Package 'dominanceanalysis'

February 5, 2024

Title Dominance Analysis

Date 2024-02-04

Encoding UTF-8

Description Dominance analysis is a method that allows to compare the relative importance of predictors in multiple regression models: ordinary least squares, generalized linear models, hierarchical linear models, beta regression and dynamic linear models. The main principles and methods of dominance analysis are described in Budescu, D. V. (1993) <doi:10.1037/0033-2909.114.3.542> and Azen, R., & Budescu, D. V. (2003) <doi:10.1037/1082-989X.8.2.129> for ordinary least squares regression. Subsequently, the extensions for multivariate regression, logistic regression and hierarchical linear models were described in Azen, R., & Budescu, D. V. (2006) <doi:10.3102/10769986031002157>, Azen, R., & Traxel, N. (2009) <doi:10.3102/1076998609332754> and Luo, W., & Azen, R. (2013) <doi:10.3102/1076998612458319>, respectively.

Version 2.1.0

Depends R (>= 4.0.0)

License GPL-2

LazyData true

Imports methods, stats, ggplot2

Suggests lme4, boot, testthat, car, covr, knitr,rmarkdown,pscl, dynlm, reshape2, betareg, performance

RoxygenNote 7.2.3

VignetteBuilder knitr

NeedsCompilation no

Author Claudio Bustos Navarrete [aut, cre, cph] (<https://orcid.org/0000-0003-3478-9858>), Filipa Coutinho Soares [aut] (<https://orcid.org/0000-0002-7579-2538>)

Maintainer Claudio Bustos Navarrete <clbustos@gmail.com>

Repository CRAN

Date/Publication 2024-02-05 20:50:17 UTC

R topics documented:

`dominanceanalysis-package`

Dominance analysis for general, generalized and mixed linear models

---

Description

The dominanceanalysis package allows to perform the dominance analysis for multiple regression models, such as OLS (univariate and multivariate), GLM and HLM. The dominance analysis on this package is performed by `dominanceAnalysis` function. To perform bootstrap procedures you should use `bootDominanceAnalysis` function. For both, standard `print` and `summary` functions are provided.

Main Features

- Provides complete, conditional and general dominance analysis for lm (univariate and multivariate), lmer and glm (family=binomial) models.
- Covariance / correlation matrixes could be used as input for OLS dominance analysis, using `lmWithCov` and `mlmWithCov` methods, respectively.
- Multiple criteria can be used as fit indices, which is useful especially for HLM.

About Dominance Analysis

Dominance analysis is a method developed to evaluate the importance of each predictor in the selected regression model: "one predictor is 'more important than another' if it contributes more to the prediction of the criterion than does its competitor at a given level of analysis." (Azen & Budescu, 2003, p.133).

The original method was developed for OLS regression (Budescu, 1993). Later, several definitions of dominance and bootstrap procedures were provided by Azen & Budescu (2003), as well as adaptations to Generalized Linear Models (Azen & Traxel, 2009) and Hierarchical Linear Models (Luo & Azen, 2013).

Author(s)

Claudio Bustos <clbustos@gmail.com>, Filipa Coutinho Soares (documentation)

References

- Budescu, D. V. (1993). Dominance analysis: A new approach to the problem of relative importance of predictors in multiple regression. Psychological Bulletin, 114(3), 542-551. doi:10.1037/0033-2909.114.3.542
- Azen, R., & Budescu, D. V. (2003). The dominance analysis approach for comparing predictors in multiple regression. Psychological Methods, 8(2), 129-148. doi:10.1037/1082-989X.8.2.129
- Azen, R., & Budescu, D. V. (2006). Comparing Predictors in Multivariate Regression Models: An Extension of Dominance Analysis. Journal of Educational and Behavioral Statistics, 31(2), 157-180. doi:10.3102/10769986031002157
- Azen, R., & Traxel, N. (2009). Using Dominance Analysis to Determine Predictor Importance in Logistic Regression. Journal of Educational and Behavioral Statistics, 34(3), 319-347. doi:10.3102/1076998609332754
- Luo, W., & Azen, R. (2013). Determining Predictor Importance in Hierarchical Linear Models Using Dominance Analysis. Journal of Educational and Behavioral Statistics, 38(1), 3-31. doi:10.3102/1076998612458319

See Also

`dominanceAnalysis` , `bootDominanceAnalysis`

Examples

```
# Basic dominance analysis

data(longley)
lm.1<-lm(Employed~.,longley)
da<-dominanceAnalysis(lm.1)
print(da)
summary(da)
plot(da,which.graph='complete')
plot(da,which.graph='conditional')
plot(da,which.graph='general')

# Dominance analysis for HLM

library(lme4)
x1<-rnorm(1000)
x2<-rnorm(1000)
g<-gl(10,100)
g.x<-rnorm(10)[g]
y<-2*x1+x2+g.x+rnorm(1000,sd=0.5)
lmm1<-lmer(y~x1+x2+(1|g))
lmm0<-lmer(y~(1|g))
da.lmm<-dominanceAnalysis(lmm1, null.model=lmm0)
print(da.lmm)
summary(da.lmm)


# GLM analysis

x1<-rnorm(1000)
x2<-rnorm(1000)
x3<-rnorm(1000)
y<-runif(1000)<(1/(1+exp(-(2*x1+x2+1.5*x3))))
glm.1<-glm(y~x1+x2+x3,family="binomial")
da.glm<-dominanceAnalysis(glm.1)
print(da.glm)
summary(da.glm)

# Bootstrap procedure


da.boot<-bootDominanceAnalysis(lm.1,R=1000)
summary(da.boot)

da.glm.boot<-bootDominanceAnalysis(glm.1,R=200)
summary(da.glm.boot)
```

`averageContribution` *Retrieve average contribution of each predictor in a dominance analysis.*

Description

Retrieve the average contribution for each predictor. Is calculated averaging all contribution by level. The average contribution defines general dominance.

Usage

```
averageContribution(da.object, fit.functions = NULL)
```

Arguments

da.object	dominanceAnalysis object
fit.functions	name of the fit indices to retrieve. If NULL, all fit indices will be retrieved

Value

a list. Key corresponds to fit-index and the value is vector, with average contribution for each variable

See Also

Other retrieval methods: contributionByLevel(), dominanceBriefing(), dominanceMatrix(), getFits()

Examples

```
data(longley)
da.longley<-dominanceAnalysis(lm(Employed~.,longley))
averageContribution(da.longley)
```

---

bootAverageDominanceAnalysis

Bootstrap Average Values for Dominance Analysis

Description

Bootstrap average values and corresponding standard errors for each predictor in the dominance analysis. These values are used for assessing general dominance.

Usage

```
bootAverageDominanceAnalysis(
  x,
  R,
  constants = c(),
  terms = NULL,
  fit.functions = "default",
  null.model = NULL,
  ...
)
```

Arguments

x	A model object, like 'lm', 'glm', or 'lmer'.
R	An integer indicating the number of bootstrap resamples to be performed.
constants	A character vector specifying predictors that should remain constant in the bootstrap analysis. Default is an empty vector.
terms	An optional vector of terms (predictors) to be analyzed. If NULL, terms are obtained from the model. Default is NULL.
fit.functions	A vector of functions providing fit indices for the model. See 'fit.functions' parameter in 'dominanceAnalysis' function.
null.model	An optional model object specifying the null model for linear mixed models, used as a baseline for testing submodels. Default is NULL.
...	Additional arguments passed to 'dominanceAnalysis' method

Details

Use `summary()` to obtain a nicely formatted `data.frame` object.

Value

An object of class 'bootAverageDominanceAnalysis' containing: -

boot	The results of the bootstrap analysis in a `boot` object.
preds	The predictors analyzed
fit.functions	The fit functions used in the analysis
R	The number of bootstrap resamples
eg	expanded grid of predictors by fit functions
terms	The terms analyzed

See Also

dominanceAnalysis, boot

Examples

```
lm.1 <- lm(Employed ~ ., longley)
da.ave.boot <- bootAverageDominanceAnalysis(lm.1, R = 1000)
summary(da.ave.boot)
```

`bootDominanceAnalysis` *Bootstrap Analysis for Dominance Analysis*

Description

Implements a bootstrap procedure as presented by Azen and Budescu (2003). Provides the expected level of dominance of predictor X_i over X_j, as the degree to which the pattern found in the sample is reproduced in the bootstrap samples.

Usage

```
bootDominanceAnalysis(
  x,
  R,
  constants = c(),
  terms = NULL,
  fit.functions = "default",
  null.model = NULL,
  ...
)
```

Arguments

`x`	An object of class `lm`, `glm`, or `lmer`.
`R`	The number of bootstrap resamples.
`constants`	A vector of predictors to remain unchanged between models, i.e., variables not subjected to bootstrap analysis.
`terms`	A vector of terms to be analyzed. By default, terms are obtained from the model.
`fit.functions`	A list of functions providing fit indices for the model. Refer to `fit.functions` parameter in `dominanceAnalysis` function.
`null.model`	Applicable only for linear mixed models. It refers to the null model against which to test the submodels, i.e., only random effects, without any fixed effects.
`...`	Additional arguments provided to `lm` or `lmer` (not implemented yet).

Details

Use `summary()` to obtain a nicely formatted `data.frame`.

Value

An object of class `bootDominanceAnalysis` containing:

`boot`	The results of the bootstrap analysis.
`preds`	The predictors analyzed.
`fit.functions`	The fit functions used in the analysis.

c.names	A vector where each value represents the name of a specific dominance analysis result. Names are prefixed with the type of dominance (complete, conditional, or general), and the fit function used, followed by the names of the first and second predictors involved in the comparison.
m.names	Names of each one the predictor pairs.
terms	The terms analyzed.
R	The number of bootstrap resamples.

Examples

```
lm.1 <- lm(Employed ~ ., longley)
da.boot <- bootDominanceAnalysis(lm.1, R = 1000)
summary(da.boot)
```

---

contributionByLevel *Retrieve average contribution by level for each predictor*

Description

Retrieve the average contribution by level for each predictor in a dominance analysis. The average contribution defines conditional dominance.

Usage

```
contributionByLevel(da.object, fit.functions = NULL)
```

Arguments

da.object	dominanceAnalysis object
fit.functions	name of the fit indices to retrieve. If NULL, all fit indices will be retrieved

Value

a list. Key corresponds to fit-index and the value is a matrix, with contribution of each variable by level

See Also

Other retrieval methods: averageContribution(), dominanceBriefing(), dominanceMatrix(), getFits()

Examples

```
data(longley)
da.longley<-dominanceAnalysis(lm(Employed~.,longley))
contributionByLevel(da.longley)
```

`da.betareg.fit` *Provides fit indices for betareg models.*

Description

Note that the Nagelkerke and Estrella coefficients are designed for discrete dependent variables and thus cannot be used in this context. Instead, the Cox and Snell coefficient is recommended, along with the pseudo-R^2. It is worth noting that McFadden's index may produce negative values and should be avoided.

Usage

```
da.betareg.fit(original.model, newdata = NULL, ...)
```

Arguments

`original.model`	Original fitted model
`newdata`	Data used in update statement
`...`	ignored

Value

A function described by using-fit-indices. You could retrieve following indices:

`r2.pseudo` Provided by betareg by default

`r2.m` McFadden(1974)

`r2.cs` Cox and Snell(1989).

References

- Cox, D. R., & Snell, E. J. (1989). The analysis of binary data (2nd ed.). London, UK: Chapman and Hall.
- Estrella, A. (1998). A new measure of fit for equations with dichotomous dependent variables. Journal of Business & Economic Statistics, 16(2), 198-205. doi: 10.1080/07350015.1998.10524753.
- McFadden, D. (1974). Conditional logit analysis of qualitative choice behavior. In P. Zarembka (Ed.), Frontiers in econometrics (pp. 104-142). New York, NY: Academic Press.
- Shou, Y., & Smithson, M. (2015). Evaluating Predictors of Dispersion:A Comparison of Dominance Analysis and Bayesian Model Averaging. Psychometrika, 80(1), 236-256.

See Also

Other fit indices: `da.clm.fit()`, `da.dynlm.fit()`, `da.glm.fit()`, `da.lm.fit()`, `da.lmWithCov.fit()`, `da.lmerMod.fit()`, `da.mlmWithCov.fit()`

da.clm.fit — *Provides fit indices for ordinal regression models, based on the Nagelkerke (1991) method.*

Description

Provides fit indices for ordinal regression models, based on the Nagelkerke (1991) method.

Usage

```
da.clm.fit(original.model, newdata = NULL, ...)
```

Arguments

original.model	Original fitted model
newdata	Data used in update statement
...	ignored

Value

A function described by using-fit-indices description for interface. You could retrieve r2.n index, corresponding to Nagelkerke method.

References

- Nagelkerke, N. J. D. (1991). A Note on a General Definition of the Coefficient of Determination. Biometrika, 78(3), 691-692. doi:10.1093/biomet/78.3.691

See Also

Other fit indices: da.betareg.fit(), da.dynlm.fit(), da.glm.fit(), da.lm.fit(), da.lmWithCov.fit(), da.lmerMod.fit(), da.mlmWithCov.fit()

da.dynlm.fit — *Provides coefficient of determination for* dynlm *models.*

Description

Uses R^2 (coefficient of determination) as fit index

Usage

```
da.dynlm.fit(original.model, newdata = NULL, ...)
```

Arguments

original.model	Original fitted model
newdata	Data used in update statement
...	ignored

Value

A function described by using-fit-indices description for interface

See Also

Other fit indices: da.betareg.fit(), da.clm.fit(), da.glm.fit(), da.lm.fit(), da.lmWithCov.fit(), da.lmerMod.fit(), da.mlmWithCov.fit()

---

da.glm.fit *Provides fit indices for GLM models.*

Description

These functions are only available for logistic regression models and are based on the work of Azen and Traxel (2009).

Usage

```
da.glm.fit(original.model, newdata = NULL, ...)
```

Arguments

original.model	Original fitted model
newdata	Data used in update statement
...	ignored

Details

Check daRawResults.

Value

A function described by using-fit-indices. You could retrieve the following indices:

r2.m McFadden(1974)

r2.cs Cox and Snell(1989). Use with caution, because don't have 1 as upper bound

r2.n Nagelkerke(1991), that corrects the upper bound of Cox and Snell(1989) index

r2.e Estrella(1998)

References

- Azen, R. and Traxel, N. (2009). Using Dominance Analysis to Determine Predictor Importance in Logistic Regression. *Journal of Educational and Behavioral Statistics, 34* (3), 319-347. doi:10.3102/1076998609332754.
- Nagelkerke, N. J. D. (1991). A note on a general definition of the coefficient of determination. *Biometrika, 78*(3), 691-692. doi:10.1093/biomet/78.3.691.
- Cox, D. R., & Snell, E. J. (1989). The analysis of binary data (2nd ed.). London, UK: Chapman and Hall.
- Estrella, A. (1998). A new measure of fit for equations with dichotomous dependent variables. Journal of Business & Economic Statistics, 16(2), 198-205. doi: 10.1080/07350015.1998.10524753
- McFadden, D. (1974). Conditional logit analysis of qualitative choice behavior. In P. Zarembka (Ed.), Frontiers in econometrics (pp. 104-142). New York, NY: Academic Press.

See Also

Other fit indices: `da.betareg.fit()`, `da.clm.fit()`, `da.dynlm.fit()`, `da.lm.fit()`, `da.lmWithCov.fit()`, `da.lmerMod.fit()`, `da.mlmWithCov.fit()`

Examples

```
x1<-rnorm(1000)
x2<-rnorm(1000)
x3<-rnorm(1000)
y<-factor(runif(1000) > exp(x1+x2+x3)/(1+exp(x1+x2+x3)))
df.1=data.frame(x1,x2,x3,y)
glm.1<-glm(y~x1+x2+x3,data=df.1,family=binomial)
da.glm.fit(original.model=glm.1)("names")
da.glm.fit(original.model=glm.1)(y~x1)
```

`da.lm.fit` *Provides coefficient of determination for* `lm` *models.*

Description

Uses R^2 (coefficient of determination) as fit index

Usage

```
da.lm.fit(original.model, newdata = NULL, ...)
```

Arguments

`original.model`	Original fitted model
`newdata`	Data used in update statement
`...`	ignored

Value

A function described by using-fit-indices description for interface. You could retrieve `r2` index.

See Also

Other fit indices: `da.betareg.fit()`, `da.clm.fit()`, `da.dynlm.fit()`, `da.glm.fit()`, `da.lmWithCov.fit()`, `da.lmerMod.fit()`, `da.mlmWithCov.fit()`

Examples

```
x1<-rnorm(1000)
x2<-rnorm(1000)
y <-x1+x2+rnorm(1000)
df.1=data.frame(y=y,x1=x1,x2=x2)
lm.1<-lm(y~x1+x2)
da.lm.fit(lm.1)("names")
da.lm.fit(lm.1)(y~x1)
```

---

da.lmerMod.fit *Provides fit indices for hierarchical linear models, based on Nakagawa et al.(2017) and Luo and Azen (2013).*

Description

Provides fit indices for hierarchical linear models, based on Nakagawa et al.(2017) and Luo and Azen (2013).

Usage

```
da.lmerMod.fit(original.model, null.model, newdata = NULL, ...)
```

Arguments

`original.model`	Original fitted model
`null.model`	needed for HLM models
`newdata`	Data used in update statement
`...`	ignored

Value

A function described by using-fit-indices description for interface. By default, four indices are provided:

`rb.r2.1`	Amount of Level-1 variance explained by the addition of the predictor.
`rb.r2.2`	Amount of Level-2 variance explained by the addition of the predictor.
`sb.r2.1`	Proportional reduction in error of predicting scores at Level 1

sb.r2.2	Proportional reduction in error of predicting cluster means at Level 2

If performance library is available, the two following indices are also available:

n.marg	Marginal R2 coefficient based on Nakagawa et al. (2017). Considers only the variance of the fixed effects.
n.cond	Conditional R2 coefficient based on Nakagawa et al. (2017). Takes both the fixed and random effects into account.

References

- Luo, W., & Azen, R. (2013). Determining Predictor Importance in Hierarchical Linear Models Using Dominance Analysis. Journal of Educational and Behavioral Statistics, 38(1), 3-31. doi:10.3102/1076998612458319
- Nakagawa, S., Johnson, P. C. D., and Schielzeth, H. (2017). The coefficient of determination R2 and intra-class correlation coefficient from generalized linear mixed-effects models revisited and expanded. Journal of The Royal Society Interface, 14(134), 20170213.

See Also

Other fit indices: da.betareg.fit(), da.clm.fit(), da.dynlm.fit(), da.glm.fit(), da.lm.fit(), da.lmWithCov.fit(), da.mlmWithCov.fit()

---

da.lmWithCov.fit *Provides coefficient of determination for linear models, using covariance/correlation matrix.*

Description

Uses R^2 (coefficient of determination). See lmWithCov.

Usage

```
da.lmWithCov.fit(base.cov, ...)
```

Arguments

base.cov	variance/covariance matrix
...	ignored

Value

A function described by using-fit-indices description for interface. You could retrieve r2 index.

See Also

Other fit indices: da.betareg.fit(), da.clm.fit(), da.dynlm.fit(), da.glm.fit(), da.lm.fit(), da.lmerMod.fit(), da.mlmWithCov.fit()

`da.mlmWithCov.fit` *Provides coefficient of determination for multivariate models.*

Description

Provides coefficient of determination for multivariate models.

Usage

```
da.mlmWithCov.fit(base.cov, ...)
```

Arguments

`base.cov`	variance/covariance matrix
`...`	ignored

Value

A list with several fit indices

`r.squared.xy` Corresponds to R^2_{XY}

`p.squared.yx` Corresponds to P^2_{YX}

See `mlmWithCov`

References

Azen, R., & Budescu, D. V. (2006). Comparing Predictors in Multivariate Regression Models: An Extension of Dominance Analysis. Journal of Educational and Behavioral Statistics, 31(2), 157-180. doi:10.3102/10769986031002157

See Also

Other fit indices: `da.betareg.fit()`, `da.clm.fit()`, `da.dynlm.fit()`, `da.glm.fit()`, `da.lm.fit()`, `da.lmWithCov.fit()`, `da.lmerMod.fit()`

`dominanceAnalysis` *Dominance analysis for OLS (univariate and multivariate), GLM and LMM models*

Description

Dominance analysis for OLS (univariate and multivariate), GLM and LMM models

Usage

```
dominanceAnalysis(
  x,
  constants = c(),
  terms = NULL,
  fit.functions = "default",
  newdata = NULL,
  null.model = NULL,
  ...
)
```

Arguments

`x`	fitted model (lm, glm, betareg), lmWithCov or mlmWithCov object
`constants`	vector of predictors to remain unchanged between models
`terms`	vector of terms to be analyzed. By default, obtained from the model
`fit.functions`	Name of the method used to provide fit indices
`newdata`	optional data.frame, that update data used on original model
`null.model`	for mixed models, null model against to test the submodels
`...`	Other arguments provided to lm or lmer (not implemented yet)

Value

`predictors`	Vector of predictors.
`constants`	Vector of constant variables.
`terms`	Vector of terms to be analyzed.
`fit.functions`	Vector of fit indices names.
`fits`	List with raw fits indices. See `daRawResults`.
`contribution.by.level`	List of mean contribution of each predictor by level for each fit index. Each element is a data.frame, with levels as rows and predictors as columns, for each fit index.
`contribution.average`	List with mean contribution of each predictor for all levels. These values are obtained for every fit index considered in the analysis. Each element is a vector of mean contributions for a given fit index.

`complete`	Matrix for complete dominance.
`conditional`	Matrix for conditional dominance.
`general`	Matrix for general dominance.

Definition of Dominance Analysis

Budescu (1993) developed a clear and intuitive definition of importance in regression models, that states that a predictor's importance reflects its contribution in the prediction of the criterion and that one predictor is 'more important than another' if it contributes more to the prediction of the criterion than does its competitor at a given level of analysis.

Types of dominance

The original paper (Bodescu, 1993) defines that variable X_1 dominates X_2 when X_1 is chosen over X_2 in all possible subset of models where only one of these two predictors is to be entered. Later, Azen & Bodescu (2003), name the previously definition as 'complete dominance' and two other types of dominance: conditional and general dominance. Conditional dominance is calculated as the average of the additional contributions to all subset of models of a given model size. General dominance is calculated as the mean of average contribution on each level.

Fit indices availables

To obtain the fit-indices for each model, a function called `da.<model>.fit` is executed. For example, for a lm model, function `da.lm.fit` provides R^2 values. Currently, seven models are implemented:

lm Provides R^2 or coefficient of determination. See `da.lm.fit`

glm Provides four fit indices recommended by Azen & Traxel (2009): Cox and Snell(1989), McFadden (1974), Nagelkerke (1991), and Estrella (1998). See `da.glm.fit`

lmerMod Provides four fit indices recommended by Lou & Azen (2012). See `da.lmerMod.fit`

lmWithCov Provides R^2 for a correlation/covariance matrix. See `lmWithCov` to create the model and `da.lmWithCov.fit` for the fit index function.

mlmWithCov Provides both R^2_{XY} and P^2_{XY} for multivariate regression models using a correlation/covariance matrix. See `mlmWithCov` to create the model and `da.mlmWithCov.fit` for the fit index function

dynlm Provides R^2 for dynamic linear models. There is no literature reference about using dominance analysis on dynamic linear models, so you're warned!. See `da.dynlm.fit`.

betareg Provides pseudo-R^2, Cox and Snell(1989), McFadden (1974), and Estrella (1998). You could set the link function using link.betareg if automatic detection of link function doesn't work.

See `da.betareg.fit`

References

- Azen, R., & Budescu, D. V. (2003). The dominance analysis approach for comparing predictors in multiple regression. Psychological Methods, 8(2), 129-148. doi:10.1037/1082-989X.8.2.129

- Azen, R., & Budescu, D. V. (2006). Comparing Predictors in Multivariate Regression Models: An Extension of Dominance Analysis. Journal of Educational and Behavioral Statistics, 31(2), 157-180. doi:10.3102/10769986031002157
- Azen, R., & Traxel, N. (2009). Using Dominance Analysis to Determine Predictor Importance in Logistic Regression. Journal of Educational and Behavioral Statistics, 34(3), 319-347. doi:10.3102/1076998609332754
- Budescu, D. V. (1993). Dominance analysis: A new approach to the problem of relative importance of predictors in multiple regression. Psychological Bulletin, 114(3), 542-551. doi:10.1037/0033-2909.114.3.542
- Luo, W., & Azen, R. (2012). Determining Predictor Importance in Hierarchical Linear Models Using Dominance Analysis. Journal of Educational and Behavioral Statistics, 38(1), 3-31. doi:10.3102/1076998612458319

Examples

```
data(longley)
lm.1<-lm(Employed~.,longley)
da<-dominanceAnalysis(lm.1)
print(da)
summary(da)
plot(da,which.graph='complete')
plot(da,which.graph='conditional')
plot(da,which.graph='general')

# Maintaining year as a constant on all submodels
da.no.year<-dominanceAnalysis(lm.1,constants='Year')
print(da.no.year)
summary(da.no.year)
plot(da.no.year,which.graph='complete')

# Parameter terms could be used to group variables
da.terms=c(GNP.rel='GNP.deflator+GNP',
           pop.rel='Unemployed+Armed.Forces+Population+Unemployed',
           year='Year')
da.grouped<-dominanceAnalysis(lm.1,terms=da.terms)
print(da.grouped)
summary(da.grouped)
plot(da.grouped, which.graph='complete')
```

---

dominanceBriefing *Retrieve a briefing for complete, conditional and general dominance*

Description

Retrieve a briefing for complete, conditional and general dominance

Usage

```
dominanceBriefing(da.object, fit.functions = NULL, abbrev = FALSE)
```

Arguments

da.object	a dominanceAnalysis object
fit.functions	name of the fit indices to retrieve. If NULL, all fit indices will be retrieved
abbrev	if TRUE

Value

a list. Each element is a data.frame, that comprises the dominance analysis for a specific fit index. Each data.frame have the predictors as row and each column reports the predictors that are dominated for each predictor

See Also

Other retrieval methods: averageContribution(), contributionByLevel(), dominanceMatrix(), getFits()

Examples

```
# For matrix or data.frame
data(longley)
da.longley<-dominanceAnalysis(lm(Employed~.,longley))
dominanceBriefing(da.longley, abbrev=FALSE)
dominanceBriefing(da.longley, abbrev=TRUE)
```

---

dominanceMatrix *Retrieve or calculates a dominance matrix for a given object*

Description

This methods calculates or retrieve dominance matrix

This methods allows a common interface to retrieve all dominance matrices from dominanceAnalysis objects

Usage

```
dominanceMatrix(x, ...)

## S3 method for class 'data.frame'
dominanceMatrix(x, undefined.value = 0.5, ordered = FALSE, ...)

## S3 method for class 'matrix'
dominanceMatrix(x, undefined.value = 0.5, ordered = FALSE, ...)

## S3 method for class 'dominanceAnalysis'
dominanceMatrix(
  x,
```

```
  type,
  fit.functions = NULL,
  drop = TRUE,
  ordered = FALSE,
  ...
)
```

Arguments

x	matrix (calculate) or dominanceAnalysis (retrieve)
...	extra arguments. Not used
undefined.value	value when no dominance can be established
ordered	Logical. If TRUE, sort the output according to dominance.
type	type of dominance matrix to retrieve. Could be complete, conditional or general
fit.functions	name of the fit indices to retrieve. If NULL, all fit indices will be retrieved
drop	if TRUE and just one fit index is available, returns a matrix. Else, returns a list

Details

To calculate a dominance matrix from a matrix or dataframe, use

`dominanceMatrix(x,undefined.value)`.

To retrieve the dominance matrices from a dominanceAnalysis object, use

`dominanceMatrix(x,type,fit.function,drop)`

Value

for matrix and data-frame, returns a matrix representing dominance. 1 represents domination of the row variable over the column variable, 0 dominance of the column over the row variable. Undefined dominance is represented by `undefined.value` parameter. For dominanceAnalysis object, returns a matrix, if `drop` parameter if TRUE and just one index is available. Else, a list is returned, with keys as name of fit-indices and values as matrices, as described previously.

See Also

Other retrieval methods: `averageContribution()`, `contributionByLevel()`, `dominanceBriefing()`, `getFits()`

Examples

```
# For matrix or data.frame
mm<-data.frame(a=c(5,3,2),b=c(4,2,1),c=c(5,4,3))
dominanceMatrix(mm)
# For dominanceAnalysis
data(longley)
da.longley<-dominanceAnalysis(lm(Employed~.,longley))
dominanceMatrix(da.longley,type="complete")
```

getFits *Retrieve fit matrix or matrices*

Description

Retrieve fit matrix or matrices for a given dominanceAnalysis object

Usage

```
getFits(da.object, fit.functions = NULL)
```

Arguments

da.object	dominanceAnalysis object
fit.functions	name of the fit indices to retrieve. If NULL, all fit indices will be retrieved

Value

a list. Key corresponds to fit-index and the value is a matrix, with fits values

See Also

Other retrieval methods: averageContribution(), contributionByLevel(), dominanceBriefing(), dominanceMatrix()

Examples

```
data(longley)
da.longley<-dominanceAnalysis(lm(Employed~.,longley))
getFits(da.longley)
```

lmmR2 *Calculates several measures of fit for Linear Mixed Models based on Lou and Azen (2013) text. Models could be lmer or lme models.*

Description

Calculates several measures of fit for Linear Mixed Models based on Lou and Azen (2013) text. Models could be lmer or lme models.

Usage

```
lmmR2(m.null, m.full)
```

Arguments

m.null	Null model (only with random intercept effects)
m.full	Full model

Value

lmmR2 class

---

lmWithCov *Uses covariance/correlation matrix for calculate OLS*

Description

Calculate regression coefficients and R^2 for an OLS regression. Could be used with dominanceAnalysis to perform a dominance analysis without the original data.

Usage

```
lmWithCov(f, x)
```

Arguments

f	formula for lm model
x	correlation/covariance matrix

Value

coef	regression coefficients
r.squared	R^2 or coefficient of determination
formula	formula provided as parameter
cov	covariance/correlation matrix provided as parameter

Examples

```
cov.m<-matrix(c(1,0.2,0.3, 0.2,1,0.5,0.3,0.5,1),3,3,
dimnames=list(c("x1","x2","y"),c("x1","x2","y")))
lm.cov<-lmWithCov(y~x1+x2,cov.m)
da<-dominanceAnalysis(lm.cov)
```

`mlmWithCov` *Uses covariance/correlation matrix to calculate multivariate index of fit*

Description

Calculate R^2_{XY} and P^2_{YX} for multivariate regression Could be used with `dominanceAnalysis` to perform a multivariate dominance analysis without original data.

Usage

```
mlmWithCov(f, x)
```

Arguments

`f`	formula. Should use `cbind(y1,y2,...,yk)~x1+x2+..+xp`
`x`	correlation/covariance matrix

Value

`r.squared.xy`	R^2_{XY} of the regression
`p.squared.yx`	P^2_{YX} of the regression
`formula`	formula provided as parameter
`cov`	covariance/correlation matrix provided as parameter

Examples

```
library(car)
cor.m<-matrix(c(
 1.0000000, 0.7951377, 0.2617168, 0.6720053, 0.3390278,
 0.7951377, 1.0000000, 0.3341037, 0.5876337, 0.3404206,
 0.2617168, 0.3341037, 1.0000000, 0.3703162, 0.2114153,
 0.6720053, 0.5876337, 0.3703162, 1.0000000, 0.3548077,
 0.3390278, 0.3404206, 0.2114153, 0.3548077, 1.0000000),
 5,5,
 byrow = TRUE,
 dimnames = list(
   c("na","ss","SAT","PPVT","Raven"),
   c("na","ss","SAT","PPVT","Raven")))
lwith<-mlmWithCov(cbind(na,ss)~SAT+PPVT+Raven,cor.m)
da<-dominanceAnalysis(lwith)
print(da)
summary(da)
```

plot.dominanceAnalysis *Plot for a* dominanceAnalysis *object*

Description

Plot for a dominanceAnalysis object

Usage

```
## S3 method for class 'dominanceAnalysis'
plot(
  x,
  which.graph = c("general", "complete", "complete_no_facet", "conditional"),
  fit.function = NULL,
  complete_flipped_axis = TRUE,
  ...
)
```

Arguments

x	a dominanceAnalysis object
which.graph	which graph to plot
fit.function	name of the fit indices to retrieve. If NULL, first index will be used
complete_flipped_axis	For complete and complete_no_facet plot, set the R2 on X axis to allow easier visualization
...	unused

Value

a ggplot object

Examples

```
data(longley)
lm.1<-lm(Employed~.,longley)
da<-dominanceAnalysis(lm.1)
# By default, plot() shows the general dominance plot
plot(da)
# Parameter which.graph defines which type of dominance to plot
plot(da,which.graph='conditional')
plot(da,which.graph='complete')
# Parameter complete_flipped_axis allows to flip axis on complete plot, to better visualization
plot(da,which.graph='complete', complete_flipped_axis=TRUE)
plot(da,which.graph='complete', complete_flipped_axis=FALSE)
```

replaceTermsInString *Replace terms by name using the terms definition*

Description

Replace terms by name using the terms definition

Usage

```
replaceTermsInString(string, replacement)
```

Arguments

string	string to be updated
replacement	string with replacement for strings. values are replaced by names

tropicbird *Distribution of a tropical native bird species inhabiting a small oceanic island.*

Description

The dataset contains information about points distributed across a small oceanic island (Soares, 2017). In each of these points, a 10-minute count was carried out to record the species presence (assuming 1 if the species was present, or 0 if it was absent). The species' presence/absence is the binary response variable (i.e., dependent variable). Additionally, all sampled points were characterized by multiple environmental variables.

Usage

```
tropicbird
```

Format

A data frame with 2398 rows and 8 variables:

ID Point identification

rem remoteness is an index that represents the difficulty of movement through the landscape, with the highest values corresponding to the most remote areas

land land use is an index that represents the land-use intensification, with the highest values corresponding to the more humanized areas (e.g., cities, agricultural areas, horticultures, oil-palm monocultures)

alt altitude is a continuous variable, with the highest values corresponding to the higher altitude areas

slo slope is a continuous variable, with the highest values corresponding to the steepest areas

rain rainfall is a continuous variable, with the highest values corresponding to the rainy wet areas

coast distance to the coast is the minimum linear distance between each point and the coast line, with the highest values corresponding to the points further away from the coastline

pres Species presence

Source

Soares, F.C., 2017. Modelling the distribution of Sao Tome bird species: Ecological determinants and conservation prioritization. Faculdade de Ciencias da Universidade de Lisboa.

`using-fit-indices` *Provides fit indices for different regression models.*

Description

`dominanceAnalysis` tries to infer, based on the class of the model provided, the appropriate fit indices, using the scheme da.CLASS.fit for name. This method has two interfaces, one for retrieving the names of the fit indices, and another to retrieve the indices based on the data.

Arguments

`original.model`	Original fitted model
`newdata`	Data used in update statement
`null.model`	Null model, only needed for HLM models.
`base.cov`	Required if only a covariance/correlation matrix is provided.

Details

Interfaces are:

- `da.CLASS.fit("names")` returns a vector with names for fit indices
- `da.CLASS.fit(original.model, data, null.model, base.cov=NULL)` returns a function with one parameter, the formula to calculate the submodel.

Index