

Package ‘plsRbeta’

February 20, 2015

Version 0.2.0

Date 2014-12-12

Depends R (>= 2.4.0)

Imports mvtnorm, boot, Formula, plsdf, MASS, betareg, plsRglm

Enhances

Suggests pls

Title Partial Least Squares Regression for Beta Regression Models

Author Frederic Bertrand <frederic.bertrand@math.unistra.fr>, Myriam Maumy-Bertrand <myriam.maumy-bertrand@math.unistra.fr>, Nicolas Meyer <Nicolas.Meyer@nmeyer@unistra.fr>.

Maintainer Frederic Bertrand <frederic.bertrand@math.unistra.fr>

Description Provides Partial least squares Regression for (weighted) beta regression models and k-fold cross-validation of such models using various criteria. It allows for missing data in the explanatory variables. Bootstrap confidence intervals constructions are also available.

License GPL-3

Encoding latin1

URL <http://www-irma.u-strasbg.fr/~fbertran/>

Classification/MSC 62J12, 62J99

NeedsCompilation no

Repository CRAN

Date/Publication 2014-12-17 16:18:28

R topics documented:

bootplsbeta	2
coefs.plsRbeta	4
kfolds2Chisq	6
kfolds2Chisqind	7
kfolds2CVinfos_beta	8
permcoefs.plsRbeta	9

plsRbeta	11
PLS_beta	15
PLS_beta_formula	18
PLS_beta_kfoldcv	22
PLS_beta_kfoldcv_formula	25
PLS_beta_wvc	30
print.plsRbetamodel	32
print.summary.plsRbetamodel	33
simul_data_UniYX_beta	34
summary.plsRbetamodel	36
tilt.bootplsbeta	37

Index 39

bootplsbeta	<i>Non-parametric Bootstrap for PLS generalized linear models</i>
-------------	---

Description

Provides a wrapper for the bootstrap function `boot` from the `boot` R package.
 Implements non-parametric bootstrap for PLS generalized linear models by case resampling.

Usage

```
bootplsbeta(object, typeboot="plsmodel", R=250, statistic=coefs.plsRbeta,
sim="ordinary", stype="i", ...)
```

Arguments

<code>object</code>	~~Explain object here~~
<code>typeboot</code>	~~Explain typeboot here~~
<code>R</code>	~~Explain R here~~
<code>statistic</code>	~~Explain statistic here~~
<code>sim</code>	~~Explain sim here~~
<code>stype</code>	~~Explain stype here~~
<code>...</code>	~~Explain ... here~~

Details

~~ More details than the description above ~~

Value

~Describe the value returned If it is a LIST, use

<code>comp1</code>	Description of 'comp1'
<code>comp2</code>	Description of 'comp2'

...

Note

~~some notes~~

Author(s)

Frédéric Bertrand
 <frederic.bertrand@math.unistra.fr>
<http://www-irma.u-strasbg.fr/~fbertran/>

References

Frédéric Bertrand, Nicolas Meyer, Michèle Beau-Faller, Karim El Bayed, Izzie-Jacques Namer, Myriam Maumy-Bertrand (2013). Régression Bata PLS. *Journal de la Société Française de Statistique*, **154**(3):143-159. http://smf4.emath.fr/Publications/JSFds/154_3/html/

See Also

[boot](#)

Examples

```
data("GasolineYield", package="betareg")

GazYield.boot <- bootplsbeta(plsRbeta(yield~., data=GasolineYield, nt=3,
modele="pls-beta"), sim="ordinary", stype="i", R=250)
boot::boot.ci(GazYield.boot, conf = c(0.90, 0.95), type = c("norm", "basic", "perc", "bca"), index=1)
boot::boot.ci(GazYield.boot, conf = c(0.90, 0.95), type = c("norm", "basic", "perc", "bca"), index=2)
boot::boot.ci(GazYield.boot, conf = c(0.90, 0.95), type = c("norm", "basic", "perc", "bca"), index=3)
boot::boot.ci(GazYield.boot, conf = c(0.90, 0.95), type = c("norm", "basic", "perc", "bca"), index=4)
boot::boot.ci(GazYield.boot, conf = c(0.90, 0.95), type = c("norm", "basic", "perc", "bca"), index=5)
boot::boot.ci(GazYield.boot, conf = c(0.90, 0.95), type = c("norm", "basic", "perc", "bca"), index=6)

plsRglm::boxplots.bootpls(GazYield.boot)
plsRglm::confints.bootpls(GazYield.boot)
plsRglm::plots.confints.bootpls(plsRglm::confints.bootpls(GazYield.boot))

plot(GazYield.boot, index=2)
boot::jack.after.boot(GazYield.boot, index=2, useJ=TRUE, nt=3)
plot(GazYield.boot, index=2, jack=TRUE)

# PLS bootstrap balanced

GazYield.boot <- bootplsbeta(plsRbeta(yield~., data=GasolineYield, nt=3,
modele="pls-beta"), sim="balanced", stype="i", R=250)
boot::boot.ci(GazYield.boot, conf = c(0.90, 0.95), type = c("norm", "basic", "perc", "bca"), index=1)
boot::boot.ci(GazYield.boot, conf = c(0.90, 0.95), type = c("norm", "basic", "perc", "bca"), index=2)
boot::boot.ci(GazYield.boot, conf = c(0.90, 0.95), type = c("norm", "basic", "perc", "bca"), index=3)
boot::boot.ci(GazYield.boot, conf = c(0.90, 0.95), type = c("norm", "basic", "perc", "bca"), index=4)
```

```
boot::boot.ci(GazYield.boot, conf = c(0.90,0.95), type = c("norm","basic","perc","bca"), index=5)
boot::boot.ci(GazYield.boot, conf = c(0.90,0.95), type = c("norm","basic","perc","bca"), index=6)
```

```
plsRglm::boxplots.bootpls(GazYield.boot)
plsRglm::confints.bootpls(GazYield.boot)
plsRglm::plots.confints.bootpls(plsRglm::confints.bootpls(GazYield.boot))
```

```
plot(GazYield.boot)
boot::jack.after.boot(GazYield.boot, index=1, useJ=TRUE, nt=3)
plot(GazYield.boot,jack=TRUE)
```

```
# PLS permutation bootstrap
```

```
GazYield.boot <- bootplsbeta(plsRbeta(yield~,data=GasolineYield,nt=3,
modele="pls-beta"), sim="permutation", stype="i", R=250)
boot::boot.ci(GazYield.boot, conf = c(0.90,0.95), type = c("norm","basic","perc"), index=1)
boot::boot.ci(GazYield.boot, conf = c(0.90,0.95), type = c("norm","basic","perc"), index=2)
boot::boot.ci(GazYield.boot, conf = c(0.90,0.95), type = c("norm","basic","perc"), index=3)
boot::boot.ci(GazYield.boot, conf = c(0.90,0.95), type = c("norm","basic","perc"), index=4)
boot::boot.ci(GazYield.boot, conf = c(0.90,0.95), type = c("norm","basic","perc"), index=5)
boot::boot.ci(GazYield.boot, conf = c(0.90,0.95), type = c("norm","basic","perc"), index=6)
plsRglm::boxplots.bootpls(GazYield.boot)
plot(GazYield.boot)
```

coefs.plsRbeta *Coefficients for bootstrap computations*

Description

~~ A (1-5 lines) description of what the function does. ~~

Usage

```
coefs.plsRbeta(dataset, ind, nt, modele, family=NULL, method="logistic",
link=NULL, link.phi=NULL, type="ML")
```

Arguments

dataset	~~Explain dataset here~~
ind	~~Explain ind here~~
nt	~~Explain nt here~~
modele	name of the PLS glm or PLS beta model to be fitted ("pls", "pls-glm-Gamma", "pls-glm-gaussian", "pls-glm-inverse.gaussian", "pls-glm-logistic", "pls-glm-poisson", "pls-glm-polr", "pls-beta"). Use "modele=pls-glm-family" to enable the family option.

family ~~~Explain family here~~~
 method ~~~Explain method here~~~
 link ~~~Explain link here~~~
 link.phi ~~~Explain link.phi here~~~
 type ~~~Explain type here~~~

Details

~~ More details than the description above ~~

Value

~Describe the value returned If it is a LIST, use

comp1 Description of 'comp1'
 comp2 Description of 'comp2'
 ...

Note

~~some notes~~

Author(s)

Frédéric Bertrand
 <frederic.bertrand@math.unistra.fr>
<http://www-irma.u-strasbg.fr/~fbertran/>

References

Frédéric Bertrand, Nicolas Meyer, Michèle Beau-Faller, Karim El Bayed, Izzie-Jacques Namer, Myriam Maumy-Bertrand (2013). Régression Bata PLS. *Journal de la Société Française de Statistique*, **154**(3):143-159. http://smf4.emath.fr/Publications/JSFds/154_3/html/

See Also

~~objects to See Also as [help](#), ~~~

Examples

data("GasolineYield", package="betareg")

kfold2Chisq	<i>Computes Predicted Chisquare for kfold cross validated partial least squares regression models.</i>
-------------	--

Description

This function computes Predicted Chisquare for kfold cross validated partial least squares regression models.

Usage

```
kfold2Chisq(pls_kfolds)
```

Arguments

pls_kfolds a kfold cross validated partial least squares regression glm model

Value

list	Total Predicted Chisquare vs number of components for the first group partition
...	...
list	Total Predicted Chisquare vs number of components for the last group partition

Note

Use [PLS_beta_kfoldcv](#) to create kfold cross validated partial least squares regression glm and beta models.

Author(s)

Frédéric Bertrand
 <frederic.bertrand@math.unistra.fr>
<http://www-irma.u-strasbg.fr/~fbertran/>

References

Frédéric Bertrand, Nicolas Meyer, Michèle Beau-Faller, Karim El Bayed, Izzie-Jacques Namer, Myriam Maumy-Bertrand (2013). Régression Bata PLS. *Journal de la Société Française de Statistique*, **154**(3):143-159. http://smf4.emath.fr/Publications/JSFds/154_3/html/

See Also

[kfold2coeff](#), [kfold2Press](#), [kfold2Pressind](#), [kfold2Chisqind](#), [kfold2Mclassifiedind](#) and [kfold2Mclassified](#) to extract and transforms results from kfold cross validation.

Examples

```
## Not run:
data("GasolineYield",package="betareg")
yGasolineYield <- GasolineYield$yield
XGasolineYield <- GasolineYield[,2:5]
bbb <- PLS_beta_kfoldcv(yGasolineYield,XGasolineYield,nt=3,modele="pls-beta")
kfold2Chisq(bbb)

## End(Not run)
```

kfold2Chisqind	<i>Computes individual Predicted Chisquare for kfold cross validated partial least squares regression models.</i>
----------------	---

Description

This function computes individual Predicted Chisquare for kfold cross validated partial least squares regression models.

Usage

```
kfold2Chisqind(pls_kfolds)
```

Arguments

pls_kfolds a kfold cross validated partial least squares regression glm model

Value

list	Individual PChisq vs number of components for the first group partition
...	...
list	Individual PChisq vs number of components for the last group partition

Note

Use [PLS_beta_kfoldcv](#) to create kfold cross validated partial least squares regression glm models.

Author(s)

Frédéric Bertrand
 <frederic.bertrand@math.unistra.fr>
<http://www-irma.u-strasbg.fr/~fbertran/>

References

Frédéric Bertrand, Nicolas Meyer, Michèle Beau-Faller, Karim El Bayed, Izzie-Jacques Namer, Myriam Maumy-Bertrand (2013). Régression Bata PLS. *Journal de la Société Française de Statistique*, **154**(3):143-159. http://smf4.emath.fr/Publications/JSFds/154_3/html/

See Also

[kfolds2coeff](#), [kfolds2Press](#), [kfolds2Pressind](#), [kfolds2Chisq](#), [kfolds2McClassedind](#) and [kfolds2McClassed](#) to extract and transforms results from kfold cross validation.

Examples

```
## Not run:
data("GasolineYield", package="betareg")
yGasolineYield <- GasolineYield$yield
XGasolineYield <- GasolineYield[,2:5]
bbb <- PLS_beta_kfoldcv(yGasolineYield, XGasolineYield, nt=3, modele="pls-beta")
kfolds2Chisqind(bbb)

## End(Not run)
```

kfolds2CVinfos_beta	<i>Extracts and computes information criteria and fits statistics for kfold cross validated partial least squares beta regression models</i>
---------------------	--

Description

This function extracts and computes information criteria and fits statistics for kfold cross validated partial least squares beta regression models for both formula or classic specifications of the model.

Usage

```
kfolds2CVinfos_beta(pls_kfolds, McClassed = FALSE)
```

Arguments

pls_kfolds	an object computed using PLS_beta_kfoldcv
McClassed	should number of miss classed be computed

Details

The McClassed option should only set to TRUE if the response is binary.

Value

list	table of fit statistics for first group partition
...	...
list	table of fit statistics for last group partition

Author(s)

Frédéric Bertrand
 <frederic.bertrand@math.unistra.fr>
<http://www-irma.u-strasbg.fr/~fbertran/>

References

Frédéric Bertrand, Nicolas Meyer, Michèle Beau-Faller, Karim El Bayed, Izzie-Jacques Namer, Myriam Maumy-Bertrand (2013). Régression à la PLS. *Journal de la Société Française de Statistique*, **154**(3):143-159. http://smf4.emath.fr/Publications/JSFds/154_3/html/

See Also

[kfold2coeff](#), [kfold2Pressind](#), [kfold2Press](#), [kfold2Mclassifiedind](#) and [kfold2Mclassified](#) to extract and transforms results from kfold cross validation.

Examples

```
## Not run:
data("GasolineYield",package="betareg")
bbb <- PLS_beta_kfoldcv_formula(yield~.,data=GasolineYield,nt=3,modele="pls-beta")
kfold2CVinfos_beta(bbb)

## End(Not run)
```

permcoefs.plsRbeta *Coefficients computation for permutation bootstrap*

Description

~~ A (1-5 lines) description of what the function does. ~~

Usage

```
permcoefs.plsRbeta(dataset, ind, nt, modele, family=NULL, method="logistic",
link="logit",link.phi=NULL,type="ML")
```

Arguments

dataset	~~Explain dataset here~~
ind	~~Explain ind here~~
nt	~~Explain nt here~~
modele	~~Explain modele here~~
family	~~Explain family here~~
method	~~Explain method here~~
link	~~Explain link here~~
link.phi	~~Explain link.phi here~~
type	~~Explain type here~~

Details

~~ More details than the description above ~~

Value

~Describe the value returned If it is a LIST, use

comp1 Description of 'comp1'

comp2 Description of 'comp2'

...

Note

~~some notes~~

Author(s)

Frédéric Bertrand
<frederic.bertrand@math.unistra.fr>
<http://www-irma.u-strasbg.fr/~fbertran/>

References

Frédéric Bertrand, Nicolas Meyer, Michèle Beau-Faller, Karim El Bayed, Izzie-Jacques Namer, Myriam Maumy-Bertrand (2013). Régression à PLS. *Journal de la Société Française de Statistique*, **154**(3):143-159. http://smf4.emath.fr/Publications/JSFds/154_3/html/

See Also

~~objects to See Also as [help](#), ~~~

Examples

```
## Not run: print("to do")
```

Description

This function implements Partial least squares Regression generalized linear models complete or incomplete datasets.

Usage

```
plsRbeta(x, ...)
## Default S3 method:
plsRbetamodel(dataY,dataX,nt=2,limQ2set=.0975,
dataPredictY=dataX,modele="pls",family=NULL,typeVC="none",EstimXNA=FALSE,
scaleX=TRUE,scaleY=NULL,pvals.expli=FALSE,alpha.pvals.expli=.05,
MClassed=FALSE,tol_Xi=10^(-12),weights,method,sparse=FALSE,sparseStop=TRUE,
naive=FALSE,link=NULL,link.phi=NULL,type="ML")
## S3 method for class 'formula'
plsRbetamodel(formula,data=NULL,nt=2,limQ2set=.0975,
dataPredictY,modele="pls",family=NULL,typeVC="none",EstimXNA=FALSE,
scaleX=TRUE,scaleY=NULL,pvals.expli=FALSE,alpha.pvals.expli=.05,
MClassed=FALSE,tol_Xi=10^(-12),weights,subset,start=NULL,etastart,
mustart,offset,method="glm.fit",control= list(),contrasts=NULL,
sparse=FALSE,sparseStop=TRUE,naive=FALSE,link=NULL,link.phi=NULL,type="ML")
```

Arguments

x	a formula or a response (training) dataset
dataY	response (training) dataset
dataX	predictor(s) (training) dataset
formula	an object of class " formula " (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
data	an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment (formula), typically the environment from which plsRbeta is called.
nt	number of components to be extracted
limQ2set	limit value for the Q2
dataPredictY	predictor(s) (testing) dataset
modele	name of the PLS glm or PLS beta model to be fitted ("pls", "pls-glm-Gamma", "pls-glm-gaussian", "pls-glm-inverse.gaussian", "pls-glm-logistic", "pls-glm-poisson", "pls-glm-polr", "pls-beta"). Use "modele=pls-glm-family" to enable the family option.

family	a description of the error distribution and link function to be used in the model. This can be a character string naming a family function, a family function or the result of a call to a family function. (See family for details of family functions.) To use the family option, please set <code>modele="pls-glm-family"</code> . User defined families can also be defined. See details.
typeVC	type of leave one out cross validation. For back compatibility purpose. none no cross validation standard no cross validation missingdata no cross validation adaptative no cross validation
EstimXNA	only for <code>modele="pls"</code> . Set whether the missing X values have to be estimated.
scaleX	scale the predictor(s) : must be set to TRUE for <code>modele="pls"</code> and should be for glms pls.
scaleY	scale the response : Yes/No. Ignored since non always possible for glm responses.
pvals.expli	should individual p-values be reported to tune model selection ?
alpha.pvals.expli	level of significance for predictors when <code>pvals.expli=TRUE</code>
MClassed	number of missclassified cases, should only be used for binary responses
tol_Xi	minimal value for $\text{Norm2}(X_i)$ and $\det(pp' \times pp)$ if there is any missing value in the dataX. It defaults to 10^{-12}
weights	an optional vector of 'prior weights' to be used in the fitting process. Should be NULL or a numeric vector.
subset	an optional vector specifying a subset of observations to be used in the fitting process.
start	starting values for the parameters in the linear predictor.
etastart	starting values for the linear predictor.
mustart	starting values for the vector of means.
offset	this can be used to specify an <i>a priori</i> known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector of length equal to the number of cases. One or more offset terms can be included in the formula instead or as well, and if more than one is specified their sum is used. See model.offset .
method	the method to be used in fitting the model. The default method " <code>glm.fit</code> " uses iteratively reweighted least squares (IWLS). User-supplied fitting functions can be supplied either as a function or a character string naming a function, with a function which takes the same arguments as <code>glm.fit</code> .
control	a list of parameters for controlling the fitting process. For <code>glm.fit</code> this is passed to glm.control .
contrasts	an optional list. See the <code>contrasts.arg</code> of <code>model.matrix.default</code> .
sparse	should the coefficients of non-significant predictors (<code><alpha.pvals.expli</code>) be set to 0

sparseStop	should component extraction stop when no significant predictors (<code><alpha.pvals.expli</code>) are found
naive	Use the naive estimates for the Degrees of Freedom in plsR? Default is FALSE.
link	character specification of the link function in the mean model (μ). Currently, "logit", "probit", "cloglog", "cauchit", "log", "loglog" are supported. Alternatively, an object of class "link-glm" can be supplied.
link.phi	character specification of the link function in the precision model (ϕ). Currently, "identity", "log", "sqrt" are supported. The default is "log" unless formula is of type $y \sim x$ where the default is "identity" (for backward compatibility). Alternatively, an object of class "link-glm" can be supplied.
type	character specification of the type of estimator. Currently, maximum likelihood ("ML"), ML with bias correction ("BC"), and ML with bias reduction ("BR") are supported.
...	arguments to pass to <code>plsRmodel.default</code> or to <code>plsRmodel.formula</code>

Details

There are seven different predefined models with predefined link functions available :

"pls" ordinary pls models

"pls-glm-Gamma" glm gaussian with inverse link pls models

"pls-glm-gaussian" glm gaussian with identity link pls models

"pls-glm-inverse-gamma" glm binomial with square inverse link pls models

"pls-glm-logistic" glm binomial with logit link pls models

"pls-glm-poisson" glm poisson with log link pls models

"pls-glm-polr" glm polr with logit link pls models

Using the "family=" option and setting "modele=pls-glm-family" allows changing the family and link function the same way as for the `glm` function. As a consequence user-specified families can also be used.

The gaussian family accepts the links (as names) identity, log and inverse.

The binomial family accepts the links logit, probit, cauchit, (corresponding to logistic, normal and Cauchy CDFs respectively) log and cloglog (complementary log-log).

The Gamma family accepts the links inverse, identity and log.

The poisson family accepts the links log, identity, and sqrt.

The inverse.gaussian family accepts the links $1/\mu^2$, inverse, identity and log.

The quasi family accepts the links logit, probit, cloglog, identity, inverse, log, $1/\mu^2$ and sqrt.

The function power can be used to create a power link function.

A typical predictor has the form `response ~ terms` where response is the (numeric) response vector and terms is a series of terms which specifies a linear predictor for response. A terms specification of the form `first + second` indicates all the terms in first together with all the terms in second with any duplicates removed.

A specification of the form first:second indicates the the set of terms obtained by taking the interactions of all terms in first with all terms in second. The specification first*second indicates the cross of first and second. This is the same as first + second + first:second.

The terms in the formula will be re-ordered so that main effects come first, followed by the interactions, all second-order, all third-order and so on: to avoid this pass a terms object as the formula.

Non-NULL weights can be used to indicate that different observations have different dispersions (with the values in weights being inversely proportional to the dispersions); or equivalently, when the elements of weights are positive integers w_i , that each response y_i is the mean of w_i unit-weight observations.

The default estimator for Degrees of Freedom is the Kramer and Sugiyama's one which only works for classical plsR models. For these models, Information criteria are computed accordingly to these estimations. Naive Degrees of Freedom and Information Criteria are also provided for comparison purposes. For more details, see Kraemer, N., Sugiyama M. (2010). "The Degrees of Freedom of Partial Least Squares Regression". preprint, <http://arxiv.org/abs/1002.4112>.

Value

Depends on the model that was used to fit the model.

Note

Use plsRbeta instead.

Author(s)

Frédéric Bertrand
 <frederic.bertrand@math.unistra.fr>
<http://www-irma.u-strasbg.fr/~fbertran/>

References

Frédéric Bertrand, Nicolas Meyer, Michèle Beau-Faller, Karim El Bayed, Izzie-Jacques Namer, Myriam Maumy-Bertrand (2013). Régression à la PLS. *Journal de la Société Française de Statistique*, **154**(3):143-159. http://smf4.emath.fr/Publications/JSFds/154_3/html/

See Also

[plsR](#) and [plsRglm](#)

Examples

```
data("GasolineYield", package="betareg")
modpls <- plsRbeta(yield~., data=GasolineYield, nt=3, modele="pls-beta")
modpls$uscores
modpls$pp
modpls$Coeffs
modpls$Std.Coeffs
```

```

modpls$InfCrit
modpls$PredictY[1,]
rm("modpls")

data("GasolineYield",package="betareg")
yGasolineYield <- GasolineYield$yield
XGasolineYield <- GasolineYield[,2:5]
modpls <- plsRbeta(yGasolineYield,XGasolineYield,nt=3,modele="pls-beta")
modpls$uscores
modpls$pp
modpls$Coeffs
modpls$Std.Coeffs
modpls$InfCrit
modpls$PredictY[1,]
rm("modpls")

```

PLS_beta

Partial least squares Regression generalized linear models

Description

This function implements Partial least squares Regression generalized linear models complete or incomplete datasets.

Usage

```

PLS_beta(dataY, dataX, nt = 2, limQ2set = 0.0975, dataPredictY = dataX,
modele = "pls", family = NULL, typeVC = "none", EstimXNA = FALSE,
scaleX = TRUE, scaleY = NULL, pvals.expli = FALSE, alpha.pvals.expli = 0.05,
MClassed = FALSE, tol_Xi = 10-12, weights, method, sparse = FALSE,
sparseStop=TRUE,naive=FALSE,link=NULL,link.phi=NULL,type="ML")

```

Arguments

dataY	response (training) dataset
dataX	predictor(s) (training) dataset
nt	number of components to be extracted
limQ2set	limit value for the Q2
dataPredictY	predictor(s) (testing) dataset
modele	name of the PLS glm or PLS beta model to be fitted ("pls", "pls-glm-Gamma", "pls-glm-gaussian", "pls-glm-inverse.gaussian", "pls-glm-logistic", "pls-glm-poisson", "pls-glm-polr", "pls-beta"). Use "modele=pls-glm-family" to enable the family option.

family	a description of the error distribution and link function to be used in the model. This can be a character string naming a family function, a family function or the result of a call to a family function. (See family for details of family functions.) To use the family option, please set <code>modele="pls-glm-family"</code> . User defined families can also be defined. See details.
typeVC	type of leave one out cross validation. For back compatibility purpose. none no cross validation standard no cross validation missingdata no cross validation adaptative no cross validation
EstimXNA	only for <code>modele="pls"</code> . Set whether the missing X values have to be estimated.
scaleX	scale the predictor(s) : must be set to TRUE for <code>modele="pls"</code> and should be for glms pls.
scaleY	scale the response : Yes/No. Ignored since not always possible for glm responses.
pvals.expli	should individual p-values be reported to tune model selection ?
alpha.pvals.expli	level of significance for predictors when <code>pvals.expli=TRUE</code>
MClassed	number of missclassified cases, should only be used for binary responses
tol_Xi	minimal value for $\text{Norm2}(X_i)$ and $\det(pp' \times pp)$ if there is any missing value in the dataX. It defaults to 10^{-12}
weights	an optional vector of 'prior weights' to be used in the fitting process. Should be NULL or a numeric vector.
method	the link function for <code>pls-glm-polr</code> , logistic, probit, complementary log-log or cauchit (corresponding to a Cauchy latent variable).
sparse	should the coefficients of non-significant predictors (<code><alpha.pvals.expli</code>) be set to 0
sparseStop	should component extraction stop when no significant predictors (<code><alpha.pvals.expli</code>) are found
naive	use the naive estimates for the Degrees of Freedom in plsR? Default is FALSE.
link	character specification of the link function in the mean model (mu). Currently, "logit", "probit", "cloglog", "cauchit", "log", "loglog" are supported. Alternatively, an object of class "link-glm" can be supplied.
link.phi	character specification of the link function in the precision model (phi). Currently, "identity", "log", "sqrt" are supported. The default is "log" unless formula is of type $y \sim x$ where the default is "identity" (for backward compatibility). Alternatively, an object of class "link-glm" can be supplied.
type	character specification of the type of estimator. Currently, maximum likelihood ("ML"), ML with bias correction ("BC"), and ML with bias reduction ("BR") are supported.

Details

There are seven different predefined models with predefined link functions available :

"pls" ordinary pls models

"pls-glm-Gamma" glm gaussian with inverse link pls models

"pls-glm-gaussian" glm gaussian with identity link pls models

"pls-glm-inverse-gamma" glm binomial with square inverse link pls models

"pls-glm-logistic" glm binomial with logit link pls models

"pls-glm-poisson" glm poisson with log link pls models

"pls-glm-polr" glm polr with logit link pls models

Using the "family=" option and setting "modele=pls-glm-family" allows changing the family and link function the same way as for the `glm` function. As a consequence user-specified families can also be used.

The gaussian family accepts the links (as names) identity, log and inverse.

The binomial family accepts the links logit, probit, cauchit, (corresponding to logistic, normal and Cauchy CDFs respectively) log and cloglog (complementary log-log).

The Gamma family accepts the links inverse, identity and log.

The poisson family accepts the links log, identity, and sqrt.

The inverse.gaussian family accepts the links $1/\mu^2$, inverse, identity and log.

The quasi family accepts the links logit, probit, cloglog, identity, inverse, log, $1/\mu^2$ and sqrt.

The function power can be used to create a power link function.

The default estimator for Degrees of Freedom is the Kramer and Sugiyama's one which only works for classical plsR models. For these models, Information criteria are computed accordingly to these estimations. Naive Degrees of Freedom and Information Criteria are also provided for comparison purposes. For more details, see Kraemer, N., Sugiyama M. (2010). "The Degrees of Freedom of Partial Least Squares Regression". preprint, <http://arxiv.org/abs/1002.4112>.

Value

Depends on the model that was used to fit the model.

Note

Use `plsRbeta` instead.

Author(s)

Frédéric Bertrand

<frederic.bertrand@math.unistra.fr>

<http://www-irma.u-strasbg.fr/~fbertran/>

References

Frédéric Bertrand, Nicolas Meyer, Michèle Beau-Faller, Karim El Bayed, Izzie-Jacques Namer, Myriam Maumy-Bertrand (2013). Régression à la PLS. *Journal de la Société Française de Statistique*, **154**(3):143-159. http://smf4.emath.fr/Publications/JSFds/154_3/html/

See Also

[PLS_beta_wvc](#) and [PLS_beta_kfoldcv](#)

Examples

```
data("GasolineYield",package="betareg")
yGasolineYield <- GasolineYield$yield
XGasolineYield <- GasolineYield[,2:5]
modpls <- PLS_beta(yGasolineYield,XGasolineYield,nt=3,modele="pls-beta")
modpls$uscores
modpls$pp
modpls$Coeffs
modpls$Std.Coeffs
modpls$InfCrit
modpls$PredictY[1,]
rm("modpls")
```

PLS_beta_formula

Partial least squares Regression generalized linear models

Description

This function implements Partial least squares Regression generalized linear models complete or incomplete datasets (formula specification of the model).

Usage

```
PLS_beta_formula(formula,data=NULL,nt=2,limQ2set=.0975,
dataPredictY=dataX,modele="pls",family=NULL,typeVC="none",
EstimXNA=FALSE,scaleX=TRUE,scaleY=NULL,pvals.expli=FALSE,
alpha.pvals.expli=.05,MClassed=FALSE,tol_Xi=10^(-12),
weights,subset,start=NULL,etastart,mustart,offset,method,
control= list(),contrasts=NULL,sparse=FALSE,sparseStop=TRUE,
naive=FALSE,link=NULL,link.phi=NULL,type="ML")
```

Arguments

formula	an object of class "formula" (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
data	an optional data frame, list or environment (or object coercible by <code>as.data.frame</code> to a data frame) containing the variables in the model. If not found in data, the variables are taken from <code>environment(formula)</code> , typically the environment from which <code>plsRbeta</code> is called.
nt	number of components to be extracted
limQ2set	limit value for the Q2
dataPredictY	predictor(s) (testing) dataset
modele	name of the PLS glm or PLS beta model to be fitted ("pls", "pls-glm-Gamma", "pls-glm-gaussian", "pls-glm-inverse.gaussian", "pls-glm-logistic", "pls-glm-poisson", "pls-glm-polr", "pls-beta"). Use "modele=pls-glm-family" to enable the family option.
family	a description of the error distribution and link function to be used in the model. This can be a character string naming a family function, a family function or the result of a call to a family function. (See <code>family</code> for details of family functions.) To use the family option, please set <code>modele="pls-glm-family"</code> . User defined families can also be defined. See details.
typeVC	type of leave one out cross validation. For back compatibility purpose. none no cross validation standard no cross validation missingdata no cross validation adaptative no cross validation
EstimXNA	only for <code>modele="pls"</code> . Set whether the missing X values have to be estimated.
scaleX	scale the predictor(s) : must be set to TRUE for <code>modele="pls"</code> and should be for glms pls.
scaleY	scale the response : Yes/No. Ignored since not always possible for glm responses.
pvals.expli	should individual p-values be reported to tune model selection ?
alpha.pvals.expli	level of significance for predictors when <code>pvals.expli=TRUE</code>
MClassed	number of missclassified cases, should only be used for binary responses
tol_Xi	minimal value for $\text{Norm2}(X_i)$ and $\det(pp' \times pp)$ if there is any missing value in the dataX. It defaults to 10^{-12}
weights	an optional vector of 'prior weights' to be used in the fitting process. Should be NULL or a numeric vector.
subset	an optional vector specifying a subset of observations to be used in the fitting process.
start	starting values for the parameters in the linear predictor.
etastart	starting values for the linear predictor.

mustart	starting values for the vector of means.
offset	this can be used to specify an <i>a priori</i> known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector of length equal to the number of cases. One or more <code>offset</code> terms can be included in the formula instead or as well, and if more than one is specified their sum is used. See <code>model.offset</code> .
method	for fitting glms with glm ("pls-glm-Gamma", "pls-glm-gaussian", "pls-glm-inverse.gaussian", the method to be used in fitting the model. The default method "glm.fit" uses iteratively reweighted least squares (IWLS). User-supplied fitting functions can be supplied either as a function or a character string naming a function, with a function which takes the same arguments as <code>glm.fit</code> . If "model.frame", the model frame is returned. pls-glm-polr logistic, probit, complementary log-log or cauchit (corresponding to a Cauchy latent variable).
control	a list of parameters for controlling the fitting process. For <code>glm.fit</code> this is passed to <code>glm.control</code> .
contrasts	an optional list. See the <code>contrasts.arg</code> of <code>model.matrix.default</code> .
sparse	should the coefficients of non-significant predictors (<code><alpha.pvals.expli</code>) be set to 0
sparseStop	should component extraction stop when no significant predictors (<code><alpha.pvals.expli</code>) are found
naive	Use the naive estimates for the Degrees of Freedom in plsR? Default is FALSE.
link	character specification of the link function in the mean model (μ). Currently, "logit", "probit", "cloglog", "cauchit", "log", "loglog" are supported. Alternatively, an object of class "link-glm" can be supplied.
link.phi	character specification of the link function in the precision model (ϕ). Currently, "identity", "log", "sqrt" are supported. The default is "log" unless formula is of type $y \sim x$ where the default is "identity" (for backward compatibility). Alternatively, an object of class "link-glm" can be supplied.
type	character specification of the type of estimator. Currently, maximum likelihood ("ML"), ML with bias correction ("BC"), and ML with bias reduction ("BR") are supported.

Details

There are seven different predefined models with predefined link functions available :

"pls" ordinary pls models

"pls-glm-Gamma" glm gaussian with inverse link pls models

"pls-glm-gaussian" glm gaussian with identity link pls models

"pls-glm-inverse-gamma" glm binomial with square inverse link pls models

"pls-glm-logistic" glm binomial with logit link pls models

"pls-glm-poisson" glm poisson with log link pls models

"pls-glm-polr" glm polr with logit link pls models

Using the "family=" option and setting "modele=pls-glm-family" allows changing the family and link function the same way as for the `glm` function. As a consequence user-specified families can also be used.

The gaussian family accepts the links (as names) identity, log and inverse.

The binomial family accepts the links logit, probit, cauchit, (corresponding to logistic, normal and Cauchy CDFs respectively) log and cloglog (complementary log-log).

The Gamma family accepts the links inverse, identity and log.

The poisson family accepts the links log, identity, and sqrt.

The inverse.gaussian family accepts the links $1/\mu^2$, inverse, identity and log.

The quasi family accepts the links logit, probit, cloglog, identity, inverse, log, $1/\mu^2$ and sqrt.

The function power can be used to create a power link function.

A typical predictor has the form `response ~ terms` where `response` is the (numeric) response vector and `terms` is a series of terms which specifies a linear predictor for response. A terms specification of the form `first + second` indicates all the terms in `first` together with all the terms in `second` with any duplicates removed.

A specification of the form `first:second` indicates the the set of terms obtained by taking the interactions of all terms in `first` with all terms in `second`. The specification `first*second` indicates the cross of `first` and `second`. This is the same as `first + second + first:second`.

The terms in the formula will be re-ordered so that main effects come first, followed by the interactions, all second-order, all third-order and so on: to avoid this pass a terms object as the formula.

Non-NULL weights can be used to indicate that different observations have different dispersions (with the values in weights being inversely proportional to the dispersions); or equivalently, when the elements of weights are positive integers w_i , that each response y_i is the mean of w_i unit-weight observations.

The default estimator for Degrees of Freedom is the Kramer and Sugiyama's one which only works for classical plsR models. For these models, Information criteria are computed accordingly to these estimations. Naive Degrees of Freedom and Information Criteria are also provided for comparison purposes. For more details, see Kraemer, N., Sugiyama M. (2010). "The Degrees of Freedom of Partial Least Squares Regression". preprint, <http://arxiv.org/abs/1002.4112>.

Value

Depends on the model that was used to fit the model.

Note

Use `plsRbeta` instead.

Author(s)

Frédéric Bertrand
 <frederic.bertrand@math.unistra.fr>
<http://www-irma.u-strasbg.fr/~fbertran/>

References

Frédéric Bertrand, Nicolas Meyer, Michèle Beau-Faller, Karim El Bayed, Izzie-Jacques Namer, Myriam Maumy-Bertrand (2013). Régression à la PLS. *Journal de la Société Française de Statistique*, **154**(3):143-159. http://smf4.emath.fr/Publications/JSFds/154_3/html/

See Also

[PLS_beta_wvc](#) and [PLS_beta_kfoldcv_formula](#)

Examples

```
data("GasolineYield",package="betareg")
modpls <- PLS_beta_formula(yield~.,data=GasolineYield,nt=3,modele="pls-beta")
modpls$uscores
modpls$pp
modpls$Coeffs
modpls$Std.Coeffs
modpls$InfCrit
modpls$PredictY[1,]
rm("modpls")
```

PLS_beta_kfoldcv	<i>Partial least squares regression beta models with kfold cross validation</i>
------------------	---

Description

This function implements kfold cross validation on complete or incomplete datasets for partial least squares regression generalized linear models

Usage

```
PLS_beta_kfoldcv(dataY, dataX, nt = 2, limQ2set = 0.0975, modele = "pls",
family = NULL, K = nrow(dataX), NK = 1, grouplist = NULL, random = FALSE,
scaleX = TRUE, scaleY = NULL, keepcoeffs = FALSE, keepfolds = FALSE,
keepdataY = TRUE, keepMclassed=FALSE, tol_Xi = 10^(-12), weights,
method,link=NULL,link.phi=NULL,type="ML")
```

Arguments

dataY	response (training) dataset
dataX	predictor(s) (training) dataset
nt	number of components to be extracted
limQ2set	limit value for the Q2

modele	name of the PLS glm or PLS beta model to be fitted ("pls", "pls-glm-Gamma", "pls-glm-gaussian", "pls-glm-inverse.gaussian", "pls-glm-logistic", "pls-glm-poisson", "pls-glm-polr", "pls-beta"). Use "modele=pls-glm-family" to enable the family option.
family	a description of the error distribution and link function to be used in the model. This can be a character string naming a family function, a family function or the result of a call to a family function. (See family for details of family functions.) To use the family option, please set modele="pls-glm-family". User defined families can also be defined. See details.
K	number of groups
NK	number of times the group division is made
grouplist	to specify the members of the K groups
random	should the K groups be made randomly
scaleX	scale the predictor(s) : must be set to TRUE for modele="pls" and should be for glms pls.
scaleY	scale the response : Yes/No. Ignored since non always possible for glm responses.
keepcoeffs	shall the coefficients for each model be returned
keepfolds	shall the groups' composition be returned
keepdataY	shall the observed value of the response for each one of the predicted value be returned
keepMclassified	shall the number of miss classed be returned (unavailable)
tol_Xi	minimal value for Norm2(Xi) and $\det(pp' \times pp)$ if there is any missing value in the dataX. It defaults to 10^{-12}
weights	an optional vector of 'prior weights' to be used in the fitting process. Should be NULL or a numeric vector.
method	logistic, probit, complementary log-log or cauchit (corresponding to a Cauchy latent variable).
link	character specification of the link function in the mean model (mu). Currently, "logit", "probit", "cloglog", "cauchit", "log", "loglog" are supported. Alternatively, an object of class "link-glm" can be supplied.
link.phi	character specification of the link function in the precision model (phi). Currently, "identity", "log", "sqrt" are supported. The default is "log" unless formula is of type $y \sim x$ where the default is "identity" (for backward compatibility). Alternatively, an object of class "link-glm" can be supplied.
type	character specification of the type of estimator. Currently, maximum likelihood ("ML"), ML with bias correction ("BC"), and ML with bias reduction ("BR") are supported.

Details

Predicts 1 group with the K-1 other groups. Leave one out cross validation is thus obtained for $K == \text{nrow}(\text{dataX})$.

There are seven different predefined models with predefined link functions available :

"pls" ordinary pls models
 "pls-glm-Gamma" glm gaussian with inverse link pls models
 "pls-glm-gaussian" glm gaussian with identity link pls models
 "pls-glm-inverse-gamma" glm binomial with square inverse link pls models
 "pls-glm-logistic" glm binomial with logit link pls models
 "pls-glm-poisson" glm poisson with log link pls models
 "pls-glm-polr" glm polr with logit link pls models

Using the "family=" option and setting "modele=pls-glm-family" allows changing the family and link function the same way as for the `glm` function. As a consequence user-specified families can also be used.

The gaussian family accepts the links (as names) identity, log and inverse.

The binomial family accepts the links logit, probit, cauchit, (corresponding to logistic, normal and Cauchy CDFs respectively) log and cloglog (complementary log-log).

The Gamma family accepts the links inverse, identity and log.

The poisson family accepts the links log, identity, and sqrt.

The inverse.gaussian family accepts the links $1/\mu^2$, inverse, identity and log.

The quasi family accepts the links logit, probit, cloglog, identity, inverse, log, $1/\mu^2$ and sqrt.

The function `power` can be used to create a power link function.

Non-NULL weights can be used to indicate that different observations have different dispersions (with the values in weights being inversely proportional to the dispersions); or equivalently, when the elements of weights are positive integers w_i , that each response y_i is the mean of w_i unit-weight observations.

Value

`results_kfolds` list of NK. Each element of the list sums up the results for a group division:
 list of K matrices of size about $nrow(\text{dataX})/K * nt$ with the predicted values for a growing number of components

 list of K matrices of size about $nrow(\text{dataX})/K * nt$ with the predicted values for a growing number of components

`folders` list of NK. Each element of the list sums up the informations for a group division:
 list of K vectors of length about $nrow(\text{dataX})$ with the numbers of the rows of `dataX` that were used as a training set

 list of K vectors of length about $nrow(\text{dataX})$ with the numbers of the rows of `dataX` that were used as a training set

`dataY_kfolds` list of NK. Each element of the list sums up the results for a group division:
 list of K matrices of size about $nrow(\text{dataX})/K * 1$ with the observed values of the response


```

... ...
list of K matrices of size about nrow(dataX)/K * 1 with the observed values
of the response
call the call of the function

```

Note

Works for complete and incomplete datasets.

Author(s)

Frédéric Bertrand
 <frederic.bertrand@math.unistra.fr>
<http://www-irma.u-strasbg.fr/~fbertran/>

References

Frédéric Bertrand, Nicolas Meyer, Michèle Beau-Faller, Karim El Bayed, Izzie-Jacques Namer, Myriam Maumy-Bertrand (2013). Régression Bâta PLS. *Journal de la SociÃ©tÃ© FranÃ§aise de Statistique*, **154**(3):143-159. http://smf4.emath.fr/Publications/JSFds/154_3/html/

See Also

[kfolds2coeff](#), [kfolds2Pressind](#), [kfolds2Press](#), [kfolds2Mclassedind](#), [kfolds2Mclassed](#) and [kfolds2CVinfos_beta](#) to extract and transform results from kfold cross validation.

Examples

```

## Not run:
data("GasolineYield", package="betareg")
yGasolineYield <- GasolineYield$yield
XGasolineYield <- GasolineYield[,2:5]
bbb <- PLS_beta_kfoldcv(yGasolineYield, XGasolineYield, nt=3, modele="pls-beta")
kfolds2CVinfos_beta(bbb)

## End(Not run)

```

PLS_beta_kfoldcv_formula

Partial least squares regression beta models with kfold cross validation

Description

This function implements kfold cross validation on complete or incomplete datasets for partial least squares regression generalized linear models (formula specification of the model).

Usage

```
PLS_beta_kfoldcv_formula(formula, data=NULL, nt=2, limQ2set=.0975,
  modele="pls", family=NULL, K=nrow(dataX), NK=1, grouplist=NULL,
  random=FALSE, scaleX=TRUE, scaleY=NULL, keepcoeffs=FALSE,
  keepfolds=FALSE, keepdataY=TRUE, keepMclassed=FALSE, tol_Xi=10^(-12),
  weights, subset, start=NULL, etastart, mustart, offset, method, control= list(),
  contrasts=NULL, sparse=FALSE, sparseStop=TRUE, naive=FALSE, link=NULL,
  link.phi=NULL, type="ML")
```

Arguments

formula	an object of class " formula " (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
data	an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which <code>plsRglm</code> is called.
nt	number of components to be extracted
limQ2set	limit value for the Q2
modele	name of the PLS glm or PLS beta model to be fitted ("pls", "pls-glm-Gamma", "pls-glm-gaussian", "pls-glm-inverse.gaussian", "pls-glm-logistic", "pls-glm-poisson", "pls-glm-polr", "pls-beta"). Use "modele=pls-glm-family" to enable the family option.
family	a description of the error distribution and link function to be used in the model. This can be a character string naming a family function, a family function or the result of a call to a family function. (See family for details of family functions.) To use the family option, please set <code>modele="pls-glm-family"</code> . User defined families can also be defined. See details.
K	number of groups
NK	number of times the group division is made
grouplist	to specify the members of the K groups
random	should the K groups be made randomly
scaleX	scale the predictor(s) : must be set to TRUE for <code>modele="pls"</code> and should be for glms pls.
scaleY	scale the response : Yes/No. Ignored since non always possible for glm responses.
keepcoeffs	shall the coefficients for each model be returned
keepfolds	shall the groups' composition be returned
keepdataY	shall the observed value of the response for each one of the predicted value be returned
keepMclassed	shall the number of miss classed be returned (unavailable)
tol_Xi	minimal value for $\text{Norm2}(X_i)$ and $\det(pp' \times pp)$ if there is any missing value in the dataX. It defaults to 10^{-12}

weights	an optional vector of 'prior weights' to be used in the fitting process. Should be NULL or a numeric vector.
subset	an optional vector specifying a subset of observations to be used in the fitting process.
start	starting values for the parameters in the linear predictor.
etastart	starting values for the linear predictor.
mustart	starting values for the vector of means.
offset	this can be used to specify an <i>a priori</i> known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector of length equal to the number of cases. One or more <code>offset</code> terms can be included in the formula instead or as well, and if more than one is specified their sum is used. See <code>model.offset</code> .
method	for fitting glms with glm ("pls-glm-Gamma", "pls-glm-gaussian", "pls-glm-inverse.gaussian", the method to be used in fitting the model. The default method "glm.fit" uses iteratively reweighted least squares (IWLS). User-supplied fitting functions can be supplied either as a function or a character string naming a function, with a function which takes the same arguments as <code>glm.fit</code> . If "model.frame", the model frame is returned. pls-glm-polr logistic, probit, complementary log-log or cauchit (corresponding to a Cauchy latent variable).
control	a list of parameters for controlling the fitting process. For <code>glm.fit</code> this is passed to <code>glm.control</code> .
contrasts	an optional list. See the <code>contrasts.arg</code> of <code>model.matrix.default</code> .
sparse	should the coefficients of non-significant predictors (<code><alpha.pvals.expli</code>) be set to 0
sparseStop	should component extraction stop when no significant predictors (<code><alpha.pvals.expli</code>) are found
naive	Use the naive estimates for the Degrees of Freedom in plsR? Default is FALSE.
link	character specification of the link function in the mean model (μ). Currently, "logit", "probit", "cloglog", "cauchit", "log", "loglog" are supported. Alternatively, an object of class "link-glm" can be supplied.
link.phi	character specification of the link function in the precision model (ϕ). Currently, "identity", "log", "sqrt" are supported. The default is "log" unless formula is of type $y \sim x$ where the default is "identity" (for backward compatibility). Alternatively, an object of class "link-glm" can be supplied.
type	character specification of the type of estimator. Currently, maximum likelihood ("ML"), ML with bias correction ("BC"), and ML with bias reduction ("BR") are supported.

Details

Predicts 1 group with the K-1 other groups. Leave one out cross validation is thus obtained for $K = \text{nrow}(\text{dataX})$.

There are seven different predefined models with predefined link functions available :

"pls" ordinary pls models

"pls-glm-Gamma" glm gaussian with inverse link pls models

"pls-glm-gaussian" glm gaussian with identity link pls models

"pls-glm-inverse-gamma" glm binomial with square inverse link pls models

"pls-glm-logistic" glm binomial with logit link pls models

"pls-glm-poisson" glm poisson with log link pls models

"pls-glm-polr" glm polr with logit link pls models

Using the "family=" option and setting "modele=pls-glm-family" allows changing the family and link function the same way as for the `glm` function. As a consequence user-specified families can also be used.

The gaussian family accepts the links (as names) identity, log and inverse.

The binomial family accepts the links logit, probit, cauchit, (corresponding to logistic, normal and Cauchy CDFs respectively) log and cloglog (complementary log-log).

The Gamma family accepts the links inverse, identity and log.

The poisson family accepts the links log, identity, and sqrt.

The inverse.gaussian family accepts the links $1/\mu^2$, inverse, identity and log.

The quasi family accepts the links logit, probit, cloglog, identity, inverse, log, $1/\mu^2$ and sqrt.

The function `power` can be used to create a power link function.

A typical predictor has the form `response ~ terms` where `response` is the (numeric) response vector and `terms` is a series of terms which specifies a linear predictor for response. A terms specification of the form `first + second` indicates all the terms in `first` together with all the terms in `second` with any duplicates removed.

A specification of the form `first:second` indicates the the set of terms obtained by taking the interactions of all terms in `first` with all terms in `second`. The specification `first*second` indicates the cross of `first` and `second`. This is the same as `first + second + first:second`.

The terms in the formula will be re-ordered so that main effects come first, followed by the interactions, all second-order, all third-order and so on: to avoid this pass a terms object as the formula.

Non-NULL weights can be used to indicate that different observations have different dispersions (with the values in weights being inversely proportional to the dispersions); or equivalently, when the elements of weights are positive integers w_i , that each response y_i is the mean of w_i unit-weight observations.

Value

`results_kfolds` list of NK . Each element of the list sums up the results for a group division:

- list of K matrices of size about $nrow(\text{dataX})/K * nt$ with the predicted values for a growing number of components
- ...
- list of K matrices of size about $nrow(\text{dataX})/K * nt$ with the predicted values for a growing number of components

<p>folds</p> <p>...</p>	<p>list of NK. Each element of the list sums up the informations for a group division:</p> <p>list of K vectors of length about $nrow(dataX)$ with the numbers of the rows of $dataX$ that were used as a training set</p> <p>...</p> <p>list of K vectors of length about $nrow(dataX)$ with the numbers of the rows of $dataX$ that were used as a training set</p>
<p>dataY_kfolds</p> <p>...</p>	<p>list of NK. Each element of the list sums up the results for a group division:</p> <p>list of K matrices of size about $nrow(dataX)/K * 1$ with the observed values of the response</p> <p>...</p> <p>list of K matrices of size about $nrow(dataX)/K * 1$ with the observed values of the response</p>
<p>call</p>	<p>the call of the function</p>

Note

Work for complete and incomplete datasets.

Author(s)

Frédéric Bertrand
 <frederic.bertrand@math.unistra.fr>
<http://www-irma.u-strasbg.fr/~fbertran/>

References

Frédéric Bertrand, Nicolas Meyer, Michèle Beau-Faller, Karim El Bayed, Izzie-Jacques Namer, Myriam Maumy-Bertrand (2013). Régression Bata PLS. *Journal de la Société Française de Statistique*, **154**(3):143-159. http://smf4.emath.fr/Publications/JSFds/154_3/html/

See Also

[kfolds2coeff](#), [kfolds2Pressind](#), [kfolds2Press](#), [kfolds2Mclassifiedind](#), [kfolds2Mclassified](#) and [kfolds2CVinfos_beta](#) to extract and transform results from kfold cross validation.

Examples

```
## Not run:
data("GasolineYield", package="betareg")
bbb <- PLS_beta_kfoldcv_formula(yield~, data=GasolineYield, nt=3, modele="pls-beta")
kfolds2CVinfos_beta(bbb)

## End(Not run)
```

 PLS_beta_wvc

Light version of PLS_beta for cross validation purposes

Description

Light version of PLS_beta for cross validation purposes either on complete or incomplete datasets.

Usage

```
PLS_beta_wvc(dataY, dataX, nt = 2, dataPredictY = dataX, modele = "pls",
family = NULL, scaleX = TRUE, scaleY = NULL, keepcoeffs = FALSE,
keepstd.coeffs=FALSE, tol_Xi = 10^(-12), weights, method = "logistic",
link=NULL,link.phi=NULL,type="ML")
```

Arguments

dataY	response (training) dataset
dataX	predictor(s) (training) dataset
nt	number of components to be extracted
dataPredictY	predictor(s) (testing) dataset
modele	name of the PLS glm or PLS beta model to be fitted ("pls", "pls-glm-Gamma", "pls-glm-gaussian", "pls-glm-inverse.gaussian", "pls-glm-logistic", "pls-glm-poisson", "pls-glm-polr", "pls-beta"). Use "modele=pls-glm-family" to enable the family option.
family	a description of the error distribution and link function to be used in the model. This can be a character string naming a family function, a family function or the result of a call to a family function. (See family for details of family functions.) To use the family option, please set modele="pls-glm-family". User defined families can also be defined. See details.
scaleX	scale the predictor(s) : must be set to TRUE for modele="pls" and should be for glms pls.
scaleY	scale the response : Yes/No. Ignored since non always possible for glm responses.
keepcoeffs	whether the coefficients of the linear fit on link scale of unstandardized explanatory variables should be returned or not.
keepstd.coeffs	whether the coefficients of the linear fit on link scale of standardized explanatory variables should be returned or not.
tol_Xi	minimal value for Norm2(Xi) and $\det(pp' \times pp)$ if there is any missing value in the dataX. It defaults to 10^{-12}
weights	an optional vector of 'prior weights' to be used in the fitting process. Should be NULL or a numeric vector.
method	logistic, probit, complementary log-log or cauchit (corresponding to a Cauchy latent variable).

link	character specification of the link function in the mean model (μ). Currently, "logit", "probit", "cloglog", "cauchit", "log", "loglog" are supported. Alternatively, an object of class "link-glm" can be supplied.
link.phi	character specification of the link function in the precision model (ϕ). Currently, "identity", "log", "sqrt" are supported. The default is "log" unless formula is of type $y \sim x$ where the default is "identity" (for backward compatibility). Alternatively, an object of class "link-glm" can be supplied.
type	character specification of the type of estimator. Currently, maximum likelihood ("ML"), ML with bias correction ("BC"), and ML with bias reduction ("BR") are supported.

Details

This function is called by [PLS_glm_kfoldcv_formula](#) in order to perform cross validation either on complete or incomplete datasets.

There are seven different predefined models with predefined link functions available :

"pls" ordinary pls models

"pls-glm-Gamma" glm gaussian with inverse link pls models

"pls-glm-gaussian" glm gaussian with identity link pls models

"pls-glm-inverse-gamma" glm binomial with square inverse link pls models

"pls-glm-logistic" glm binomial with logit link pls models

"pls-glm-poisson" glm poisson with log link pls models

"pls-glm-polr" glm polr with logit link pls models

Using the "family=" option and setting "modele=pls-glm-family" allows changing the family and link function the same way as for the [glm](#) function. As a consequence user-specified families can also be used.

The gaussian family accepts the links (as names) identity, log and inverse.

The binomial family accepts the links logit, probit, cauchit, (corresponding to logistic, normal and Cauchy CDFs respectively) log and cloglog (complementary log-log).

The Gamma family accepts the links inverse, identity and log.

The poisson family accepts the links log, identity, and sqrt.

The inverse.gaussian family accepts the links $1/\mu^2$, inverse, identity and log.

The quasi family accepts the links logit, probit, cloglog, identity, inverse, log, $1/\mu^2$ and sqrt.

The function power can be used to create a power link function.

Non-NULL weights can be used to indicate that different observations have different dispersions (with the values in weights being inversely proportional to the dispersions); or equivalently, when the elements of weights are positive integers w_i , that each response y_i is the mean of w_i unit-weight observations.

Value

valsPredict nrow(dataPredictY) * nt matrix of the predicted values
 coeffs If the coefficients of the eXplanatory variables were requested:
 i.e. keepcoeffs=TRUE.
 ncol(dataX) * 1 matrix of the coefficients of the the eXplanatory variables

Author(s)

Frédéric Bertrand
 <frederic.bertrand@math.unistra.fr>
<http://www-irma.u-strasbg.fr/~fbertran/>

References

Frédéric Bertrand, Nicolas Meyer, Michèle Beau-Faller, Karim El Bayed, Izzie-Jacques Namer, Myriam Maumy-Bertrand (2013). Régression Bâta PLS. *Journal de la Société Française de Statistique*, **154**(3):143-159. http://smf4.emath.fr/Publications/JSFds/154_3/html/

See Also

[PLS_beta](#) for more detailed results, [PLS_beta_kfoldcv](#) for cross validating models and [PLS_lm_wvc](#) for the same function dedicated to plsR models

Examples

```
data("GasolineYield",package="betareg")
yGasolineYield <- GasolineYield$yield
XGasolineYield <- GasolineYield[,2:5]
modpls <- PLS_beta_wvc(yGasolineYield,XGasolineYield,nt=3,modele="pls-beta")
modpls
rm("modpls")
```

print.plsRbetamodel *Print method for plsRbeta models*

Description

This function provides a print method for the class "plsRbetamodel"

Usage

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


Arguments

x an object of the class "plsRbetamodel"
 ... not used

Value

NULL

Author(s)

Frédéric Bertrand
 <frederic.bertrand@math.unistra.fr>
<http://www-irma.u-strasbg.fr/~fbertran/>

References

Frédéric Bertrand, Nicolas Meyer, Michèle Beau-Faller, Karim El Bayed, Izzie-Jacques Namer, Myriam Maumy-Bertrand (2013). Régression à la PLS. *Journal de la Société Française de Statistique*, **154**(3):143-159. http://smf4.emath.fr/Publications/JSFds/154_3/html/

See Also

[print](#)

Examples

```
data("GasolineYield", package="betareg")
modpls <- plsRbeta(yield~., data=GasolineYield, nt=3, modele="pls-beta")
print(modpls)
```

```
print.summary.plsRbetamodel
```

Print method for summaries of plsRbeta models

Description

This function provides a print method for the class "summary.plsRbetamodel"

Usage

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

Arguments

x an object of the class "summary.plsRbetamodel"
 ... not used

Value

language call of the model

Author(s)

Frédéric Bertrand
 <frederic.bertrand@math.unistra.fr>
<http://www-irma.u-strasbg.fr/~fbertran/>

References

Frédéric Bertrand, Nicolas Meyer, Michèle Beau-Faller, Karim El Bayed, Izzie-Jacques Namer, Myriam Maumy-Bertrand (2013). Régression à PLS. *Journal de la Société Française de Statistique*, **154**(3):143-159. http://smf4.emath.fr/Publications/JSFds/154_3/html/

See Also

[print](#) and [summary](#)

Examples

```
data("GasolineYield", package="betareg")
modpls <- plsRbeta(yield~., data=GasolineYield, nt=3, modele="pls-beta")
print(summary(modpls))
```

simul_data_UniYX_beta *Data generating function for univariate beta plsR models*

Description

This function generates a single univariate rate response value Y and a vector of explanatory variables (X_1, \dots, X_{totdim}) drawn from a model with a given number of latent components.

Usage

```
simul_data_UniYX_beta(totdim, ncomp, disp=1, link="logit", type="a", phi0=20)
```

Arguments

totdim	Number of columns of the X vector (from ncomp to hardware limits)
ncomp	Number of latent components in the model (from 2 to 6)
disp	Tune the shape of the beta distribution (defaults to 1)
link	Character specification of the link function in the mean model (μ). Currently, "logit", "probit", "cloglog", "cauchit", "log", "loglog" are supported. Alternatively, an object of class "link-glm" can be supplied.
type	Simulation scheme
phi0	Simulation scheme "a" parameter

Details

This function should be combined with the replicate function to give rise to a larger dataset. The algorithm used is a modification of a R port of the one described in the article of Li which is a multivariate generalization of the algorithm of Naes and Martens.

Value

vector $(Y, X_1, \dots, X_{totdim})$

Author(s)

Frédéric Bertrand
 <frederic.bertrand@math.unistra.fr>
<http://www-irma.u-strasbg.fr/~fbertran/>

References

- Frédéric Bertrand, Nicolas Meyer, Michèle Beau-Faller, Karim El Bayed, Izzie-Jacques Namer, Myriam Maumy-Bertrand (2013). Régression à la PLS. *Journal de la Société Française de Statistique*, **154**(3):143-159. http://smf4.emath.fr/Publications/JSFds/154_3/html/
- T. Naes, H. Martens (1985). Comparison of prediction methods for multicollinear data. *Commun. Stat., Simul.*, **14**:545-576. <http://dx.doi.org/10.1080/03610918508812458>
- Baibing Li, Julian Morris, Elaine B. Martin (2002). Model selection for partial least squares regression, *Chemometrics and Intelligent Laboratory Systems*, **64**:79-89. [http://dx.doi.org/10.1016/S0169-7439\(02\)00051-5](http://dx.doi.org/10.1016/S0169-7439(02)00051-5)

See Also

[simul_data_UniYX](#)

Examples

```
# logit link
layout(matrix(1:4,nrow=2))
hist(t(replicate(100,simul_data_UniYX_beta(4,4)))[,1])
hist(t(replicate(100,simul_data_UniYX_beta(4,4,disp=3)))[,1])
hist(t(replicate(100,simul_data_UniYX_beta(4,4,disp=5)))[,1])
hist(t(replicate(100,simul_data_UniYX_beta(4,4,disp=15)))[,1])
layout(1)

# probit link
layout(matrix(1:4,nrow=2))
hist(t(replicate(100,simul_data_UniYX_beta(4,4,link="probit")))[,1])
hist(t(replicate(100,simul_data_UniYX_beta(4,4,disp=3,link="probit")))[,1])
hist(t(replicate(100,simul_data_UniYX_beta(4,4,disp=5,link="probit")))[,1])
hist(t(replicate(100,simul_data_UniYX_beta(4,4,disp=15,link="probit")))[,1])
layout(1)
```

```

# cloglog link
layout(matrix(1:4,nrow=2))
hist(t(replicate(100,simul_data_UniYX_beta(4,4,link="cloglog")))[,1])
hist(t(replicate(100,simul_data_UniYX_beta(4,4,disp=3,link="cloglog")))[,1])
hist(t(replicate(100,simul_data_UniYX_beta(4,4,disp=5,link="cloglog")))[,1])
hist(t(replicate(100,simul_data_UniYX_beta(4,4,disp=15,link="cloglog")))[,1])
layout(1)

# cauchit link
layout(matrix(1:4,nrow=2))
hist(t(replicate(100,simul_data_UniYX_beta(4,4,link="cauchit")))[,1])
hist(t(replicate(100,simul_data_UniYX_beta(4,4,disp=3,link="cauchit")))[,1])
hist(t(replicate(100,simul_data_UniYX_beta(4,4,disp=5,link="cauchit")))[,1])
hist(t(replicate(100,simul_data_UniYX_beta(4,4,disp=15,link="cauchit")))[,1])
layout(1)

# loglog link
layout(matrix(1:4,nrow=2))
hist(t(replicate(100,simul_data_UniYX_beta(4,4,link="loglog")))[,1])
hist(t(replicate(100,simul_data_UniYX_beta(4,4,disp=3,link="loglog")))[,1])
hist(t(replicate(100,simul_data_UniYX_beta(4,4,disp=5,link="loglog")))[,1])
hist(t(replicate(100,simul_data_UniYX_beta(4,4,disp=15,link="loglog")))[,1])
layout(1)

# log link
layout(matrix(1:4,nrow=2))
hist(t(replicate(100,simul_data_UniYX_beta(4,4,link="log")))[,1])
hist(t(replicate(100,simul_data_UniYX_beta(4,4,disp=3,link="log")))[,1])
hist(t(replicate(100,simul_data_UniYX_beta(4,4,disp=5,link="log")))[,1])
hist(t(replicate(100,simul_data_UniYX_beta(4,4,disp=15,link="log")))[,1])
layout(1)

```

summary.plsRbetamodel *Summary method for plsRbeta models*

Description

This function provides a summary method for the class "plsRbetamodel"

Usage

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

Arguments

object an object of the class "plsRbetamodel"
 ... further arguments to be passed to or from methods.

Value

call function call of plsR beta models

Author(s)

Frédéric Bertrand
 <frederic.bertrand@math.unistra.fr>
<http://www-irma.u-strasbg.fr/~fbertran/>

References

Frédéric Bertrand, Nicolas Meyer, Michèle Beau-Faller, Karim El Bayed, Izzie-Jacques Namer, Myriam Maumy-Bertrand (2013). Régression Bâta PLS. *Journal de la Société Française de Statistique*, **154**(3):143-159. http://smf4.emath.fr/Publications/JSFds/154_3/html/

See Also

[summary](#)

Examples

```
data("GasolineYield", package="betareg")
modpls <- plsRbeta(yield~., data=GasolineYield, nt=3, modele="pls-beta")
summary(modpls)
```

tilt.bootplsbeta *Tilted bootstrap for PLS models*

Description

~~ A (1-5 lines) description of what the function does. ~~

Usage

```
tilt.bootplsbeta(object, typeboot="plsmodel", statistic=coefs.plsRbeta,
R=c(499, 250, 250), alpha=c(0.025, 0.975), sim="ordinary", stype="i",
index=1)
```

Arguments

object ~~Explain object here~~
 typeboot ~~Explain typeboot here~~
 statistic ~~Explain statistic here~~
 R ~~Explain R here~~
 alpha ~~Explain alpha here~~

sim ~~Explain sim here~~
 stype ~~Explain stype here~~
 index ~~Explain index here~~

Details

~~ More details than the description above ~~

Value

~Describe the value returned If it is a LIST, use

comp1 Description of 'comp1'
 comp2 Description of 'comp2'
 ...

Note

~~some notes~~

Author(s)

Frédéric Bertrand
 <frederic.bertrand@math.unistra.fr>
<http://www-irma.u-strasbg.fr/~fbertran/>

References

Frédéric Bertrand, Nicolas Meyer, Michèle Beau-Faller, Karim El Bayed, Izzie-Jacques Namer, Myriam Maumy-Bertrand (2013). Régression à la PLS. *Journal de la Société Française de Statistique*, **154**(3):143-159. http://smf4.emath.fr/Publications/JSFds/154_3/html/

See Also

[tilt.boot](#)

Examples

```
## Not run:
data("GasolineYield",package="betareg")

GazYield.tilt.boot <- tilt.bootplsbeta(plsRbeta(yield~.,data=GasolineYield,nt=3,
modele="pls-beta"), statistic=coefs.plsRbeta, R=c(499, 100, 100),
alpha=c(0.025, 0.975), sim="balanced", stype="i", index=1)
boxplots.bootpls(GazYield.tilt.boot,1:2)

## End(Not run)
```

Index

- *Topic **datagen**
 - simul_data_UniYX_beta, 34
- *Topic **methods**
 - print.plsRbetamodel, 32
 - print.summary.plsRbetamodel, 33
 - summary.plsRbetamodel, 36
- *Topic **models**
 - bootplsbeta, 2
 - coefs.plsRbeta, 4
 - kfolds2Chisq, 6
 - kfolds2Chisqind, 7
 - kfolds2CVinfos_beta, 8
 - permcoefs.plsRbeta, 9
 - PLS_beta, 15
 - PLS_beta_formula, 18
 - PLS_beta_kfoldcv, 22
 - PLS_beta_kfoldcv_formula, 25
 - PLS_beta_wvc, 30
 - plsRbeta, 11
 - tilt.bootplsbeta, 37
- *Topic **print**
 - print.plsRbetamodel, 32
 - print.summary.plsRbetamodel, 33
 - summary.plsRbetamodel, 36
- *Topic **regression**
 - kfolds2Chisq, 6
 - kfolds2Chisqind, 7
 - kfolds2CVinfos_beta, 8
 - PLS_beta, 15
 - PLS_beta_formula, 18
 - PLS_beta_kfoldcv, 22
 - PLS_beta_kfoldcv_formula, 25
 - PLS_beta_wvc, 30
 - plsRbeta, 11
- *Topic **utilities**
 - simul_data_UniYX_beta, 34
- as.data.frame, 11, 19, 26
- boot, 3
- bootplsbeta, 2
- coefs.plsRbeta, 4
- family, 12, 16, 19, 23, 26, 30
- formula, 11, 19, 26
- glm, 13, 17, 21, 24, 28, 31
- glm.control, 12, 20, 27
- help, 5, 10
- kfolds2Chisq, 6, 8
- kfolds2Chisqind, 6, 7
- kfolds2coeff, 6, 8, 9, 25, 29
- kfolds2CVinfos_beta, 8, 25, 29
- kfolds2Mclassified, 6, 8, 9, 25, 29
- kfolds2Mclassifiedind, 6, 8, 9, 25, 29
- kfolds2Press, 6, 8, 9, 25, 29
- kfolds2Pressind, 6, 8, 9, 25, 29
- model.offset, 12, 20, 27
- offset, 12, 20, 27
- permcoefs.plsRbeta, 9
- PLS_beta, 15, 32
- PLS_beta_formula, 18
- PLS_beta_kfoldcv, 6–8, 18, 22, 32
- PLS_beta_kfoldcv_formula, 22, 25
- PLS_beta_wvc, 18, 22, 30
- PLS_glm_kfoldcv_formula, 31
- PLS_lm_wvc, 32
- plsR, 14
- plsRbeta, 11
- plsRbetamodel.default(plsRbeta), 11
- plsRbetamodel.formula(plsRbeta), 11
- plsRglm, 14
- print, 33, 34
- print.plsRbetamodel, 32
- print.summary.plsRbetamodel, 33

`simul_data_UniYX`, [35](#)
`simul_data_UniYX_beta`, [34](#)
`summary`, [34](#), [37](#)
`summary.plsRbetamodel`, [36](#)

`tilt.boot`, [38](#)
`tilt.bootplsbeta`, [37](#)