Package 'copcor'

August 1, 2024

LazyLoad yes

Version 2024.7-31

Title Correlates of Protection and Correlates of Risk Functions

Depends R (>= 3.6), kyotil

Imports methods

Suggests RUnit, R.rsp, survival

Description Correlates of protection (CoP) and correlates of risk (CoR) study the immune biomarkers associated with an infectious disease outcome, e.g. COVID or HIV-1 infection. This package contains shared functions for analyzing CoP and CoR, including bootstrapping procedures, competing risk estimation, and bootstrapping marginalized risks.

VignetteBuilder R.rsp

License GPL (>= 2)

NeedsCompilation no

Author Youyi Fong [cre],
Yiwen He [aut],
Chenchen Yu [aut],
Bhavesh Borate [aut],
Peter Gilbert [aut]

Maintainer Youyi Fong <youyifong@gmail.com>

Repository CRAN

Date/Publication 2024-07-31 22:30:31 UTC

Contents

`copcor` *copcor*

Description

Functions used in the study of correlates of protection and correlates of risk.

See the Index link below for a list of available functions.

`cove.boost.collapse.strata`
Collapse sample strata for COVE boost correlates study

Description

Collapse sample strata for COVE boost correlates study

Usage

```
cove.boost.collapse.strata (dat.b, n.demo)
```

Arguments

`dat.b`	data frame
`n.demo`	number of demographics strata, e.g. 6 for COVE correlates

Details

This function is used by both correlates_processing repo and correlates_reporting3 repo

Value

dat.b, whose Wstratum has been updated

plotting *Plotting Helper Functions*

Description

Functions for plotting.

Usage

```
draw.x.axis.cor(xlim, llox, llox.label, for.ggplot=FALSE)
get.xlim(dat, marker, lloxs)
```

Arguments

xlim	xlim
llox	lower limit
llox.label	label for lower limit
for.ggplot	Boolean
dat	data frame
marker	name of the biomarker variable
lloxs	list of lloxs

Details

draw.x.axis.cor is used by both cor_coxph and cor_threshold

Value

real

predictCompetingRisk *Cumulative Incidence Function (CIF) Under Competing Risk*

Description

Offers two approaches (Approach 2 is recommended, pcr2 is just an alias for predictCompetingRisk2). Weights are allowed in the optional arguments.

Usage

```
predictCompetingRisk2(formula.list, data, t0, newdata = data, ...)
pcr2(formula.list, data, t0, newdata=data, ...)

predictCompetingRisk(formula, formula.all, data, t0, newdata=data, stype=2, ctype=2, ...)
pcr(formula, formula.all, data, t0, newdata=data, stype=2, ctype=2, ...)
```

Arguments

formula.list	list of formulae for cause-specific failures. Assume the first cause to be the cause of interest
formula	formula for the cause-specific failure
formula.all	formula for all-cause failure
data	data frame
t0	the time till which cumulative incidence function is computed
newdata	new data for making prediction, default to the data for fitting the models
stype	computation of the survival curve, 1=direct, 2= exponenial of the cumulative hazard. Default 2, which is the default of basehaz and predict.coxph
ctype	whether the cumulative hazard computation should have a correction for ties, 1=no, 2=yes. Default 2, which is the default of basehaz and predict.coxph
...	optional arguments that are passed to coxph, the most import of which is weights

Details

Approach 2, predictCompetingRisk2, fits cause-specific Cox models to each cause to compute cumulative incidence function for the cause of interest under competing risk.

When there is only one cause, CIF is conceptually 1 - surival prob.(https://www.publichealth.columbia.edu/research/populatio
health-methods/competing-risk-analysis)

The function is implemented in R with matrix operation. Because looping through time points and subjects is vectorized, it is quite fast (faster than riskRegression in limited testing, which implements in C, but pcr uses more memory.)

One way to check the implementation of this function is to compare its results with the results of predict.coxph when there is only one cause. The tests in the examples code below show that when the risk is small (e.g. shorter followup time), the CIF computed by this function and the 1-survival estimated via 1-exp(-H) by predict.coxph, where H is cumulative hazard, are close to each other. But when the risk is high, the difference between the two are more noticeable. These results make sense because, e.g.,

If t0 = the first time failure point, CIF = h1 = H1 ~ 1-exp(-H1) If t0 = the second time point,

CIF = H1 + exp(-H1) * h2 (by def)

~ H1 + exp(-H1)(1 - exp(-h2))

= H1 + exp(-H1) - exp(-H2)

~ 1 - exp(-H2)

Approach 1, predictCompetingRisk, fits a cause-specific Cox model and a all-cause Cox model to compute cumulative incidence function for the cause of interest under competing risk.

The difference between predictCompetingRisk and predictCompetingRisk2 is that instead of fitting a model to the overall failure, a model is fit for each cause, including the cause of interest. The overall survival is computed by adding together the cumulative hazard from individual causes.

The second approach is recommended because it is more stable.

Value

A vector of real numbers as the risk till t0 for each subject in newdata

References

riskRegression: Predicting the Risk of an Event using Cox Regression Models by Brice Ozenne, Anne Lyngholm Sorensen, Thomas Scheike, Christian Torp-Pedersen, Thomas Alexander Gerds https://journal.r-project.org/archive/2017/RJ-2017-062/RJ-2017-062.pdf Thanks to Professor Gerds for helpful discussion.

Competing Risk Analysis Columbia Public Health https://www.publichealth.columbia.edu/research/population-health-methods/competing-risk-analysis

Introduction to the Analysis of Survival Data in the Presence of Competing Risks Peter C Austin, Douglas S Lee, Jason P Fine https://www.ahajournals.org/doi/full/10.1161/CIRCULATIONAHA.115.017719

See Also

predictCompetingRisk.

Examples

```
library(survival)

# prepare a dataset with competing risk
lung1=lung[order(lung$time),]
lung1$status=lung1$status-1
lung1$status[1:50]=2
with(lung1, table(status))
lung1$status.1=ifelse(lung1$status==1,1,0)
lung1$status.2=ifelse(lung1$status==2,1,0)
lung1$status.a=ifelse(lung1$status==0,0,1)
lung1$wt=rep(1, nrow(lung1))


################################################################################
# predictCompetingRisk2

t0=1000
formula.list=list(
    Surv(time, status.1) ~ age,
    Surv(time, status.2) ~ age
)

cif.2=pcr2(formula.list, lung1, t0)

fit=coxph(formula.list[[1]], lung1)
newdata=lung1
newdata$time=t0
coxpred = 1 - exp(-predict(fit, newdata=newdata, type="expected"))
```

```
plot(cif.2, coxpred)


# dealing with weights
lung1$wt=c(rep(2,50), rep(1, nrow(lung1)-50))
cif=predictCompetingRisk2(formula.list, lung1, t0, weights=lung1$wt)


#############################################################################
# predictCompetingRisk

t0=1000
form  =Surv(time, status.1) ~ age
form.a=Surv(time, status.a) ~ age
cif=predictCompetingRisk(form, form.a, lung1, t0, newdata=lung1, weights=lung1$wt, stype=2,ctype=2)

# more validation code

# when there is no covariate and one cause, CIF = 1 - KM estimate of survival prob

lung1=lung[order(lung$time),]
lung1$status=lung1$status-1
with(lung1, table(status))
lung1$status.1=ifelse(lung1$status==1,1,0)
lung1$status.a=ifelse(lung1$status==0,0,1)
lung1$wt=rep(1, nrow(lung1))

# stype=2 is surv=prod limit
fitKM <- survfit(Surv(time, status.1) ~ 1, data=lung1, stype=1, ctype=2)
cbind(summary(fitKM)$cumhaz, exp(-summary(fitKM)$cumhaz), summary(fitKM)$surv)[1:2,]
#[1,] 0.004385965 0.9956236 0.9956140
#[2,] 0.017660474 0.9824946 0.9824561
cif=predictCompetingRisk(Surv(time, status.1) ~ 1, Surv(time, status.1) ~ 1, lung1, t0=11,
    newdata=lung1[1,,drop=FALSE], weights=lung1$wt, stype=1, ctype=2)
cif # 0.01754386
1-cif # 0.9824561 = summary(fitKM)$surv at t=11


# when there are covariates and one cause, CIF and 1-exp(-H) are close to each other
# when H is small but not close when H is large
form  =Surv(time, status.1) ~ age
form.a=Surv(time, status.a) ~ age

oldpar <- par(mfrow = c(1,2))
for (t0 in c(12,1000)) {
    fit=coxph(form, lung1, weights=lung1$wt)
    lung2=lung1; lung2$time=t0
    r=predict(fit, type="expected", newdata=lung2)
    message(head(basehaz(fit, centered=TRUE)))
   cif=predictCompetingRisk(form, form.a, lung1, t0, newdata=lung1, weights=lung1$wt, stype=2,
        ctype=2)
    plot(cif, 1-exp(-r), xlab="Cumulative incidence function",
        ylab="Expected number of events from predict.coxph", main=paste0("t0: ", t0))
```

```
    abline(0,1)
    message(head(cbind(cif, "1-exp(-r)"=1-exp(-r))))
}
par(oldpar)
```

`utils` *Utility Functions*

Description

Helpful functions.

Usage

```
marker.name.to.assay (a)
```

Arguments

a	assay name

Details

This function ...

Value

string

Index