in the global environment or in the data-set data specified below. Note that the treatment variable itself needs to be defined as a numeric variable, with control coded as 0, and treatment coded as 1. For interactive use it is also possible to use unquoted names (as for the resp argument. see above).
subgr	Character vector giving the variable names in data to use as subgroup identifiers. Note that the subgroup variables in data need to be numeric 0-1 variables.
covars	Formula, specifying additional (prognostic) covariates to be included in the models (need to be available in data). It is crucial for the model averaging approach to include the important prognostic covariates (in particular if the corresponding prognostic covariate also defines a subgroup; otherwise models/subgroup might get upweighted just because the variable has prognostic value, but not because the treatment effect is modified).
data	Data frame containing the variables referenced in resp, trt, subgr and covars (and possibly event and exposure).
fitfunc	Model fitting functions. Currrently one of 'lm', 'glm', 'glm.nb', 'survreg', 'coxph' or 'rlm'.
event	Character giving the name of the event variable. Has to be specified when using fit functions 'survreg' and 'coxph'. The variable can be either defined in the global environment or in the data-set data.
exposure	Character giving the name of the exposure variable, needed for negative binomial regression, when using fit functions 'glm.nb'. This is typically the time each patient is exposed to the drug. The fitted model uses the call glm.nb(.~.+offset(log(exposure)) The variable needs to be defined either in the global environment or in the data-set data.
level	Significance level for confidence intervals will be calculated for treatment effect estimates.
prior	Numeric vector of prior model/subgroup probabilities of the same length as subgr. Order is assumed to be the same as in subgr. Probabilities can be specified up to proportionality. If a vector of length 1 is specified automatically equal prior weights are assumed (equal weights are the default).

`nullprior` Numeric giving the prior model probability of the model without any subgroup effect. This needs to be specified on the same scale as the prior argument. E.g. if there are 2 subgroups, `prior = c(1,1)` (or `prior = 1`) and `nullprior=2` the prior probabilities will be 1/4 and 1/4 for the two subgroup models and 1/2 for the null model. By default a prior probability of 0 is attached to this model.

`...` other arguments passed to the model fitting function.

Details

In the simple linear case (e.g when using fitfunc `lm`) for each of the P candidate subgroups the fitted model is of the form

$$M_p : y_i \sim N(\mu_i^{(p)}, \sigma_p^2), i = 1, ..., n$$

where

$$\mu_i^{(p)} = \alpha_p + \beta_p z_i + (\gamma_p + \delta_p z_i) s_{pi} + \sum_{k=1}^{K} \tau_k x_{ik}$$

where s_i denotes the subgroup indicators (the column vectors of `subgr`), z_i is the treatment indicator (from `trt`) and $x_{i1}, ..., x_{iK}$ are additional covariates as specified in `covars`. For other fitting functions the models are of similar form, including prognostic and predictive effects of subgroups.

A treatment effect (on the scale determined by `fitfunc`) of the candidate subgroups can be derived naively as $\hat{\beta}_p + \hat{\delta}_p$ and a treatment effect estimate for the complement is given by $\hat{\beta}_p$. Note that choosing subgroups based on these unadjusted treatment effect estimates may lead to overoptimistic conclusions in regards to the treatment effect in that subgroup. Naive estimates do not consider model selection uncertainty and will often suffer from selection bias.

For each subgroup a treatment effect is obtained by estimating the treatment effect for that subgroup under all models (by averaging the individual predictions in that subgroup) and approximating the resulting estimate within each model by a normal distribution for details see Bornkamp et al, 2017. Posterior model weights are obtained using BIC model weights (Raftery, 1995), so that overall a normal mixture is used to approximate the posterior distribution for every subgroup effect.

The returned treatment effect estimates are based on the median of this distribution, credible bounds are based on posterior quantiles.

Estimates of the interaction (difference in treatment effect between subgroup and complement) are also derived as the median and quantiles of the corresponding mixture distribution.

Value

A list (object of class `subtee`). The most important entries are (i) `fitmods` containing all fitted subgroup models and the overall model (ii) `trtEff` containing the treatment effect estimates and CI for subgroup and subgroup complements. (iii) `trtEffDiff` containing the differences in treatment effect estimates (subgroup vs complement) and CI.

References

Thomas, M., and Bornkamp, B. (2017) "Comparing Approaches to Treatment Effect Estimation for Subgroups in Early Phase Clinical Trials." Statistics in Biopharmaceutical Research, 9, 160-171, doi: 10.1080/19466315.2016.1251490

Bornkamp, B., Ohlssen, D., Magnusson, B. P., and Schmidli, H. (2017) "Model averaging for treatment effect estimation in subgroups." Pharmaceutical Statistics, 16, 133-142, doi: 10.1002/pst.1796

Raftery, A. E. (1995) "Bayesian model selection in social research." Sociological Methodology, 25, 111-163.

See Also

summary.subtee, plot.subtee, lm, glm, glm.nb, survreg, coxph

Examples

```
## toy example calls using the simulated datnorm data-set without
## treatment and subgroup effect, see ?datnorm for details
data(datnorm)
head(datnorm)

## first need to create candidate subgroups (if not already defined in data-set)
## here generate candidate subgroups manually (need to be numeric 0-1 variables)
groups <- data.frame(labvalL.5=as.numeric(datnorm$labvalue < 0.5),
                     regUS=as.numeric(datnorm$region == "US"),
                     hgtL175=as.numeric(datnorm$height < 175))
fitdat <- cbind(datnorm, groups) # bind subgroup variables to main data
## subgroups of interest
subgr <- c("labvalL.5", "regUS", "hgtL175")
res <- modav(resp = "y", trt = "treat", subgr = subgr, data = fitdat,
             covars = ~ x1 + x2, fitfunc = "lm")
summary(res)
plot(res, show.compl=TRUE)
## compare to unadjusted analysis
res <- unadj(resp = "y", trt = "treat", subgr = subgr, data = fitdat,
             covars = ~ x1 + x2, fitfunc = "lm")
summary(res)
plot(res)

## generate candidate subgroups using the subbuild function
## semi-automatically i.e. some groups specified directly (height and
## smoker), for region and labvalue subbuild generates subgroups (see
## ?subbuild).
cand.groups <- subbuild(datnorm, height < 175, smoker == 1, region, labvalue)
head(cand.groups)
fitdat <- cbind(datnorm, cand.groups)
subgr <- colnames(cand.groups)
resMA <- modav(resp = "y", trt = "treat", subgr = subgr, data = fitdat,
               covars = ~ x1 + x2, fitfunc = "lm")
summary(resMA)
plot(resMA, show.compl = TRUE)

## toy example call for binary data on simulated datbin data-set
data(datbin)
cand.groups <- subbuild(datbin, height < 175, smoker == 1, region, labvalue)
fitdat <- cbind(datbin, cand.groups)
subgr <- colnames(cand.groups)
```

```
res <- modav(resp = "y", trt = "treat", subgr = subgr, data = fitdat,
             covars = ~ x1 + x2, fitfunc = "glm",
             family = binomial(link = "logit"))
## scale of the treatment effect estimate: difference on log-odds scale
summary(res)
plot(res)

## toy example call for parametric and semi-parametric survival data on
## datsurv data-set
data(datsurv)
cand.groups <- subbuild(datsurv, height < 175, smoker == 1, region, labvalue)
fitdat <- cbind(datsurv, cand.groups)
subgr <- colnames(cand.groups)
res.survreg <- modav(resp = "y", trt = "treat", subgr = subgr, data = fitdat,
                     covars = ~ x1 + x2,
                     fitfunc = "survreg", event = "event", dist = "exponential")
## parametric survival model (here exponential distribution)
## scale of treatment effect estimate: log scale (see ?survreg for details)
summary(res.survreg)
plot(res.survreg)
res.cox <- modav(resp = "y", trt = "treat", subgr = subgr, data = fitdat,
                 covars = ~ x1 + x2, fitfunc = "coxph", event = "event")
## scale of treatment effect estimate: difference in log-hazard rate
summary(res.cox)
plot(res.cox)

## toy example call overdispersed count data on datcount data-set
data(datcount)
cand.groups <- subbuild(datcount, height < 175, smoker == 1, region, labvalue)
fitdat <- cbind(datcount, cand.groups)
subgr <- colnames(cand.groups)
res <- modav(resp = "y", trt = "treat", subgr = subgr, data = fitdat,
             covars = ~ x1 + x2, fitfunc = "glm.nb", exposure = "exposure")
## scale of treatment effect estimate: difference on log scale
summary(res)
plot(res)
```

`plot.subtee` *Plotting subgroup treatment effect estimates*

Description

Plotting function for objects of class 'subtee'. Visualizes estimates and confidence intervals for all candidate subgroups (and complements) in a forest plot.

Usage

```
## S3 method for class 'subtee'
plot(x, y = NULL, z = NULL,
     type = c("trtEff", "trtEffDiff"),
```

```
show.compl = FALSE,
xlab = "default", ylab = "default", main = "default",
them,
point.size = 2.5, line.size = 1,
palette = "default", ...)
```

Arguments

x	An object of class subtee, usually a result of a call to modav, unadj or bagged.
y	An object of class subtee, usually a result of a call to modav, unadj or bagged. If this is used, a comparison between the estimates will be provided.
z	An object of class subtee, usually a result of a call to modav, unadj or bagged. If this is used, a comparison between the estimates will be provided.
type	A character specifyng if treatment effects should be plotted ("trtEff") or the treatment-subgroup interactions. ("trtEffDiff")
show.compl	Logical. If true estimates for candidate subgroup complements should be plotted as well. Only available if type = "trtEff".
xlab	Character. Label for x-axis.
ylab	Character. Label for y-axis.
main	Character. Title. The default is to provide a string with the level of the uncertainty intervals.
them	ggplot2 theme. Use theme() if you just need to tweak the display, or provide a complete ggplot2 theme (e.g theme_bw()).
point.size	Size for points, which denote point estimates of treatment effects. Default to 2.5.
line.size	Size for points, which denote confidence interval of treatment effects. Default to 1.
palette	A string providing a ggplot2 colour palette to use. This will be passed to the palette option in a scale_colour_brewer sentence.
...	Not used.

Value

Forest plot visualizing treatment effect estimates (if type = "trtEff") or treatment-subgroup interactions in candidate subgroups (if type = "trtEffDiff").

See Also

summary.subtee

Examples

```
data(datnorm)
cand.groups <- subbuild(datnorm, height, labvalue, region, smoker)
fitd <- cbind(datnorm, cand.groups)
subgr <- colnames(cand.groups)
```

```
### Plot unadjusted estimates
res_unadj <- unadj(resp = "y", trt = "treat", subgr = subgr, data = fitd,
             covars = ~ x1 + x2, fitfunc = "lm")
summary(res_unadj)
plot(res_unadj)
plot(res_unadj, show.compl = TRUE)
plot(res_unadj, type = "trtEffDiff")

### Compare unadjusted with model averaging estimates
res_modav <- modav(resp = "y", trt = "treat", subgr = subgr, data = fitd,
             covars = ~ x1 + x2, fitfunc = "lm")
plot(res_unadj, res_modav, show.compl = TRUE)
plot(res_unadj, res_modav, type = "trtEffDiff")
```

`Simulated data-sets` *Simulated example data-sets*

Description

Simulated test data-sets to illustrate and test methods in this package. The underlying simulation truth assumed that there is no treatment effect and no subgroup effect.

The data set was simulated using the R code in the tests/data-sets.R file. The data-sets (`datnorm`, `datbin`, `datsurv`, `datcount`) only differ in the y variable (covariates are the same).

Usage

```
data(datbin)

data(datnorm)

data(datcount)

data(datsurv)
```

Format

Data frames with 100 observations on the following variables.

`y` Response variable

`treat` Categorical variable

`height` Numeric variable

`labvalue` Numeric variable

`region` Categorical variable

`smoker` Categorical variable

`x1` Numeric variable

`x2` Numeric variable

subbuild *Generating candidate subgroups based on an input data set*

Description

Takes categorical or continuous baseline covariate vectors and builds a matrix of binary candidate subgroups indicators.

Usage

```
subbuild(data, ..., n.cuts = 2, dig.lab = 3, dupl.rm = FALSE,
         make.valid.names = FALSE)
```

Arguments

data	Data frame. Contains baseline covariates from which candidate subgroups are generated. Categorical covariates should be of type factor.
...	Optional subgroup definitions or variable names, which are used to generate candidate subgroups.
n.cuts	Integer. Number of cutoffs for each covariate.
dig.lab	Integer. Digit to which subgroup cutoffs are rounded.
dupl.rm	Logical. Remove duplicate subgroups. Note that this applies also to two subgroup vectors a and b that satisfy a=1-b (i.e. only labels 0 and 1 exchanged), because these will give the same model fit (only the label of "subgroup" and "complement" are exchanged).
make.valid.names	Logical. If TRUE subgroup names in the final output are transformed to be syntactically valid names (see ?make.names)

Details

The ... argument allows manual specification of subgroups that should be included. Subgroup definitions should be passed as (typically logical) expressions, that can be evaluated on data and result in binary subgroup indicator variables.

If only a variable name is specified in ..., subgroup definitions based on this covariate are automatically generated in the following way: For covariates of type factor or character candidate subgroups are the patients in each category. For covariates of type numeric or integer, cutpoints for candidate subgroups are generated based on covariate quantiles. For each continuous covariates 'n.cuts' + 1 non-overlapping subgroups of (roughly) the same size are generated.

If no information about subgroups is supplied in ..., candidate subgroups are automatically generated for all variables in data. Subgroup names are taken from the column names of the 'data' data frame (or set to x1, x2,... if no names are supplied)

If dupl.rm is TRUE any duplicate columns (either subgroup or complement is equal to another column) are removed from the final output.

Value

A data frame of candidate subgroups.

See Also

bagged, modav, unadj

Examples

```
data(datnorm)
## data frame of covariates considered for subgroup analysis
cov.dat <- datnorm[,c("height", "labvalue", "region", "smoker")]
## by default generate all subgroups for each categorical variable and
## use cut-offs based on quantiles for numeric variables
cand.groups <- subbuild(cov.dat)
head(cand.groups)
## alternatively use
cand.groups <- subbuild(datnorm, height, labvalue, region, smoker)
head(cand.groups)

## use more cutpoints
cand.groups2 <- subbuild(cov.dat, n.cuts = 4)
ncol(cand.groups)
ncol(cand.groups2)

## remove duplicate columns for smoker
cand.groups3 <- subbuild(cov.dat, dupl.rm = TRUE)
head(cand.groups3)
ncol(cand.groups3)

## syntactically valid names
cand.groups4 <- subbuild(cov.dat, make.valid.names = TRUE)
head(cand.groups4)

## manually specify subgroup definitions and which covariates to consider
cand.groups5 <- subbuild(cov.dat, region == "EU", height > 172, labvalue)
## note that for labvalue cut-offs are generated automatically based on quantiles
head(cand.groups5)

## further examples for manual specification of subgroups
cand.groups6 <- subbuild(cov.dat, region %in% c("Japan","EU"), smoker != 0)
## note that for labvalue cut-offs are generated automatically based on quantiles
head(cand.groups6)

## missing values in data-set are propagated through
cov.dat$labvalue[sample(1:nrow(cov.dat),10)] <- NA
cov.dat$region[sample(1:nrow(cov.dat),20)] <- NA
cov.dat$smoker[sample(1:nrow(cov.dat),10)] <- NA
cand.groups7 <- subbuild(cov.dat)
head(cand.groups7)

## if covariates in the data frame contain missing values consider
```

```
## imputing them for example with the rfImpute function from the
## randomForest package
```

---

summary.subtee *Summarizing subgroup analyses estimates*

Description

Summary function for subtee objects. Shows estimates and confidence interval boundaries for all candidate subgroups (and complements).

Usage

```
## S3 method for class 'subtee'
summary(object, ...)
```

Arguments

object	An object of class subtee, usually a result of a call to modav, unadj or bagged.
...	Not used.

Value

A dataframe containing information about treatment effects and group sizes in candidate subgroups.

See Also

plot.subtee

Examples

```
data(datnorm)
cand.groups <- subbuild(datnorm, height, labvalue, region, smoker)
fitd <- cbind(datnorm, cand.groups)
subgr <- colnames(cand.groups)
res <- modav(resp = "y", trt = "treat", subgr = subgr, data = fitd,
             covars = ~ x1 + x2, fitfunc = "lm")
summary(res)
```

unadj *Treatment effect estimation based on marginal subgroup models.*

Description

Unadjusted estimation of treatment effects in subgroups. Fits separate (marginal) models for each candidate subgroup, i.e. including the subgroup as a main effect and interaction with treatment for each model.

Usage

```
unadj(resp, trt, subgr, covars = NULL, data,
      fitfunc = c("lm", "glm", "glm.nb", "survreg", "coxph", "rlm"),
      event, exposure, level = 0.1, ...)
```

Arguments

resp	Character giving the name of the response variable. The variable can be either defined in the global environment or in the data-set data specified below. For interactive use it is also possible to use unquoted names (i.e. unadj(resp,...) instead of unadj("resp",...)), avoid this for non-interactive use of the function.
trt	Character giving the name of the treatment variable. The variable can be either defined in the global environment or in the data-set data specified below. Note that the treatment variable itself needs to be defined as a numeric variable, with control coded as 0, and treatment coded as 1. For interactive use it is also possible to use unquoted names (as for the resp argument. see above).
subgr	Character vector giving the variable names in data to use as subgroup identifiers. Note that the subgroup variables in data need to be numeric 0-1 variables.
covars	Formula, specifying additional (prognostic) covariates to be included in the models (need to be available in data). It is crucial for the model averaging approach to include the important prognostic covariates (in particular if the corresponding prognostic covariate also defines a subgroup; otherwise models/subgroup might get upweighted just because the variable has prognostic value, but not because the treatment effect is modified).
data	Data frame containing the variables referenced in resp, trt, subgr and covars (and possibly event and exposure).
fitfunc	Model fitting functions. Currrently one of 'lm', 'glm', 'glm.nb', 'survreg', 'coxph' or 'rlm'.
event	Character giving the name of the event variable. Has to be specified when using fit functions 'survreg' and 'coxph'. The variable can be either defined in the global environment or in the data-set data.
exposure	Character giving the name of the exposure variable, needed for negative binomial regression, when using fit functions 'glm.nb'. This is typically the time each patient is exposed to the drug. The fitted model uses the call glm.nb(.~.+offset(log(exposure))

	The variable needs to be defined either in the global environment or in the dataset `data`.
`level`	Significance level for confidence intervals will be calculated for treatment effect estimates.
...	other arguments passed to the model fitting function.

Details

In the simple linear case (e.g when using fitfunc `lm`) for each of the P candidate subgroups the fitted model is of the form

$$M_p : y_i \sim N(\mu_i^{(p)}, \sigma_p^2), i = 1, ..., n$$

where

$$\mu_i^{(p)} = \alpha_p + \beta_p z_i + (\gamma_p + \delta_p z_i) s_{pi} + \sum_{k=1}^{K} \tau_k x_{ik}$$

where s_i denotes the subgroup indicators (the column vectors of `subgr`), z_i is the treatment indicator (from `trt`) and $x.1, ..., x.K$ are additional covariates as specified in `covars`. For other fitting functions the models are of similar form, including prognostic and predictive effects of subgroups.

A treatment effect (on the scale determined by `fitfunc`) for the candidate subgroups is estimated as $\hat{\beta} + \hat{\delta}_p$ and a treatment effect estimate for the complement is given by $\hat{\beta}$. Note that choosing subgroups based on these unadjusted treatment effect estimates may lead to overoptimistic conclusions in regards to the treatment effect in that subgroup. Naive estimates do not consider model selection uncertainty and will often suffer from selection bias.

Value

A list (object of class `subtee`). The most important entries are (i) `fitmods` containing all fitted subgroup models and the overall model (ii) `trtEff` containing the treatment effect estimates and CI for subgroup and subgroup complements. (iii) `trtEffDiff` containing the differences in treatment effect estimates (subgroup vs complement) and CI.

References

Thomas, M., and Bornkamp, B. (2017) "Comparing Approaches to Treatment Effect Estimation for Subgroups in Early Phase Clinical Trials." Statistics in Biopharmaceutical Research, 9, 160-171, doi: 10.1080/19466315.2016.1251490

Bornkamp, B., Ohlssen, D., Magnusson, B. P., and Schmidli, H. (2017) "Model averaging for treatment effect estimation in subgroups." Pharmaceutical Statistics, 16, 133-142, doi: 10.1002/pst.1796

Raftery, A. E. (1995) "Bayesian model selection in social research." Sociological Methodology, 25, 111-163.

See Also

`summary.subtee`, `plot.subtee`, `lm`, `glm`, `glm.nb`, `survreg`, `coxph`

Examples

```
## toy example calls using the simulated datnorm data-set without
## treatment and subgroup effect, see ?datnorm for details
data(datnorm)
head(datnorm)

## first need to create candidate subgroups (if not already defined in data-set)
## here generate candidate subgroups manually (need to be numeric 0-1 variables)
groups <- data.frame(labvalL.5=as.numeric(datnorm$labvalue < 0.5),
                     regUS=as.numeric(datnorm$region == "US"),
                     hgtL175=as.numeric(datnorm$height < 175))
fitdat <- cbind(datnorm, groups) # bind subgroup variables to main data
## subgroups of interest
subgr <- c("labvalL.5", "regUS", "hgtL175")
res <- unadj(resp = "y", trt = "treat", subgr = subgr, data = fitdat,
             covars = ~ x1 + x2, fitfunc = "lm")
summary(res)
plot(res)

## generate candidate subgroups using the subbuild function
## semi-automatically i.e. some groups specified directly (height and
## smoker), for region and labvalue subbuild generates subgroups (see
## ?subbuild).
cand.groups <- subbuild(datnorm, height < 175, smoker == 1, region, labvalue)
head(cand.groups)
fitdat <- cbind(datnorm, cand.groups)
subgr <- colnames(cand.groups)
res <- unadj(resp = "y", trt = "treat", subgr = subgr, data = fitdat,
             covars = ~ x1 + x2, fitfunc = "lm")
summary(res)
plot(res)

## toy example call for binary data on simulated datbin data-set
data(datbin)
cand.groups <- subbuild(datbin, height < 175, smoker == 1, region, labvalue)
fitdat <- cbind(datbin, cand.groups)
subgr <- colnames(cand.groups)
res <- unadj(resp = "y", trt = "treat", subgr = subgr, data = fitdat,
             covars = ~ x1 + x2, fitfunc = "glm",
             family = binomial(link = "logit"))
## scale of the treatment effect estimate: difference on log-odds scale
summary(res)
plot(res)

## toy example call for parametric and semi-parametric survival data on
## datsurv data-set
data(datsurv)
cand.groups <- subbuild(datsurv, height < 175, smoker == 1, region, labvalue)
fitdat <- cbind(datsurv, cand.groups)
subgr <- colnames(cand.groups)
res.survreg <- unadj(resp = "y", trt = "treat", subgr = subgr, data = fitdat,
                     covars = ~ x1 + x2,
```

```
                    fitfunc = "survreg", event = "event", dist = "exponential")
## parametric survival model (here exponential distribution)
## scale of treatment effect estimate: log scale (see ?survreg for details)
summary(res.survreg)
plot(res.survreg)
res.cox <- unadj(resp = "y", trt = "treat", subgr = subgr, data = fitdat,
                 covars = ~ x1 + x2, fitfunc = "coxph", event = "event")
## scale of treatment effect estimate: difference in log-hazard rate
summary(res.cox)
plot(res.cox)

## toy example call overdispersed count data on datcount data-set
data(datcount)
cand.groups <- subbuild(datcount, height < 175, smoker == 1, region, labvalue)
fitdat <- cbind(datcount, cand.groups)
subgr <- colnames(cand.groups)
res <- unadj(resp = "y", trt = "treat", subgr = subgr, data = fitdat,
             covars = ~ x1 + x2, fitfunc = "glm.nb", exposure = "exposure")
## scale of treatment effect estimate: difference on log scale
summary(res)
plot(res)
```

Index