Package 'obsSens'

May 9, 2026

Type Package

Title Sensitivity Analysis for Observational Studies

Version 1.4

Date 2022-04-23

Author Greg Snow <538280@gmail.com>

Maintainer Greg Snow <538280@gmail.com>

Description Observational studies are limited in that there could be an unmeasured variable related to both the response variable and the primary predictor. If this unmeasured variable were included in the analysis it would change the relationship (possibly changing the conclusions). Sensitivity analysis is a way to see how much of a relationship needs to exist with the unmeasured variable before the conclusions change. This package provides tools for doing a sensitivity analysis for regression (linear, logistic, and cox) style models.

License GPL-2

Repository CRAN

Repository/R-Forge/Project obssens

Repository/R-Forge/Revision 19

Repository/R-Forge/DateTimeStamp 2022-04-23 19:32:34

Date/Publication 2022-04-23 23:50:07 UTC

NeedsCompilation no

Contents

obsSens-package 2
obsSensCCC 3
print.obsSens 6

Index **8**

obsSens-package *Perform sensitivity analysis on observational studies to explore possible effects of unmeasured (lurking) variables.*

Description

This package provides functions for doing sensitivity analysis on coefficients of regression type models (regression, logistic regression, cox proportional hazards). These assume a true model of the form: g(y)=beta*x+gamma*u+theta*z where u is an unmeasured potential lurking variable, x is the main variable of interest (treatment) and z represents other potential variables in the model. The response variable (y) can be continuous, binary, or a survival object. These functions examine the effect of u on beta for different values of gamma and the relationship between u and y.

Details

Package:	obsSens
Type:	Package
Version:	1.0
Date:	2007-12-21
License:	Artistic-2.0

The key functions are all of the form obsSensYXU where Y specifies the type of variable used as the response variable (y), X specifies the type of variable used as the main predictor variable to be tested (x), and U specifies the type of unmeasured variable to use. They can take on the following values: S - survival analysis (Y only), C - Categorical (logistic regression, currently only handles 2 levels), or N - normal (or continuous variables).

All the functions take either a fitted model object (lm, glm, or coxph) or a coefficient value and its confidence interval. You then specify values (vector) for the possible relationship between Y and U and X and U. The return value is a list with a matrix or array with the adjusted coefficients and upper and lower confidence limits.

Author(s)

Greg Snow <538280@gmail.com>

References

Lin, DY and Psaty, BM and Kronmal, RA. (1998): Assessing the Sensitivity of Regression Results to Unmeasured Confounders in Observational Studies. Biometrics, 54 (3), Sep, pp. 948-963.

Baer, VL et. als (2007): Do Platelet Transfusions in the NICU Adversely Affect Survival? Analysis of 1600 Thrombocytopenic neonates in a mulihospital healthcare system. Journal of Perinatology, 27, pp. 790-796.

Examples

```
# Recreate tables from above references

obsSensCCC( log(23.1), log(c(6.9, 77.7)), g0=c(2,6,10),
  p0=seq(0,.5,.1), p1=seq(0,1,.2) )

obsSensSCC( log(1.21), log(c(1.09,1.25)),
  p0=seq(0,.5,.1), p1=seq(0,1,.1), g0=3 )


obsSensCNN( log(1.14), log(c(1.10,1.18)),
  rho=c(0,.5, .75, .85, .9, .95, .98, .99),
  gamma=seq(0,1,.2), sdx=4.5 )
```

---

obsSensCCC *Compute Sensitivity analysis for Observational Studies*

Description

Computes a Sensitivity analysis for a coefficient from a regression model (linear, logistic, Cox) on Observational data. Computes new coefficient estimate and confidence interval for various relationships between the response, predictor of interest, and an unmeasured variable. The last 3 letters of each function refer to the type of the variables in the order Y, X, U with C meaning categorical, N meaning normal (continuous), and S being a survival object.

Usage

```
obsSensCCC(model, which = 2, g0 = c(2, 6, 10), g1, p0 = seq(0, 1, 0.2),
           p1 = p0, logOdds = FALSE, method = c("approx", "sim"))

obsSensCCN(model, which = 2, gamma = round(seq(0, 2 * bstar,
           length = 6), 4), delta = seq(0, 3, 0.5), logOdds = FALSE,
           method = c("approx", "sim"))

obsSensCNN(model, which = 2,
           gamma = round(seq(0, 2 * bstar, length = 6), 4),
           rho = c(0, 0.5, 0.75, 0.85, 0.9, 0.95, 0.98, 0.99), sdx,
           logOdds = FALSE, method = c("approx", "sim"))

obsSensNCC(model, which = 2, g0 = c(2, 6, 10), g1, p0 = seq(0, 1, 0.2),
           p1 = p0, log = TRUE, method = c("approx", "sim"))

obsSensNCN(model, which = 2, gamma = round(seq(0, 3, length = 6), 4),
           delta = seq(0, 3, 0.5), log = TRUE,
           method = c("approx", "sim"))
```

```
obsSensNNN(model, which = 2,
           gamma = round(seq(0, 2 * bstar, length = 6), 4),
           rho = c(0, 0.5, 0.75, 0.85, 0.9, 0.95, 0.98, 0.99),
           sdx, log = TRUE, method = c("approx", "sim"))

obsSensSCC(model, which = 2, g0 = c(2, 6, 10), g1, p0 = seq(0, 1, 0.2),
           p1 = p0, logHaz = FALSE, method = c("approx", "sim"))

obsSensSCN(model, which = 2,
           gamma = round(seq(0, 3, length = 6), 4),
           delta = seq(0, 3, 0.5), logHaz = FALSE,
           method = c("approx", "sim"))

obsSensSNN(model, which = 2, gamma = round(seq(0, 3, length = 6), 4),
           rho = c(0, 0.5, 0.75, 0.85, 0.9, 0.95, 0.98, 0.99), sdx,
           logHaz = FALSE, method = c("approx", "sim"))
```

Arguments

model	A regression model object (result of lm, glm, or coxph or other object with a coef method) or a single numerical value representing the estimated coefficient from a regression model.
which	Which coefficient (in the results from coef(model)) corresponds to the predictor of interest. If model is a number then this should be a 2 element vector with the initial confidence interval for model.
g0	The slopes on the unmeasured variable when x=0 (on the log scale, not response).
g1	The slopes on the unmeasured variable when x=1, if missing then g1=g0=gamma is assumed.
p0	Probability that U=1 given x=0.
p1	Probability that U=1 given x=1.
logOdds	Should the resulting table be on the log scale or response scale.
log	Should the resulting table be on the log scale or not.
logHaz	Should the resulting table be on the log Hazard ratio scale or Hazard ratio scale.
method	Either "approx" or "sim", only approx is currently implemented.
gamma	Slopes for U (unmeasured variable).
delta	The difference between the mean of U|x=0 and mean of U|x=1.
rho	Correlation coefficient between x and U.
sdx	Standard Deviation of x (default will try to extract this from model, it will need to be provided if model is a number rather than a model object).

Details

These functions are all used to do sensitivity analysis on regression models for observational data. Currently it works with linear regression, logistic regression (using glm), and survival regressions (using coxph). All models are of the general form:

y = b1*x + gamma*U + Beta*Z

Where y is the response (or function of the response). The first letter in the triplet at the end of each function name corresponds to the type of response variable, N-normal/numeric (continuous), C-Categorical (binary, logistic regression), and S-Survival (coxph models).

The x variable is the coefficient of interest (usually the treatment variable, or primary predictor of interest). The 2nd letter in the triplet can be C-Categorical (x is 0 or 1, usually control vs. treatment) or N-Numeric, a continuous variable.

The U variable is an unmeasured potential confounder that we believe may be related to y and x. The final letter in the final triplet refers to the type of unmeasured variable that we want to correct for: C-Categorical/binary (0-1, present-absent) or N-Numerical/continuous (in this case it is assumed to be normal with mean 0 and sd 1).

The Z represents additional covariates in the model that are not of primary interest in the sensitivity analysis. It is assumed that Z and U are independent of each other.

For all the functions you specify the potential relationships between U and y (`gamma`, `g0`, and `g1`) and potential relationships between x and U (`p0`, `p1`, `delta`, and `rho`). Then the functions compute a new estimate of b1 (the slope for x, the variable of interest) and its confidence interval given each combination of the relationships between U, y, and x.

Currently only the approximation method by Lin et. al. is available. In the future a simulation method will also be implemented.

Value

An obsSens object (S3) stored as a list with the following elements:

`beta`	A matrix/array with the adjusted slopes for the different conditions.
`lcl`	A matrix/array with the lower confidence intervals for the slopes.
`ucl`	A matrix/array with the upper confidence intervals for the slopes.
`log`	A logical indicating if the values are on the log scale.
`xname`	The 'name' of the x variable of interest
`type`	Character field with the type of y variable, can be 'cat', 'surv', or 'num'.

Note

Note: Currently there are no checks on whether the conditions will be appropriate for the approximation to be close. See the first paper below for the conditions when the approximation is good.

Author(s)

Greg Snow <538280@gmail.com>

References

Lin, DY and Psaty, BM and Kronmal, RA. (1998): Assessing the Sensitivity of Regression Results to Unmeasured Confounders in Observational Studies. Biometrics, 54 (3), Sep, pp. 948-963.

Baer, VL et. als (2007): Do Platelet Transfusions in the NICU Adversely Affect Survival? Analysis of 1600 Thrombocytopenic neonates in a mulihospital healthcare system. Journal of Perinatology, 27, pp. 790-796.

See Also

print.obsSens, summary.obsSens

Examples

```
# Recreate tables from above references

obsSensCCC( log(23.1), log(c(6.9, 77.7)), g0=c(2,6,10),
  p0=seq(0,.5,.1), p1=seq(0,1,.2) )

obsSensSCC( log(1.21), log(c(1.09,1.25)),
  p0=seq(0,.5,.1), p1=seq(0,1,.1), g0=3 )


obsSensCNN( log(1.14), log(c(1.10,1.18)),
  rho=c(0,.5, .75, .85, .9, .95, .98, .99),
  gamma=seq(0,1,.2), sdx=4.5 )
```

---

print.obsSens *Print and Summarize ObsSens objects.*

Description

Print a nice table of the results from a sensitivity analysis on observational data.

Usage

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

## S3 method for class 'obsSens'
summary(object, digits=3, ...)

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

Arguments

x	An obsSens object or summary.obsSens object.
object	An obsSens object.
digits	Passed to format.
...	Additional arguments passed on to print methods for matrix/array.

Details

These functions print and summarize the results of the obsSens functions. The only difference in the print and summary methods is the return value (the information printed to the screen is the same). The `print` function returns a copy of the original object. The `summary` function returns a matrix/array of character strings corresponding to what is printed that can hopefully be used along with other functions (latex, xtable, odfTable, ...) for typesetting results in other formats.

Value

The `print` method returns a copy of the original object. The `summary` method returns a character array/matrix with the coefficients and confidence intervals to be passed to other typesetting/printing functions.

Author(s)

Greg Snow <538280@gmail.com>

Examples

```
# Recreate tables from above references

obsSensCCC( log(23.1), log(c(6.9, 77.7)), g0=c(2,6,10),
  p0=seq(0,.5,.1), p1=seq(0,1,.2) )

obsSensSCC( log(1.21), log(c(1.09,1.25)),
  p0=seq(0,.5,.1), p1=seq(0,1,.1), g0=3 )


obsSensCNN( log(1.14), log(c(1.10,1.18)),
  rho=c(0,.5, .75, .85, .9, .95, .98, .99),
  gamma=seq(0,1,.2), sdx=4.5 )
```

Index

∗ **models**
  obsSens-package, 2
  obsSensCCC, 3
  print.obsSens, 6
∗ **package**
  obsSens-package, 2
∗ **regression**
  obsSens-package, 2
  obsSensCCC, 3
∗ **survival**
  obsSens-package, 2
  obsSensCCC, 3

obsSens *(*obsSens-package*)*, 2
obsSens-package, 2
obsSensCCC, 3
obsSensCCN *(*obsSensCCC*)*, 3
obsSensCNN *(*obsSensCCC*)*, 3
obsSensNCC *(*obsSensCCC*)*, 3
obsSensNCN *(*obsSensCCC*)*, 3
obsSensNNN *(*obsSensCCC*)*, 3
obsSensSCC *(*obsSensCCC*)*, 3
obsSensSCN *(*obsSensCCC*)*, 3
obsSensSNN *(*obsSensCCC*)*, 3

print.obsSens, *6*, 6
print.summary.obsSens *(*print.obsSens*)*, 6

summary.obsSens, *6*
summary.obsSens *(*print.obsSens*)*, 6