Package 'ResourceSelection'

July 22, 2019

Type Package

Title Resource Selection (Probability) Functions for Use-Availability Data

Version 0.3-5

Date 2019-07-22

Author Subhash R. Lele [aut],
Jonah L. Keim [aut],
Peter Solymos [aut, cre] (<https://orcid.org/0000-0001-7337-1740>)

Maintainer Peter Solymos <solymos@ualberta.ca>

Description Resource Selection (Probability) Functions
for use-availability wildlife data
based on weighted distributions as described in
Lele and Keim (2006) <doi:10.1890/0012-9658(2006)87%5B3021:WDAEOR%5D2.0.CO;2>,
Lele (2009) <doi:10.2193/2007-535>,
and Solymos & Lele (2016) <doi:10.1111/2041-210X.12432>.

Depends R (>= 2.13.0)

Imports MASS, pbapply, Matrix

URL https://github.com/psolymos/ResourceSelection

BugReports https://github.com/psolymos/ResourceSelection/issues

License GPL-2

LazyLoad yes

LazyData true

NeedsCompilation no

Repository CRAN

Date/Publication 2019-07-22 15:20:02 UTC

R topics documented:

`ResourceSelection-package`
Resource Selection (Probability) Functions for Use-Availability Data

---

Description

Resource Selection (Probability) Functions for use-availability wildlife data based on weighted distributions as described in Lele and Keim (2006), Lele (2009), and Solymos & Lele (2016).

Details

`rsf`: Resource Selection Functions (RSF)

`rspf`: Resource Selection Probability Functions (RSPF)

`hoslem.test`: Hosmer-Lemeshow Goodness of Fit (GOF) Test

Visual summaries: `kdepairs` for 2D scatterplots and `mep` for marginal effect plots based on fitted model objects.

Author(s)

Subhash R. Lele, Jonah L. Keim, Peter Solymos

Maintainer: Peter Solymos <solymos@ualberta.ca>

References

Lele, S.R. (2009) A new method for estimation of resource selection probability function. Journal of Wildlife Management 73, 122–127. <doi:10.2193/2007-535>

Lele, S. R. & Keim, J. L. (2006) Weighted distributions and estimation of resource selection probability function. Ecology 87, 3021–3028. <doi:10.1890/0012-9658(2006)87

Solymos, P. & Lele, S. R. (2016) Revisiting resource selection probability functions and single-visit methods: clarification and extensions. Methods in Ecology and Evolution 7, 196–205. <doi:10.1111/2041-210X.12432>

See Also

`rsf`, `rspf`, `kdepairs`, `mep`, `hoslem.test`

CAIC *Consistent AIC*

Description

Consistent AIC

Usage

```
CAIC(object, ..., alpha)
## Default S3 method:
CAIC(object, ..., alpha)
CAICtable(object, ..., alpha)
```

Arguments

object	A fitted model object.
...	More fitted model objects.
alpha	Weight factor between 0 and 1 (see Details). Default value is 0.5.

Details

CAIC = alpha * AIC + (1 - alpha) * BIC

Value

Atomic vector if only one input object provided, a data frame similar to what is returned by AIC and BIC if there are more than one input objects.

CAICtable returns a data frame with delta CAIC (dCAIC = CAIC - min(CAIC)) and CAIC weights (wCAIC = exp(-0.5 dCAIC_i) / sum(exp(-0.5 dCAIC_i))) where i = 1,...,m are candidate models.

Author(s)

Subhash Lele and Peter Solymos

References

Bozdogan, H. 1987. Model selection and Akaike's information criterion (AIC): the general theory and its analytical extensions. Psychometrika, 52, 345-370.

Taper, M. 2004. Model identification from many candidates. In: Taper, M. and Lele, S. R. (eds), The Nature of Scientific Evidence: Statistical, Philosophical, and Empirical Considerations. The University of Chicago Press, Chicago, IL, 567 pp.

See Also

AIC, BIC

Examples

```
## compare some random models
y <- rnorm(10)
a <- lm(y ~ runif(10))
b <- lm(y ~ runif(10))

0.5*(AIC(a) + BIC(a))
CAIC(a)
AIC(a)
CAIC(a, alpha=1)
BIC(a)
CAIC(a, alpha=0)

CAIC(a, b)
CAIC(a, b, alpha=0.2)

CAICtable(a, b, alpha=1)

## you can use global option
## useful when inside of xv or bootstrap
## no need for extra argument
getOption("CAIC_alpha")
op <- options(CAIC_alpha = 0.2)
getOption("CAIC_alpha")
CAIC(a,b)
options(op)
getOption("CAIC_alpha")
```

---

goats *Mountain Goats Data Set*

Description

GPS collar data of mountain goats (*Oreamnos americanus*) from Lele and Keim (2006).

Usage

```
data(goats)
```

Format

A data frame with 19014 observations on the following 8 variables.

`STATUS` a numeric vector, 1: used, 0: available

`ID` a numeric vector, individuals

`ELEVATION` a numeric vector (m)

`SLOPE` a numeric vector (degrees, steep)

`ET` a numeric vector, access to escape terrain (distance from steep slopes, m)

`ASPECT` a numeric vector (degrees)

`HLI` a numeric vector, heat load index (0-1)

`TASP` a numeric vector, transformed aspect

Details

Mountain goat telemetry data were collected in the Coast Mountains of northwest British Columbia, Canada, as described in Lele and Keim (2006).

Source

Ecological Archives E087-181-S1, `http://www.esapubs.org/archive/ecol/E087/181/`

References

Lele, S. R. & Keim, J. L. (2006) Weighted distributions and estimation of resource selection probability functions. Ecology 87, 3021–3028.

Examples

```
data(goats)
str(goats)
summary(goats)

## Not run:
goats$exp.HLI <- exp(goats$HLI)
goats$sin.SLOPE <- sin(pi * goats$SLOPE / 180)
goats$ELEVATION <- scale(goats$ELEVATION)
goats$ET <- scale(goats$ET)
goats$TASP <- scale(goats$TASP)
m1 <- rspf(STATUS ~ TASP + sin.SLOPE + ELEVATION, goats, m=0, B = 99)
m2 <- rspf(STATUS ~ TASP + ELEVATION, goats, m=0, B = 99)
summary(m1)
summary(m2)
AIC(m1, m2)
plot(m1)

## End(Not run)
```

---

hoslem.test *Hosmer-Lemeshow Goodness of Fit (GOF) Test*

Description

Hosmer-Lemeshow Goodness of Fit (GOF) Test.

Usage

```
hoslem.test(x, y, g = 10)
```

Arguments

x	a numeric vector of observations, binary (0/1).
y	expected values.
g	number of bins to use to calculate quantiles.

Details

The Hosmer-Lemeshow test is a statistical test for goodness of fit for logistic regression models.

Value

A list with class "htest" containing the following components:

statistic	the value of the chi-squared test statistic, (sum((observed -expected)^2 / expected)).
parameter	the degrees of freedom of the approximate chi-squared distribution of the test statistic (g -2).
p.value	the p-value for the test.
method	a character string indicating the type of test performed.
data.name	a character string giving the name(s) of the data.
observed	the observed frequencies in a g-by-2 contingency table.
expected	the expected frequencies in a g-by-2 contingency table.

Author(s)

Peter Solymos by adapting code pieces from R help mailing list

References

Hosmer D W, Lemeshow S 2000. Applied Logistic Regression. New York, USA: John Wiley and Sons.

Examples

```
set.seed(123)
n <- 500
x <- rnorm(n)
y <- rbinom(n, 1, plogis(0.1 + 0.5*x))
m <- glm(y ~ x, family=binomial)
hoslem.test(m$y, fitted(m))
```

kdepairs *Scatterplot Matrix with 2D Kernel Density*

Description

Scatterplot matrix with 2D kernel density.

Usage

```
kdepairs(x, ...)

## Default S3 method:
kdepairs(x, n=25, density=TRUE, contour=TRUE, ...)

## S3 method for class 'rsf'
kdepairs(x, n=25, density=TRUE, contour=TRUE, ...)
```

Arguments

x	a matrix or data frame (or a fitted model object of class "rsf" or "rspf").
n	number of bins to be used in kernel density estimation.
density	logical, if shades corresponding to densities should be plotted.
contour	logical, if contour on top of shades should be plotted.
...	other possible arguments passed to pairs.

Value

Produces a scatterplot matrix with histograms in diagonal, 2D kernel density estimates and contours in the lower half and bivariate scatterplots with lowess smooth curves and Pearson correlation values in the upper half as a side effect. Returns NULL invisibly.

Author(s)

Peter Solymos

See Also

pairs, lowess, kde2d, contour

Examples

```
kdepairs(iris[1:4])
```

`makeUsedAvail` *Make a Used-Available Data Frame*

Description

Make a used-available data frame from a presence-absence type data.

Usage

```
makeUsedAvail(x, ...)

## Default S3 method:
makeUsedAvail(x, y, ...)

## S3 method for class 'formula'
makeUsedAvail(formula, data = parent.frame(), ...)
```

Arguments

`x`	a matrix or data frame.
`y`	a vector with 0/1 entries, 1s are taken as used observations.
`formula`	two sided model formula of the form `y ~ x`.
`data`	data.
`...`	other arguments.

Value

The function returns a data frame, where used and available portions of the input data are bound on top of each other, the first column refers to `y`, where used (1) and available (0) locations are indicated different from the input values. All locations in the input data are treated as available (0), while only nonzero observations in `y` are treated as used (1).

Author(s)

Peter Solymos

Examples

```
(x <- data.frame(species=rep(1:0,each=4), var1=1:8, var2=11:18))
makeUsedAvail(species ~ var1 + var2, x)
```

mep *Marginal Effect Plots*

Description

Scatterplot of marginal effects based on fitted model objects.

Usage

```
mep(object, ...)

## Default S3 method:
mep(object, which=NULL, link=NULL,
    level=0.95, unique=10, n=25, minbucket=5, digits=4,
    col.points, col.lines=c(2, 2), pch=19, lty=c(1, 2), lwd=c(2,2),
    ask, subset=NULL, ylab, ...)
```

Arguments

`object`	a fitted model object.
`which`	numeric, logical, or character. Indices for the variables in the model frame if only one or a subset is desired.
`link`	character accepted by `make.link`, optional argument to determine scaling. It is guessed when value cannot be determined based on `family(object)$link` (see Details).
`level`	numeric [0, 1], the confidence level required.
`unique, digits`	numeric, the number of unique points above which bins are used. If the number of unique values is less than or equal to this number, unique values are used without binning. Unique values are subject to rounding to `digits`.
`n, minbucket`	number of bins (`n`) to be used in quantile estimation when variable is not treated as unique points. `minbucket` is the minimum number of points within each bin. `n` is decreased until `minbucket` condition is satisfied.
`col.points, pch`	color and type of points to be plotted.
`col.lines, lty, lwd`	color, type, and width of quantile lines to be plotted. The 1st value correspond to the median, the 2nd value to the upper and lower quantiles, respectively.
`ask`	logical. If `TRUE`, the user is asked before each plot, see `par(ask=.)`.
`subset`	an optional vector specifying a subset of the data to be used for plotting.
`ylab`	character or expression, optional y axis label.
`...`	other possible arguments passed to graphical functions.

Details

The input object must have a `fitted` and `model.frame` method, and possibly a well identifiable family/link component (`family(object)$link`). In the absence of family/link information, the range of the fitted value will be used to guess the scaling (identity, log, or logit) unless directly supplied via the `link` argument.

Fitted values (f(x) = f(x_1,...,x_i,...,x_p); i = 1,...,p) are plotted against x_i. The visual display is determined by the type of x_i (un-ordered factor, ordered factor, unique numeric values, binned numeric values). For each unique vale or bin, the median and confidence intervals (quantiles corresponding to `level`) of f(x) are calculated. Binned values are smoothed by `lowess` unless `n` < 3.

Jitter is added to factor and unique value types. Jitter is calculated based on kernel `density`.

The model frame includes the response variable as well. Plotting f(x) as a function of the observations might be a useful visualization too to indicate goodness of fit or the lack of it.

Value

The produces one or several marginal plots as a side effect. Returns a list of quantiles of fitted values corresponding to binned/unique values of variables in the input object.

Author(s)

Peter Solymos and Subhash Lele

References

Avgar, T., Lele, S. R., Keim, J. L. & Boyce, M. S. (2017) Relative Selection Strength: Quantifying effect size in habitat- and step-selection inference. Ecology and Evolution 7, 5322–5330.

See Also

`kdepairs` for 2D kernel density estimates and contours.

`fitted` for fitted values and `model.frame` for model frames.

`density` and `lowess` for smoothing.

Examples

```
data(goats)
goats$ELEVATION <- goats$ELEVATION/1000
goats$TASPc <- cut(goats$TASP, 3, ordered_result=FALSE)
goats$SLOPEc <- cut(goats$SLOPE, 3, ordered_result=TRUE)

fit <- rspf(STATUS ~ TASPc + SLOPEc + ELEVATION + I(ELEVATION^2), goats, m=0, B=0)

op <- par(mfrow=c(2,2))
mep(fit, which=1:4)#, subset=sample.int(nrow(goats), 10^4))
par(op)
```

rsf *Resource Selection (Probability) Functions for Use-Availability Data*

Description

Resource Selection (Probability) Functions for use-availability wildlife data as described in Lele and Keim (2006) and Lele (2009).

Usage

```
rsf(formula, data, m, B = 99, inits, method = "Nelder-Mead",
control, model = TRUE, x = FALSE, ...)

rspf(formula, data, m, B = 99, link = "logit", inits,
method = "Nelder-Mead", control, model = TRUE, x = FALSE, ...)

rsf.fit(X, Y, m, link = "logit", B = 99,
inits, method = "Nelder-Mead", control, ...)

rsf.null(Y, m, inits, ...)
```

Arguments

formula	two sided model formula of the form y ~ x, where y is a vector of observations, x is the set of covariates.
m	argument describing the matching of use and available points, see Details.
data	data.
B	number of bootstrap iterations to make.
link	character, type of link function to be used.
inits	initial values, optional.
method	method to be used in optim for numerical optimization.
control	control options for optim.
model	a logical value indicating whether model frame should be included as a component of the returned value
x	logical values indicating whether the model matrix used in the fitting process should be returned as components of the returned value.
Y	vector of observations.
X	covariate matrix.
...	other arguments passed to the functions.

Details

The `rsf` function fits the Exponential Resource Selection Function (RSF) model to presence only data.

The `rspf` function fits the Resource Selection Probability Function (RSPF) model to presence only data Link function `"logit"`, `"cloglog"`, and `"probit"` can be specified via the `link` argument.

The `rsf.fit` is the workhorse behind the two functions. `link="log"` leads to Exponential RSF.

The `rsf.null` function fits the 'no selection' version of the Exponential Resource Selection Function (RSF) model to presence only data.

LHS of the `formula` data must be binary, ones indicating used locations, while zeros indicating available location.

All available points are used for each use points if `m=0` (global availability). If `m` is a single value, e.g. `m=5`, it is assumed that available data points are grouped in batches of 5, e.g. with IDs `c(1,2)` for used point locations and `c(1,1,1,1,1,2,2,2,2,2)` for available locations (local availability, matched use-available design). Similarly, a vector of matching IDs can also be provided, e.g. `c(1,2,1,1,1,1,1,2,2,2,2,2)` by combining the above two. This potentially could allow for unbalanced matching (e.g. `c(1,2,1,1,1,2,2,2,2,2,2,2)`) and for easier subsetting of the data, but comes with an increased computing time. Note, the response in the LHS of the formula should be coded as `c(1,1,0,0,0,0,0,0,0,0,0,0)` for all of the above examples. When `m` is defined as a mapping vector or the value is 0, the order of course does not matter. However, ordering matters when `m` is constant because that implies a certain structure.

For model description and estimation details, see Lele and Keim (2006), Lele (2009), and Solymos and Lele (2016).

Value

A list with class `"rsf"`, `"rsf.null"`, or `"rspf"` containing the following components:

`call`	the matched call.
`y`	vector from LHS of the formula.
`coefficients`	a named vector of coefficients.
`std.error`	a named vector of standard errors for the coefficients.
`loglik`	the maximized pseudo log-likelihood according to Lele 2009.
`results`	`optim` results.
`link`	character, value of the link function used.
`control`	control parameters for `optim`.
`inits`	initial values used in optimization.
`m`	value of the `m` argument with possibly matched use-available design.
`np`	number of active parameters.
`fitted.values`	vector of fitted values. These are relative selection values for RSF models, and probability of selection for RSPF models.

nobs	number of used locations.
bootstrap	component to store bootstrap results if B>0.
converged	logical, indicating convergence of the optimization.
formula	the formula supplied.
terms	the terms object used.
levels	a record of the levels of the factors used in fitting.
contrasts	the contrasts used.
model	if requested, the model frame.
x	if requested, the model matrix.

Author(s)

Subhash R. Lele, Jonah L. Keim, Peter Solymos

References

Lele, S.R. (2009) A new method for estimation of resource selection probability function. Journal of Wildlife Management 73, 122–127.

Lele, S. R. & Keim, J. L. (2006) Weighted distributions and estimation of resource selection probability functions. Ecology 87, 3021–3028.

Solymos, P. & Lele, S. R. (2016) Revisiting resource selection probability functions and single-visit methods: clarification and extensions. Methods in Ecology and Evolution 7, 196–205.

Examples

```
## --- Simulated data example ---

## settings
n.used <- 1000
m <- 10
n <- n.used * m
set.seed(1234)
x <- data.frame(x1=rnorm(n), x2=runif(n))
cfs <- c(1.5,-1,0.5)
## fitting Exponential RSF model
dat1 <- simulateUsedAvail(x, cfs, n.used, m, link="log")
m1 <- rsf(status ~ .-status, dat1, m=0, B=0)
summary(m1)
## fitting Logistic RSPF model
dat2 <- simulateUsedAvail(x, cfs, n.used, m, link="logit")
m2 <- rspf(status ~ .-status, dat2, m=0, B=0)
summary(m2)

## --- Real data analysis from Lele & Keim 2006 ---

## Not run:
goats$exp.HLI <- exp(goats$HLI)
goats$sin.SLOPE <- sin(pi * goats$SLOPE / 180)
```

```
goats$ELEVATION <- scale(goats$ELEVATION)
goats$ET <- scale(goats$ET)
goats$TASP <- scale(goats$TASP)

## Fit two RSPF models:
## global availability (m=0) and bootstrap (B=99)
m1 <- rspf(STATUS ~ TASP + sin.SLOPE + ELEVATION, goats, m=0, B = 99)
m2 <- rspf(STATUS ~ TASP + ELEVATION, goats, m=0, B = 99)

## Inspect the summaries
summary(m1)
summary(m2)

## Compare models: looks like m1 is better supported
CAIC(m1, m2)

## Visualize the relationships
plot(m1)
mep(m1) # marginal effects similar to plot but with CIs
kdepairs(m1) # 2D kernel density estimates
plot(m2)
kdepairs(m2)
mep(m2)

## fit and compare to null RSF model (not available for RSPF)
m3 <- rsf(STATUS ~ TASP + ELEVATION, goats, m=0, B = 0)
m4 <- rsf.null(Y=goats$STATUS, m=0)
CAIC(m3, m4)

## End(Not run)
```

`simulateUsedAvail` *Simulate Used-Available Data*

Description

Simulates used-available data.

Usage

```
simulateUsedAvail(data, parms, n.used, m, link="logit")
```

Arguments

`data`	a matrix or data frame.
`parms`	coefficients corresponding to the columns of the design matrix derived as `model.matrix(~.,data)`.
`n.used, m`	number of used points (`n.used`) and number of available points for each (`m`).
`link`	character, the type of link function to be used.

Value

A used-available data frame.

Author(s)

Subhash Lele, Peter Solymos

Examples

```
n.used <- 1000
m <- 10
n <- n.used * m
set.seed(1234)
x <- data.frame(x1=rnorm(n), x2=runif(n))
cfs <- c(1.5,-1,0.5)
dat1 <- simulateUsedAvail(x, cfs, n.used, m, link="log")
str(dat1)
dat2 <- simulateUsedAvail(x, cfs, n.used, m, link="logit")
str(dat2)
```

---

sindex *Weighted relative suitability index*

Description

Calculates weighted relative suitability index.

Usage

```
sindex(y, x)
wrsi(y, x)
```

Arguments

y	matrix of observations for `sindex`, vector of observations for `wrsi`.
x	a matrix of proportions (i.e. the values 0 and 1 should have consistent meaning across the columns, often through a unit sum constraint).

Value

`wrsi` returns a data frame (class 'wrsi') with the following columns:

`WRSI` weighted relative suitability index, range (0- Inf).

`zWRSI` log of `WRSI` (z-transformed), range (-Inf, Inf).

`rWRSI` inverse Fisher z-transformed `zWRSI`, range (-1, 1).

`Pused` **and** `Pavail` total proportion of used (`y > 0`) and available of each feature (column) in `x`.

`Pw` weighted proportions from `y`.

`u` **and** `a` used and available totals for each feature (column) in `x`.

`sindex` returns a data frame (class 'sindex') with one column for each species, and one row for each feature (column) in `x`. Cell values are inverse Fisher z-transformed (`zWRSI`) indices.

Author(s)

Peter Solymos <solymos@ualberta.ca>

Examples

```
## --- habitat composition matrix
set.seed(1234)
n <- 1000 # sample size
k <- 5 # habitat classes
s <- runif(n, 1, 5)
p <- plogis(rnorm(n*k, 0, rep(s, k)))
p <- p*t(replicate(n, sample(c(10,4,2,1,1))))
x <- p / rowSums(p)
summary(x)
summary(rowSums(x))

## --- observations
## expected abundance in each habitat class
lam <- c(0.8, 0.6, 0.5, 0.4, 0.1)*1
## sample x habitat level abundances
yy <- t(sapply(seq_len(n), function(i) {
    ## intercept and modifier combined
    rpois(k, (x[i,]*lam))
    }))
## total: sum over habitat classes
## this is what we observe
y <- rowSums(yy)
colSums(yy)
table(y)

## --- wrsi calculations
(w <- wrsi(y, x))
op <- par(mfrow=c(1,2))
## habitat level observations are unknown
plot(lam, colSums(yy) / sum(yy), type="b")
## this is approximated by the wrsi
plot(lam, w$rWRSI, type="b")
abline(h=0, lty=2)
par(op)

## --- sindex calculations for multiple species
y2 <- cbind(Spp1=y, Spp2=rev(y), Spp3=sample(y))
(w2 <- sindex(y2, x))
heatmap(t(as.matrix(w2)), scale="none")
```