Package ‘FAMILY’

June 21, 2015

Type Package

Title A Convex Formulation for Modeling Interactions with Strong Heredity

Version 0.1.19

Date 2015-06-20

Author Asad Haris

Maintainer Asad Haris <aharis@uw.edu>

Description Fits penalized linear and logistic regression models with pairwise interaction terms.

License GPL (>= 2)

Imports pheatmap, pROC

NeedsCompilation no

Repository CRAN

Date/Publication 2015-06-21 13:05:26

R topics documented:

`FAMILY-package` *A Convex Formulation for Modeling Interactions with Strong Heredity.*

Description

Fits linear and logistic regression models with pairwise interactions for two sets of covariates. The models fitted by this package obey strong heredity which implies that interaction of two covariates is only included if both main effects are included in the model. In the case of high dimensional data we may not wish to model all possible covariates and their interactions but a small subset of them, thus we utilize the penalized regression methods of Haris, Witten and Simon (2014).

Details

Package:	FAMILY
Type:	Package
Version:	0.1.19
Date:	2015-06-20
License:	GPL-2

The package includes the following functions:

`FAMILY`:	Fit a penalized regression model for a grid of α and λ values
`predict.FAMILY`:	Predict `yhat` or `phat` for a for a given data set from the fitted model
`coef.FAMILY`:	Extract the set of estimated non-zero coefficients

Author(s)

Asad Haris, Daniela Witten and Noah Simon

Maintainer: Asad Haris <aharis@uw.edu>

References

Haris, Witten and Simon (2014). Convex Modeling of Interactions with Strong Heredity. Available on ArXiv at http://arxiv.org/abs/1410.3517

See Also

`FAMILY`, `predict.FAMILY`, `coef.FAMILY`

Examples

```
library(FAMILY)
library(pROC)
library(pheatmap)

#########################################################################################
#########################################################################################
############################## EXAMPLE - CONTINUOUS RESPONSE ###########################
#########################################################################################
#########################################################################################

############################### GENERATE DATA ##########################################

#Generate training set of covariates X and Z
set.seed(1)
X.tr<- matrix(rnorm(10*100),ncol = 10, nrow = 100)
Z.tr<- matrix(rnorm(15*100),ncol = 15, nrow = 100)
```

```
#Generate test set of covariates X and Z
X.te<- matrix(rnorm(10*100),ncol = 10, nrow = 100)
Z.te<- matrix(rnorm(15*100),ncol = 15, nrow = 100)

#Scale appropiately
meanX<- apply(X.tr,2,mean)
meanY<- apply(Z.tr,2,mean)

X.tr<- scale(X.tr, scale = FALSE)
Z.tr<- scale(Z.tr, scale = FALSE)
X.te<- scale(X.te,center = meanX,scale = FALSE)
Z.te<- scale(Z.te,center = meanY,scale = FALSE)

#Generate full matrix of Covariates
w.tr<- c()
w.te<- c()
X1<- cbind(1,X.tr)
Z1<- cbind(1,Z.tr)
X2<- cbind(1,X.te)
Z2<- cbind(1,Z.te)

for(i in 1:16){
  for(j in 1:11){
    w.tr<- cbind(w.tr,X1[,j]*Z1[,i])
    w.te<- cbind(w.te, X2[,j]*Z2[,i])
  }
}

#Generate response variables with signal from
#First 5 X features and 5 Z features.

#We construct the coefficient matrix B.
#B[1,1] contains the intercept
#B[-1,1] contains the main effects for X.
# For instance, B[2,1] is the main effect for the first feature in X.
#B[1,-1] contains the main effects for Z.
# For instance, B[1,10] is the coefficient for the 10th feature in Z.
#B[i+1,j+1] is the coefficient of X_i Z_j
B<- matrix(0,ncol = 16,nrow = 11)
rownames(B)<- c("inter" , paste("X",1:(nrow(B)-1),sep = ""))
colnames(B)<- c("inter" , paste("Z",1:(ncol(B)-1),sep = ""))

# First, we simulate data as follows:
# The first five features in X, and the first five features in Z, are non-zero.
# And given the non-zero main effects, all possible interactions are involved.
# We call this "high strong heredity"
B_high_SH<- B
B_high_SH[1:6,1:6]<- 1
#View true coefficient matrix
pheatmap(as.matrix(B_high_SH), scale="none",
         cluster_rows=FALSE, cluster_cols=FALSE)

Y_high_SH <- as.vector(w.tr%*%as.vector(B_high_SH))+rnorm(100,sd = 2)
```

```
Y_high_SH.te <- as.vector(w.te%*%as.vector(B_high_SH))+rnorm(100,sd = 2)

# Now a new setting:
# Again, the first five features in X, and the first five features in Z, are involved.
# But this time, only a subset of the possible interactions are involved.
# Strong heredity is still maintained.
# We call this "low strong heredity"
B_low_SH<- B_high_SH
B_low_SH[2:6,2:6]<-0
B_low_SH[3:4,3:5]<- 1
#View true coefficient matrix
pheatmap(as.matrix(B_low_SH), scale="none",
         cluster_rows=FALSE, cluster_cols=FALSE)
Y_low_SH <- as.vector(w.tr%*%as.vector(B_low_SH))+rnorm(100,sd = 1.5)
Y_low_SH.te <- as.vector(w.te%*%as.vector(B_low_SH))+rnorm(100,sd = 1.5)


############################## FIT SOME MODELS ########################################

#Define alphas and lambdas
#Define 3 different alpha values
#Low alpha values penalize groups more
#High alpha values penalize individual Interactions more
alphas<- c(0.01,0.5,0.99)
lambdas<- seq(0.1,1,length = 50)

#high Strong heredity with l2 norm
fit_high_SH<- FAMILY(X.tr, Z.tr, Y_high_SH, lambdas ,
                     alphas, quad = TRUE,iter=500, verbose = TRUE )
yhat_hSH<- predict(fit_high_SH, X.te, Z.te)
mse_hSH <-apply(yhat_hSH,c(2,3), "-" ,Y_high_SH.te)
mse_hSH<- apply(mse_hSH^2,c(2,3),sum)

#Find optimal model and plot matrix
im<- which(mse_hSH==min(mse_hSH),TRUE)
plot(fit_high_SH$Estimate[[im[2] ]][[im[1]]])


#Plot some matrices for different alpha values
#Low alpha, higher penalty on groups
plot(fit_high_SH$Estimate[[ 1 ]][[ 25 ]])
#Medium alpha, equal penalty on groups and individual interactions
plot(fit_high_SH$Estimate[[ 2 ]][[ 25  ]])
#High alpha, more penalty on individual interactions
plot(fit_high_SH$Estimate[[ 3 ]][[ 40 ]])


#View Coefficients
coef(fit_high_SH)[[im[2]]][[im[1]]]

############################## Uncomment code for EXAMPLE ###########################
# #high Strong heredity with l_infinity norm norm
# fit_high_SH<- FAMILY(X.tr, Z.tr, Y_high_SH, lambdas ,
#                      alphas, quad = TRUE,iter=500, verbose = TRUE,
#                      norm = "l_inf")
# yhat_hSH<- predict(fit_high_SH, X.te, Z.te)
# mse_hSH <-apply(yhat_hSH,c(2,3), "-" ,Y_high_SH.te)
# mse_hSH<- apply(mse_hSH^2,c(2,3),sum)
#
# #Find optimal model and plot matrix
# im<- which(mse_hSH==min(mse_hSH),TRUE)
# plot(fit_high_SH$Estimate[[im[2] ]][[im[1]]])
#
#
# #Plot some matrices for different alpha values
# #Low alpha, higher penalty on groups
# plot(fit_high_SH$Estimate[[ 1 ]][[ 30 ]])
# #Medium alpha, equal penalty on groups and individual interactions
# plot(fit_high_SH$Estimate[[ 2 ]][[ 10 ]])
# #High alpha, more penalty on individual interactions
# plot(fit_high_SH$Estimate[[ 3 ]][[ 20 ]])
#
#
# #View Coefficients
# coef(fit_high_SH)[[im[2]]][[im[1]]]


############################## Uncomment code for EXAMPLE ##########################
# #Redefine lambdas
# lambdas<- seq(0.1,0.5,length = 50)
#
# #low Strong heredity with l_2 norm
# fit_low_SH<- FAMILY(X.tr, Z.tr, Y_low_SH, lambdas ,
#                      alphas, quad = TRUE,iter=500, verbose = TRUE )
# yhat_lSH<- predict(fit_low_SH, X.te, Z.te)
# mse_lSH <-apply(yhat_lSH,c(2,3), "-" ,Y_low_SH.te)
# mse_lSH<- apply(mse_lSH^2,c(2,3),sum)
#
# #Find optimal model and plot matrix
# im<- which(mse_lSH==min(mse_lSH),TRUE)
# plot(fit_low_SH$Estimate[[im[2] ]][[im[1]]])
#
#
# #Plot some matrices for different alpha values
# #Low alpha, higher penalty on groups
# plot(fit_low_SH$Estimate[[ 1 ]][[ 25 ]])
# #Medium alpha, equal penalty on groups and individual interactions
# plot(fit_low_SH$Estimate[[ 2 ]][[ 10 ]])
# #High alpha, more penalty on individual interactions
# plot(fit_low_SH$Estimate[[ 3 ]][[ 10 ]])
#
#
# #View Coefficients
# coef(fit_low_SH)[[im[2]]][[im[1]]]
```

```
####################################################################################
####################################################################################
############################### EXAMPLE - BINARY RESPONSE ##########################
####################################################################################
####################################################################################

############################## GENERATE DATA ########################################

#Generate data for logistic regression
Yp_high_SH<- as.vector((w.tr)%*%as.vector(B_high_SH))
Yp_high_SH.te<- as.vector((w.te)%*%as.vector(B_high_SH))

Yprobs_high_SH<- 1/(1+exp(-Yp_high_SH))
Yprobs_high_SH.te<- 1/(1+exp(-Yp_high_SH.te))

Yp_high_SH<- rbinom(100, size = 1, prob = Yprobs_high_SH)
Yp_high_SH.te<- rbinom(100, size = 1, prob = Yprobs_high_SH.te)

lambdas<- seq(0.01,0.15,length = 50)

############################## FIT SOME MODELS ##########################################

#Fit glm via l_2 norm
fit_high_SH<- FAMILY(X.tr, Z.tr, Yp_high_SH, lambdas ,
                    alphas, quad = TRUE,iter=500, verbose = TRUE,
                    family = "binomial")
yhp_hSH<- predict(fit_high_SH, X.te, Z.te)
mse_high_SH <-apply(yhp_hSH,c(2,3), "-" ,Yp_high_SH.te)
mse_hSH<- apply(mse_high_SH^2,c(2,3),sum)
im<- which(mse_hSH==min(mse_hSH),TRUE)
plot(fit_high_SH$Estimate[[im[2] ]][[im[1]]])
roc( Yp_high_SH.te,yhp_hSH[,im[1],im[2]],plot = TRUE)

#View Coefficients
coef(fit_high_SH)[[im[2]]][[im[1]]]

############################## Uncomment code for EXAMPLE ###########################
# #Fit glm via l_infinity norm
# fit_high_SH<- FAMILY(X.tr, Z.tr, Yp_high_SH, lambdas , norm = "l_inf",
#                       alphas, quad = TRUE,iter=500, verbose = TRUE,
#                       family = "binomial")
# yhp_hSH<- predict(fit_high_SH, X.te, Z.te)
# mse_high_SH <-apply(yhp_hSH,c(2,3), "-" ,Yp_high_SH.te)
# mse_hSH<- apply(mse_high_SH^2,c(2,3),sum)
# im<- which(mse_hSH==min(mse_hSH),TRUE)
# plot(fit_high_SH$Estimate[[im[2] ]][[im[1]]])
# roc( Yp_high_SH.te,yhp_hSH[,im[1],im[2]],plot = TRUE)
#
# #View Coefficients
# coef(fit_high_SH)[[im[2]]][[im[1]]]

####################################################################################
####################################################################################
```

```
############################## EXAMPLE WHERE X=Z ####################################
######################## Uncomment Code for EXAMPLE ##################################
######################################################################################

############################## GENERATE DATA #########################################
# #Redefine Lambdas
# lambdas<- seq(0.01,0.3,length = 50)
#
#
# #We consider the case X=Z now
# w.tr<- c()
# w.te<- c()
# X1<- cbind(1,X.tr)
# X2<- cbind(1,X.te)
#
# for(i in 1:11){
#   for(j in 1:11){
#     w.tr<- cbind(w.tr,X1[,j]*X1[,i])
#     w.te<- cbind(w.te, X2[,j]*X2[,i])
#   }
# }
#
# B<- matrix(0,ncol = 11,nrow = 11)
# rownames(B)<- c("inter" , paste("X",1:(nrow(B)-1),sep = ""))
# colnames(B)<- c("inter" , paste("X",1:(ncol(B)-1),sep = ""))
#
#
# B_high_SH<- B
# B_high_SH[1:6,1:6]<- 1
# #We exclude quadratic terms in this example
# diag(B_high_SH)[-1]<-0
# #View true coefficient matrix
# pheatmap(as.matrix(B_high_SH), scale="none",
#          cluster_rows=FALSE, cluster_cols=FALSE)
#
# #With high Strong heredity: all possible interactions
# Y_high_SH <- as.vector(w.tr%*%as.vector(B_high_SH))+rnorm(100)
# Y_high_SH.te <- as.vector(w.te%*%as.vector(B_high_SH))+rnorm(100)
#
# ############################## FIT SOME MODELS #######################################
#
# #high Strong heredity with l_2 norm
# fit_high_SH<- FAMILY(X.tr, X.tr, Y_high_SH, lambdas ,
#                      alphas, quad = FALSE,iter=500, verbose = TRUE )
# yhat_hSH<- predict(fit_high_SH, X.te, X.te)
# mse_hSH <-apply(yhat_hSH,c(2,3), "-" ,Y_high_SH.te)
# mse_hSH<- apply(mse_hSH^2,c(2,3),sum)
#
# #Find optimal model and plot matrix
# im<- which(mse_hSH==min(mse_hSH),TRUE)
# plot(fit_high_SH$Estimate[[im[2] ]][[im[1]]])
#
#
```

```
# #Plot some matrices for different alpha values
# #Low alpha, higher penalty on groups
# plot(fit_high_SH$Estimate[[ 1 ]][[ 50 ]])
# #Medium alpha, equal penalty on groups and individual interactions
# plot(fit_high_SH$Estimate[[ 2 ]][[ 50 ]])
# #High alpha, more penalty on individual interactions
# plot(fit_high_SH$Estimate[[ 3 ]][[ 50 ]])
#
#
# #View Coefficients
# coef(fit_high_SH,XequalZ = TRUE)[[im[2]]][[im[1]]]
```

`coef.FAMILY` *coef.FAMILY*

Description

Similar to R's generic `coef` function which extracts the coefficients from the model for each value of `alpha` and `lambda`.

Usage

```
## S3 method for class 'FAMILY'
coef(object, XequalZ = FALSE, Bias.corr = FALSE, ...)
```

Arguments

`object`	The fitted object from the output of the main function `FAMILY`.
`XequalZ`	A logical variable indicating if $X = Z$.
`Bias.corr`	A logical variable indicating if we wish to re-fit the selected variables using `glm` or `lm`.
`...`	Extra arguments for the generic S3 `coef` function

Value

The function returns a list of dimensions `length(alphas)*length(lambdas)` where each element is a list with the following components:

`Intercept`	The estimated coefficient for the intercept of the model.
`mains, mainsX, mainsZ`	A matrix with two columns which show the selected non-zero coefficients in the model and the estimated coefficient. When `XequalsZ == TRUE` the function only returns the matrix `mains` and it returns `mainsX` and `mainsZ` otherwise.
`interacts`	The matrix showing the estimated non-zero interaction terms with corresponding coefficient estimates.
`alpha`	The alpha value used for the particular fitted model
`lambda`	The lambda value used for the particular fitted model

Author(s)

Asad Haris

References

Haris, Witten and Simon (2014). Convex Modeling of Interactions with Strong Heredity. Available on ArXiv at http://arxiv.org/abs/1410.3517

See Also

FAMILY

Examples

```
library(FAMILY)
library(pROC)
library(pheatmap)

#######################################################################################
#######################################################################################
############################# EXAMPLE - CONTINUOUS RESPONSE ###########################
#######################################################################################
#######################################################################################

############################## GENERATE DATA ##########################################

#Generate training set of covariates X and Z
set.seed(1)
X.tr<- matrix(rnorm(10*100),ncol = 10, nrow = 100)
Z.tr<- matrix(rnorm(15*100),ncol = 15, nrow = 100)


#Generate test set of covariates X and Z
X.te<- matrix(rnorm(10*100),ncol = 10, nrow = 100)
Z.te<- matrix(rnorm(15*100),ncol = 15, nrow = 100)

#Scale appropiately
meanX<- apply(X.tr,2,mean)
meanY<- apply(Z.tr,2,mean)

X.tr<- scale(X.tr, scale = FALSE)
Z.tr<- scale(Z.tr, scale = FALSE)
X.te<- scale(X.te,center = meanX,scale = FALSE)
Z.te<- scale(Z.te,center = meanY,scale = FALSE)

#Generate full matrix of Covariates
w.tr<- c()
w.te<- c()
X1<- cbind(1,X.tr)
Z1<- cbind(1,Z.tr)
X2<- cbind(1,X.te)
Z2<- cbind(1,Z.te)
```

```
for(i in 1:16){
  for(j in 1:11){
    w.tr<- cbind(w.tr,X1[,j]*Z1[,i])
    w.te<- cbind(w.te, X2[,j]*Z2[,i])
  }
}

#Generate response variables with signal from
#First 5 X features and 5 Z features.

#We construct the coefficient matrix B.
#B[1,1] contains the intercept
#B[-1,1] contains the main effects for X.
# For instance, B[2,1] is the main effect for the first feature in X.
#B[1,-1] contains the main effects for Z.
# For instance, B[1,10] is the coefficient for the 10th feature in Z.
#B[i+1,j+1] is the coefficient of X_i Z_j
B<- matrix(0,ncol = 16,nrow = 11)
rownames(B)<- c("inter" , paste("X",1:(nrow(B)-1),sep = ""))
colnames(B)<- c("inter" , paste("Z",1:(ncol(B)-1),sep = ""))

# First, we simulate data as follows:
# The first five features in X, and the first five features in Z, are non-zero.
# And given the non-zero main effects, all possible interactions are involved.
# We call this "high strong heredity"
B_high_SH<- B
B_high_SH[1:6,1:6]<- 1
#View true coefficient matrix
pheatmap(as.matrix(B_high_SH), scale="none",
         cluster_rows=FALSE, cluster_cols=FALSE)

Y_high_SH <- as.vector(w.tr%*%as.vector(B_high_SH))+rnorm(100,sd = 2)
Y_high_SH.te <- as.vector(w.te%*%as.vector(B_high_SH))+rnorm(100,sd = 2)

# Now a new setting:
# Again, the first five features in X, and the first five features in Z, are involved.
# But this time, only a subset of the possible interactions are involved.
# Strong heredity is still maintained.
# We call this "low strong heredity"
B_low_SH<- B_high_SH
B_low_SH[2:6,2:6]<-0
B_low_SH[3:4,3:5]<- 1
#View true coefficient matrix
pheatmap(as.matrix(B_low_SH), scale="none",
         cluster_rows=FALSE, cluster_cols=FALSE)
Y_low_SH <- as.vector(w.tr%*%as.vector(B_low_SH))+rnorm(100,sd = 1.5)
Y_low_SH.te <- as.vector(w.te%*%as.vector(B_low_SH))+rnorm(100,sd = 1.5)


############################## FIT SOME MODELS #######################################

#Define alphas and lambdas
```

```
#Define 3 different alpha values
#Low alpha values penalize groups more
#High alpha values penalize individual Interactions more
alphas<- c(0.01,0.5,0.99)
lambdas<- seq(0.1,1,length = 50)

#high Strong heredity with l2 norm
fit_high_SH<- FAMILY(X.tr, Z.tr, Y_high_SH, lambdas ,
                     alphas, quad = TRUE,iter=500, verbose = TRUE )
yhat_hSH<- predict(fit_high_SH, X.te, Z.te)
mse_hSH <-apply(yhat_hSH,c(2,3), "-" ,Y_high_SH.te)
mse_hSH<- apply(mse_hSH^2,c(2,3),sum)

#Find optimal model and plot matrix
im<- which(mse_hSH==min(mse_hSH),TRUE)
plot(fit_high_SH$Estimate[[im[2] ]][[im[1]]])


#Plot some matrices for different alpha values
#Low alpha, higher penalty on groups
plot(fit_high_SH$Estimate[[ 1 ]][[ 25 ]])
#Medium alpha, equal penalty on groups and individual interactions
plot(fit_high_SH$Estimate[[ 2 ]][[ 25  ]])
#High alpha, more penalty on individual interactions
plot(fit_high_SH$Estimate[[ 3 ]][[ 40 ]])


#View Coefficients
coef(fit_high_SH)[[im[2]]][[im[1]]]

############################## Uncomment code for EXAMPLE ###########################
# #high Strong heredity with l_infinity norm norm
# fit_high_SH<- FAMILY(X.tr, Z.tr, Y_high_SH, lambdas ,
#                      alphas, quad = TRUE,iter=500, verbose = TRUE,
#                      norm = "l_inf")
# yhat_hSH<- predict(fit_high_SH, X.te, Z.te)
# mse_hSH <-apply(yhat_hSH,c(2,3), "-" ,Y_high_SH.te)
# mse_hSH<- apply(mse_hSH^2,c(2,3),sum)
#
# #Find optimal model and plot matrix
# im<- which(mse_hSH==min(mse_hSH),TRUE)
# plot(fit_high_SH$Estimate[[im[2] ]][[im[1]]])
#
#
# #Plot some matrices for different alpha values
# #Low alpha, higher penalty on groups
# plot(fit_high_SH$Estimate[[ 1 ]][[ 30 ]])
# #Medium alpha, equal penalty on groups and individual interactions
# plot(fit_high_SH$Estimate[[ 2 ]][[ 10 ]])
# #High alpha, more penalty on individual interactions
# plot(fit_high_SH$Estimate[[ 3 ]][[ 20 ]])
#
#
```

```
# #View Coefficients
# coef(fit_high_SH)[[im[2]]][[im[1]]]


############################## Uncomment code for EXAMPLE ###########################
# #Redefine lambdas
# lambdas<- seq(0.1,0.5,length = 50)
#
# #low Strong heredity with l_2 norm
# fit_low_SH<- FAMILY(X.tr, Z.tr, Y_low_SH, lambdas ,
#                       alphas, quad = TRUE,iter=500, verbose = TRUE )
# yhat_lSH<- predict(fit_low_SH, X.te, Z.te)
# mse_lSH <-apply(yhat_lSH,c(2,3), "-" ,Y_low_SH.te)
# mse_lSH<- apply(mse_lSH^2,c(2,3),sum)
#
# #Find optimal model and plot matrix
# im<- which(mse_lSH==min(mse_lSH),TRUE)
# plot(fit_low_SH$Estimate[[im[2] ]][[im[1]]])
#
#
# #Plot some matrices for different alpha values
# #Low alpha, higher penalty on groups
# plot(fit_low_SH$Estimate[[ 1 ]][[ 25 ]])
# #Medium alpha, equal penalty on groups and individual interactions
# plot(fit_low_SH$Estimate[[ 2 ]][[ 10 ]])
# #High alpha, more penalty on individual interactions
# plot(fit_low_SH$Estimate[[ 3 ]][[ 10 ]])
#
#
# #View Coefficients
# coef(fit_low_SH)[[im[2]]][[im[1]]]


#####################################################################################
#####################################################################################
############################### EXAMPLE - BINARY RESPONSE ###########################
#####################################################################################
#####################################################################################

############################## GENERATE DATA ########################################

#Generate data for logistic regression
Yp_high_SH<- as.vector((w.tr)%*%as.vector(B_high_SH))
Yp_high_SH.te<- as.vector((w.te)%*%as.vector(B_high_SH))

Yprobs_high_SH<- 1/(1+exp(-Yp_high_SH))
Yprobs_high_SH.te<- 1/(1+exp(-Yp_high_SH.te))

Yp_high_SH<- rbinom(100, size = 1, prob = Yprobs_high_SH)
Yp_high_SH.te<- rbinom(100, size = 1, prob = Yprobs_high_SH.te)

lambdas<- seq(0.01,0.15,length = 50)
```

```
############################## FIT SOME MODELS ########################################

#Fit glm via l_2 norm
fit_high_SH<- FAMILY(X.tr, Z.tr, Yp_high_SH, lambdas ,
                    alphas, quad = TRUE,iter=500, verbose = TRUE,
                    family = "binomial")
yhp_hSH<- predict(fit_high_SH, X.te, Z.te)
mse_high_SH <-apply(yhp_hSH,c(2,3), "-" ,Yp_high_SH.te)
mse_hSH<- apply(mse_high_SH^2,c(2,3),sum)
im<- which(mse_hSH==min(mse_hSH),TRUE)
plot(fit_high_SH$Estimate[[im[2] ]][[im[1]]])
roc( Yp_high_SH.te,yhp_hSH[,im[1],im[2]],plot = TRUE)

#View Coefficients
coef(fit_high_SH)[[im[2]]][[im[1]]]

############################## Uncomment code for EXAMPLE ############################
# #Fit glm via l_infinity norm
# fit_high_SH<- FAMILY(X.tr, Z.tr, Yp_high_SH, lambdas , norm = "l_inf",
#                     alphas, quad = TRUE,iter=500, verbose = TRUE,
#                     family = "binomial")
# yhp_hSH<- predict(fit_high_SH, X.te, Z.te)
# mse_high_SH <-apply(yhp_hSH,c(2,3), "-" ,Yp_high_SH.te)
# mse_hSH<- apply(mse_high_SH^2,c(2,3),sum)
# im<- which(mse_hSH==min(mse_hSH),TRUE)
# plot(fit_high_SH$Estimate[[im[2] ]][[im[1]]])
# roc( Yp_high_SH.te,yhp_hSH[,im[1],im[2]],plot = TRUE)
#
# #View Coefficients
# coef(fit_high_SH)[[im[2]]][[im[1]]]

######################################################################################
######################################################################################
############################## EXAMPLE WHERE X=Z ####################################
######################## Uncomment Code for EXAMPLE ##################################
######################################################################################

############################## GENERATE DATA ########################################
# #Redefine Lambdas
# lambdas<- seq(0.01,0.3,length = 50)
#
#
# #We consider the case X=Z now
# w.tr<- c()
# w.te<- c()
# X1<- cbind(1,X.tr)
# X2<- cbind(1,X.te)
#
# for(i in 1:11){
#   for(j in 1:11){
#     w.tr<- cbind(w.tr,X1[,j]*X1[,i])
#     w.te<- cbind(w.te, X2[,j]*X2[,i])
#   }
# }
#
# B<- matrix(0,ncol = 11,nrow = 11)
# rownames(B)<- c("inter" , paste("X",1:(nrow(B)-1),sep = ""))
# colnames(B)<- c("inter" , paste("X",1:(ncol(B)-1),sep = ""))
#
#
# B_high_SH<- B
# B_high_SH[1:6,1:6]<- 1
# #We exclude quadratic terms in this example
# diag(B_high_SH)[-1]<-0
# #View true coefficient matrix
# pheatmap(as.matrix(B_high_SH), scale="none",
#          cluster_rows=FALSE, cluster_cols=FALSE)
#
# #With high Strong heredity: all possible interactions
# Y_high_SH <- as.vector(w.tr%*%as.vector(B_high_SH))+rnorm(100)
# Y_high_SH.te <- as.vector(w.te%*%as.vector(B_high_SH))+rnorm(100)
#
# ############################## FIT SOME MODELS ########################################
#
# #high Strong heredity with l_2 norm
# fit_high_SH<- FAMILY(X.tr, X.tr, Y_high_SH, lambdas ,
#                      alphas, quad = FALSE,iter=500, verbose = TRUE )
# yhat_hSH<- predict(fit_high_SH, X.te, X.te)
# mse_hSH <-apply(yhat_hSH,c(2,3), "-" ,Y_high_SH.te)
# mse_hSH<- apply(mse_hSH^2,c(2,3),sum)
#
# #Find optimal model and plot matrix
# im<- which(mse_hSH==min(mse_hSH),TRUE)
# plot(fit_high_SH$Estimate[[im[2] ]][[im[1]]])
#
#
# #Plot some matrices for different alpha values
# #Low alpha, higher penalty on groups
# plot(fit_high_SH$Estimate[[ 1 ]][[ 50 ]])
# #Medium alpha, equal penalty on groups and individual interactions
# plot(fit_high_SH$Estimate[[ 2 ]][[ 50 ]])
# #High alpha, more penalty on individual interactions
# plot(fit_high_SH$Estimate[[ 3 ]][[ 50 ]])
#
#
# #View Coefficients
# coef(fit_high_SH,XequalZ = TRUE)[[im[2]]][[im[1]]]
```

FAMILY *Framework for Modeling Interactions with Strong Heredity*

Description

This function runs the main algorithm presented in Haris, Witten and Simon (2014) for fitting an interaction model with strong heredity.

Usage

```
FAMILY(X, Z, Y, lambdas , alphas, family = c("gaussian","binomial"),
    rho = 1, B = NULL, norm = "l2", quad = TRUE,iter=500,
    e.abs = 1e-3, e.rel = 1e-3,
    maxiter.B = 50, tol.B = 1e-04, verbose = FALSE)
```

Arguments

`X`	A $n \times p_1$-matrix of covariates `X`.
`Z`	A $n \times p_2$-matrix of covariates `Z`. The number of rows of this matrix must coincide with that of `X`. For most cases we have `X=Z`.
`Y`	The response vector of length n. This has to be a numeric vector. For the case of logistic regression the response variable must be a binary vector.
`lambdas`	The vector of different penalty parameters λ for which we wish to evaluate the function. For details see Haris, Witten and Simon (2014).
`alphas`	The second tuning parameter to control the magnitude of penalties on groups Of variables versus individual interaction terms. The values of this vector must be in the interval $[0, 1]$. The output will fit the model for a grid of α and λ values.
`family`	A character string specifying the type of model to fit. "gaussian" for modeling continuous variables via linear regression (default), "binomial" for logistic regression.
`rho`	The starting value of $\rho > 0$, the augmented Lagrangian parameter.
`B`	Initial $(p_1 + 1) \times (p_2 + 1)$ matrix of coefficients, `B`. `B[1,1]` is the intercept, `B[1,-1]` and `B[-1,1]` are the main effects of `Z` and `X`, respectively, and `B[j+1,k+1]` is the coefficient of the interaction term $X_j Z_k$.
`norm`	The penalty to use for the rows and columns of matrix `B`. The two possible parameters are `"l2"` and `"l_inf"` for the gorup lasso and the infinity norm.
`quad`	A logical variable indicating if we wish to include quadratic terms when `X=Z`.
`iter`	The maximum number of iterations for the ADMM algorithm.
`e.abs`	An absolute tolerance for convergence.
`e.rel`	A relative tolerance for convergence. These are used to find a stopping criterion for the ADMM as done in Section 3.3.1 of Boyd, Stephen, et al. 2011
`maxiter.B`	The maximum number of iterations for updating `B` via the iterative algorithm for logistic regression.
`tol.B`	The absolute tolerance for the convergence of `B` for each iteration of the ADMM algorithm in the case of logistic regression.
`verbose`	Logical variable which indicates if extra statements should be printed showing progress of the algorithm.

Details

This function fits a regression model based with pair-wise interaction terms by solving the optimization problem (33)(linear regression) or (35)(logistic regression) in Haris, Witten and Simon (2014). The optimization problem is solved via an ADMM algorithm.

Value

The function returns a list where the first component, `Estimate`, is a list of dimensions length(alphas)*length(lambdas) where $Estimate[[$alpha[a]]][[$lambda[l]]] is an object with components

`finB`	The estimated coefficient matrix $\hat{B}$ obtained by the ADMM algorithm for minimizing the above objective function.
`B, D, E, F`	The matrices used in intermediate steps of the ADMM algorithm. We note that numerically all these matrices converge to `finB`. These matrices are primarily used internally within the function. For details regarding these matrices/notation, we refer the reader to Haris, Witten and Simon (2014).
`glist`	A list of final estimates for the dual variable of the ADMM algorithm
`rho`	The last value of the augmented Lagrangian parameter ρ used for the ADMM.
`conv`	A logical variable stating if the algorithm converged within the maximum number of iterations
`iter`	The number of iterations for which our algorithm ran. If the algorithm did not converge this will just be equal to the input parameter `iter`.

The function also returns the training data used to fit the model and the path of penalty parameters for which we estimated the model.

References

Haris, Witten and Simon (2014). Convex Modeling of Interactions with Strong Heredity. Available on ArXiv at http://arxiv.org/abs/1410.3517.

Boyd, Stephen, et al. "Distributed optimization and statistical learning via the alternating direction method of multipliers." Foundations and Trends? in Machine Learning 3.1 (2011): 1-122.

See Also

`coef.FAMILY`, `predict.FAMILY`

Examples

```
library(FAMILY)
library(pROC)
library(pheatmap)

######################################################################################
######################################################################################
############################# EXAMPLE - CONTINUOUS RESPONSE ##########################
######################################################################################
######################################################################################
```

```
############################## GENERATE DATA #######################################

#Generate training set of covariates X and Z
set.seed(1)
X.tr<- matrix(rnorm(10*100),ncol = 10, nrow = 100)
Z.tr<- matrix(rnorm(15*100),ncol = 15, nrow = 100)


#Generate test set of covariates X and Z
X.te<- matrix(rnorm(10*100),ncol = 10, nrow = 100)
Z.te<- matrix(rnorm(15*100),ncol = 15, nrow = 100)

#Scale appropiately
meanX<- apply(X.tr,2,mean)
meanY<- apply(Z.tr,2,mean)

X.tr<- scale(X.tr, scale = FALSE)
Z.tr<- scale(Z.tr, scale = FALSE)
X.te<- scale(X.te,center = meanX,scale = FALSE)
Z.te<- scale(Z.te,center = meanY,scale = FALSE)

#Generate full matrix of Covariates
w.tr<- c()
w.te<- c()
X1<- cbind(1,X.tr)
Z1<- cbind(1,Z.tr)
X2<- cbind(1,X.te)
Z2<- cbind(1,Z.te)

for(i in 1:16){
  for(j in 1:11){
    w.tr<- cbind(w.tr,X1[,j]*Z1[,i])
    w.te<- cbind(w.te, X2[,j]*Z2[,i])
  }
}

#Generate response variables with signal from
#First 5 X features and 5 Z features.

#We construct the coefficient matrix B.
#B[1,1] contains the intercept
#B[-1,1] contains the main effects for X.
# For instance, B[2,1] is the main effect for the first feature in X.
#B[1,-1] contains the main effects for Z.
# For instance, B[1,10] is the coefficient for the 10th feature in Z.
#B[i+1,j+1] is the coefficient of X_i Z_j
B<- matrix(0,ncol = 16,nrow = 11)
rownames(B)<- c("inter" , paste("X",1:(nrow(B)-1),sep = ""))
colnames(B)<- c("inter" , paste("Z",1:(ncol(B)-1),sep = ""))

# First, we simulate data as follows:
# The first five features in X, and the first five features in Z, are non-zero.
# And given the non-zero main effects, all possible interactions are involved.
# We call this "high strong heredity"
B_high_SH<- B
B_high_SH[1:6,1:6]<- 1
#View true coefficient matrix
pheatmap(as.matrix(B_high_SH), scale="none",
         cluster_rows=FALSE, cluster_cols=FALSE)

Y_high_SH <- as.vector(w.tr%*%as.vector(B_high_SH))+rnorm(100,sd = 2)
Y_high_SH.te <- as.vector(w.te%*%as.vector(B_high_SH))+rnorm(100,sd = 2)

# Now a new setting:
# Again, the first five features in X, and the first five features in Z, are involved.
# But this time, only a subset of the possible interactions are involved.
# Strong heredity is still maintained.
# We call this "low strong heredity"
B_low_SH<- B_high_SH
B_low_SH[2:6,2:6]<-0
B_low_SH[3:4,3:5]<- 1
#View true coefficient matrix
pheatmap(as.matrix(B_low_SH), scale="none",
         cluster_rows=FALSE, cluster_cols=FALSE)
Y_low_SH <- as.vector(w.tr%*%as.vector(B_low_SH))+rnorm(100,sd = 1.5)
Y_low_SH.te <- as.vector(w.te%*%as.vector(B_low_SH))+rnorm(100,sd = 1.5)


############################## FIT SOME MODELS ########################################

#Define alphas and lambdas
#Define 3 different alpha values
#Low alpha values penalize groups more
#High alpha values penalize individual Interactions more
alphas<- c(0.01,0.5,0.99)
lambdas<- seq(0.1,1,length = 50)

#high Strong heredity with l2 norm
fit_high_SH<- FAMILY(X.tr, Z.tr, Y_high_SH, lambdas ,
                     alphas, quad = TRUE,iter=500, verbose = TRUE )
yhat_hSH<- predict(fit_high_SH, X.te, Z.te)
mse_hSH <-apply(yhat_hSH,c(2,3), "-" ,Y_high_SH.te)
mse_hSH<- apply(mse_hSH^2,c(2,3),sum)

#Find optimal model and plot matrix
im<- which(mse_hSH==min(mse_hSH),TRUE)
plot(fit_high_SH$Estimate[[im[2] ]][[im[1]]])


#Plot some matrices for different alpha values
#Low alpha, higher penalty on groups
plot(fit_high_SH$Estimate[[ 1 ]][[ 25 ]])
#Medium alpha, equal penalty on groups and individual interactions
plot(fit_high_SH$Estimate[[ 2 ]][[ 25  ]])
#High alpha, more penalty on individual interactions
```

```
plot(fit_high_SH$Estimate[[ 3 ]][[ 40 ]])


#View Coefficients
coef(fit_high_SH)[[im[2]]][[im[1]]]

############################### Uncomment code for EXAMPLE ###########################
# #high Strong heredity with l_infinity norm norm
# fit_high_SH<- FAMILY(X.tr, Z.tr, Y_high_SH, lambdas ,
#                       alphas, quad = TRUE,iter=500, verbose = TRUE,
#                       norm = "l_inf")
# yhat_hSH<- predict(fit_high_SH, X.te, Z.te)
# mse_hSH <-apply(yhat_hSH,c(2,3), "-" ,Y_high_SH.te)
# mse_hSH<- apply(mse_hSH^2,c(2,3),sum)
#
# #Find optimal model and plot matrix
# im<- which(mse_hSH==min(mse_hSH),TRUE)
# plot(fit_high_SH$Estimate[[im[2] ]][[im[1]]])
#
#
# #Plot some matrices for different alpha values
# #Low alpha, higher penalty on groups
# plot(fit_high_SH$Estimate[[ 1 ]][[ 30 ]])
# #Medium alpha, equal penalty on groups and individual interactions
# plot(fit_high_SH$Estimate[[ 2 ]][[ 10 ]])
# #High alpha, more penalty on individual interactions
# plot(fit_high_SH$Estimate[[ 3 ]][[ 20 ]])
#
#
# #View Coefficients
# coef(fit_high_SH)[[im[2]]][[im[1]]]


############################### Uncomment code for EXAMPLE ###########################
# #Redefine lambdas
# lambdas<- seq(0.1,0.5,length = 50)
#
# #low Strong heredity with l_2 norm
# fit_low_SH<- FAMILY(X.tr, Z.tr, Y_low_SH, lambdas ,
#                       alphas, quad = TRUE,iter=500, verbose = TRUE )
# yhat_lSH<- predict(fit_low_SH, X.te, Z.te)
# mse_lSH <-apply(yhat_lSH,c(2,3), "-" ,Y_low_SH.te)
# mse_lSH<- apply(mse_lSH^2,c(2,3),sum)
#
# #Find optimal model and plot matrix
# im<- which(mse_lSH==min(mse_lSH),TRUE)
# plot(fit_low_SH$Estimate[[im[2] ]][[im[1]]])
#
#
# #Plot some matrices for different alpha values
# #Low alpha, higher penalty on groups
# plot(fit_low_SH$Estimate[[ 1 ]][[ 25 ]])
# #Medium alpha, equal penalty on groups and individual interactions
# plot(fit_low_SH$Estimate[[ 2 ]][[ 10 ]])
# #High alpha, more penalty on individual interactions
# plot(fit_low_SH$Estimate[[ 3 ]][[ 10 ]])
#
#
# #View Coefficients
# coef(fit_low_SH)[[im[2]]][[im[1]]]


######################################################################################
######################################################################################
############################### EXAMPLE - BINARY RESPONSE ###########################
######################################################################################
######################################################################################

############################## GENERATE DATA ########################################

#Generate data for logistic regression
Yp_high_SH<- as.vector((w.tr)%*%as.vector(B_high_SH))
Yp_high_SH.te<- as.vector((w.te)%*%as.vector(B_high_SH))

Yprobs_high_SH<- 1/(1+exp(-Yp_high_SH))
Yprobs_high_SH.te<- 1/(1+exp(-Yp_high_SH.te))

Yp_high_SH<- rbinom(100, size = 1, prob = Yprobs_high_SH)
Yp_high_SH.te<- rbinom(100, size = 1, prob = Yprobs_high_SH.te)

lambdas<- seq(0.01,0.15,length = 50)

############################## FIT SOME MODELS ########################################

#Fit glm via l_2 norm
fit_high_SH<- FAMILY(X.tr, Z.tr, Yp_high_SH, lambdas ,
                    alphas, quad = TRUE,iter=500, verbose = TRUE,
                    family = "binomial")
yhp_hSH<- predict(fit_high_SH, X.te, Z.te)
mse_high_SH <-apply(yhp_hSH,c(2,3), "-" ,Yp_high_SH.te)
mse_hSH<- apply(mse_high_SH^2,c(2,3),sum)
im<- which(mse_hSH==min(mse_hSH),TRUE)
plot(fit_high_SH$Estimate[[im[2] ]][[im[1]]])
roc( Yp_high_SH.te,yhp_hSH[,im[1],im[2]],plot = TRUE)

#View Coefficients
coef(fit_high_SH)[[im[2]]][[im[1]]]

############################## Uncomment code for EXAMPLE ###########################
# #Fit glm via l_infinity norm
# fit_high_SH<- FAMILY(X.tr, Z.tr, Yp_high_SH, lambdas , norm = "l_inf",
#                       alphas, quad = TRUE,iter=500, verbose = TRUE,
#                       family = "binomial")
# yhp_hSH<- predict(fit_high_SH, X.te, Z.te)
# mse_high_SH <-apply(yhp_hSH,c(2,3), "-" ,Yp_high_SH.te)
# mse_hSH<- apply(mse_high_SH^2,c(2,3),sum)
# im<- which(mse_hSH==min(mse_hSH),TRUE)
# plot(fit_high_SH$Estimate[[im[2] ]][[im[1]]])
# roc( Yp_high_SH.te,yhp_hSH[,im[1],im[2]],plot = TRUE)
#
# #View Coefficients
# coef(fit_high_SH)[[im[2]]][[im[1]]]

#####################################################################################
#####################################################################################
############################## EXAMPLE WHERE X=Z ####################################
######################## Uncomment Code for EXAMPLE #################################
#####################################################################################

############################## GENERATE DATA ########################################
# #Redefine Lambdas
# lambdas<- seq(0.01,0.3,length = 50)
#
#
# #We consider the case X=Z now
# w.tr<- c()
# w.te<- c()
# X1<- cbind(1,X.tr)
# X2<- cbind(1,X.te)
#
# for(i in 1:11){
#   for(j in 1:11){
#     w.tr<- cbind(w.tr,X1[,j]*X1[,i])
#     w.te<- cbind(w.te, X2[,j]*X2[,i])
#   }
# }
#
# B<- matrix(0,ncol = 11,nrow = 11)
# rownames(B)<- c("inter" , paste("X",1:(nrow(B)-1),sep = ""))
# colnames(B)<- c("inter" , paste("X",1:(ncol(B)-1),sep = ""))
#
#
# B_high_SH<- B
# B_high_SH[1:6,1:6]<- 1
# #We exclude quadratic terms in this example
# diag(B_high_SH)[-1]<-0
# #View true coefficient matrix
# pheatmap(as.matrix(B_high_SH), scale="none",
#          cluster_rows=FALSE, cluster_cols=FALSE)
#
# #With high Strong heredity: all possible interactions
# Y_high_SH <- as.vector(w.tr%*%as.vector(B_high_SH))+rnorm(100)
# Y_high_SH.te <- as.vector(w.te%*%as.vector(B_high_SH))+rnorm(100)
#
# ############################## FIT SOME MODELS #######################################
#
# #high Strong heredity with l_2 norm
# fit_high_SH<- FAMILY(X.tr, X.tr, Y_high_SH, lambdas ,
#                       alphas, quad = FALSE,iter=500, verbose = TRUE )
# yhat_hSH<- predict(fit_high_SH, X.te, X.te)
# mse_hSH <-apply(yhat_hSH,c(2,3), "-" ,Y_high_SH.te)
# mse_hSH<- apply(mse_hSH^2,c(2,3),sum)
#
# #Find optimal model and plot matrix
# im<- which(mse_hSH==min(mse_hSH),TRUE)
# plot(fit_high_SH$Estimate[[im[2] ]][[im[1]]])
#
#
# #Plot some matrices for different alpha values
# #Low alpha, higher penalty on groups
# plot(fit_high_SH$Estimate[[ 1 ]][[ 50 ]])
# #Medium alpha, equal penalty on groups and individual interactions
# plot(fit_high_SH$Estimate[[ 2 ]][[ 50 ]])
# #High alpha, more penalty on individual interactions
# plot(fit_high_SH$Estimate[[ 3 ]][[ 50 ]])
#
#
# #View Coefficients
# coef(fit_high_SH,XequalZ = TRUE)[[im[2]]][[im[1]]]
```

predict.FAMILY *predict.FAMILY*

Description

Similar to R's generic `predict` function which predicts the model for new data for different values of α and λ.

Usage

```
## S3 method for class 'FAMILY'
predict(object, new.X, new.Z, Bias.corr = FALSE, XequalZ = FALSE, ...)
```

Arguments

`object`	The fitted object as the output from the main function `FAMILY`.
`new.X`	Matrix of covariates `X`. Must have the same number of columns used for fitting the model.
`new.Z`	Matrix of covariates `Z`. Must have the same number of columns used for fitting the model.
`Bias.corr`	A logical variable indicating if we wish to re-fit the selected variables using `glm` or `lm`.
`XequalZ`	A logical variable indicating if $X = Z$ or if we have two different sets of covariates.
`...`	Extra arguments for the generic S3 `predict` function

Value

The function returns an array of dimensions `[n, length(alphas), length(lambdas)]` where `n = nrow(new.X)`. This array contains one the following:

`yhat`	The fitted values using the data given
`phat`	The fitted estimated probabilities for logistic regression

Author(s)

Asad Haris

References

Haris, Witten and Simon (2014). Convex Modeling of Interactions with Strong Heredity. Available on ArXiv at http://arxiv.org/abs/1410.3517

See Also

`FAMILY`

Examples

```
library(FAMILY)
library(pROC)
library(pheatmap)

#####################################################################################
#####################################################################################
############################# EXAMPLE - CONTINUOUS RESPONSE ########################
#####################################################################################
#####################################################################################

############################## GENERATE DATA ########################################

#Generate training set of covariates X and Z
set.seed(1)
X.tr<- matrix(rnorm(10*100),ncol = 10, nrow = 100)
Z.tr<- matrix(rnorm(15*100),ncol = 15, nrow = 100)


#Generate test set of covariates X and Z
X.te<- matrix(rnorm(10*100),ncol = 10, nrow = 100)
Z.te<- matrix(rnorm(15*100),ncol = 15, nrow = 100)

#Scale appropiately
meanX<- apply(X.tr,2,mean)
meanY<- apply(Z.tr,2,mean)

X.tr<- scale(X.tr, scale = FALSE)
Z.tr<- scale(Z.tr, scale = FALSE)
X.te<- scale(X.te,center = meanX,scale = FALSE)
```

```
Z.te<- scale(Z.te,center = meanY,scale = FALSE)

#Generate full matrix of Covariates
w.tr<- c()
w.te<- c()
X1<- cbind(1,X.tr)
Z1<- cbind(1,Z.tr)
X2<- cbind(1,X.te)
Z2<- cbind(1,Z.te)

for(i in 1:16){
  for(j in 1:11){
    w.tr<- cbind(w.tr,X1[,j]*Z1[,i])
    w.te<- cbind(w.te, X2[,j]*Z2[,i])
  }
}

#Generate response variables with signal from
#First 5 X features and 5 Z features.

#We construct the coefficient matrix B.
#B[1,1] contains the intercept
#B[-1,1] contains the main effects for X.
# For instance, B[2,1] is the main effect for the first feature in X.
#B[1,-1] contains the main effects for Z.
# For instance, B[1,10] is the coefficient for the 10th feature in Z.
#B[i+1,j+1] is the coefficient of X_i Z_j
B<- matrix(0,ncol = 16,nrow = 11)
rownames(B)<- c("inter" , paste("X",1:(nrow(B)-1),sep = ""))
colnames(B)<- c("inter" , paste("Z",1:(ncol(B)-1),sep = ""))

# First, we simulate data as follows:
# The first five features in X, and the first five features in Z, are non-zero.
# And given the non-zero main effects, all possible interactions are involved.
# We call this "high strong heredity"
B_high_SH<- B
B_high_SH[1:6,1:6]<- 1
#View true coefficient matrix
pheatmap(as.matrix(B_high_SH), scale="none",
         cluster_rows=FALSE, cluster_cols=FALSE)

Y_high_SH <- as.vector(w.tr%*%as.vector(B_high_SH))+rnorm(100,sd = 2)
Y_high_SH.te <- as.vector(w.te%*%as.vector(B_high_SH))+rnorm(100,sd = 2)

# Now a new setting:
# Again, the first five features in X, and the first five features in Z, are involved.
# But this time, only a subset of the possible interactions are involved.
# Strong heredity is still maintained.
# We call this "low strong heredity"
B_low_SH<- B_high_SH
B_low_SH[2:6,2:6]<-0
B_low_SH[3:4,3:5]<- 1
#View true coefficient matrix
```

```
pheatmap(as.matrix(B_low_SH), scale="none",
         cluster_rows=FALSE, cluster_cols=FALSE)
Y_low_SH <- as.vector(w.tr%*%as.vector(B_low_SH))+rnorm(100,sd = 1.5)
Y_low_SH.te <- as.vector(w.te%*%as.vector(B_low_SH))+rnorm(100,sd = 1.5)


############################## FIT SOME MODELS ########################################

#Define alphas and lambdas
#Define 3 different alpha values
#Low alpha values penalize groups more
#High alpha values penalize individual Interactions more
alphas<- c(0.01,0.5,0.99)
lambdas<- seq(0.1,1,length = 50)

#high Strong heredity with l2 norm
fit_high_SH<- FAMILY(X.tr, Z.tr, Y_high_SH, lambdas ,
                     alphas, quad = TRUE,iter=500, verbose = TRUE )
yhat_hSH<- predict(fit_high_SH, X.te, Z.te)
mse_hSH <-apply(yhat_hSH,c(2,3), "-" ,Y_high_SH.te)
mse_hSH<- apply(mse_hSH^2,c(2,3),sum)

#Find optimal model and plot matrix
im<- which(mse_hSH==min(mse_hSH),TRUE)
plot(fit_high_SH$Estimate[[im[2] ]][[im[1]]])


#Plot some matrices for different alpha values
#Low alpha, higher penalty on groups
plot(fit_high_SH$Estimate[[ 1 ]][[ 25 ]])
#Medium alpha, equal penalty on groups and individual interactions
plot(fit_high_SH$Estimate[[ 2 ]][[ 25  ]])
#High alpha, more penalty on individual interactions
plot(fit_high_SH$Estimate[[ 3 ]][[ 40 ]])


#View Coefficients
coef(fit_high_SH)[[im[2]]][[im[1]]]

############################## Uncomment code for EXAMPLE ###########################
# #high Strong heredity with l_infinity norm norm
# fit_high_SH<- FAMILY(X.tr, Z.tr, Y_high_SH, lambdas ,
#                      alphas, quad = TRUE,iter=500, verbose = TRUE,
#                      norm = "l_inf")
# yhat_hSH<- predict(fit_high_SH, X.te, Z.te)
# mse_hSH <-apply(yhat_hSH,c(2,3), "-" ,Y_high_SH.te)
# mse_hSH<- apply(mse_hSH^2,c(2,3),sum)
#
# #Find optimal model and plot matrix
# im<- which(mse_hSH==min(mse_hSH),TRUE)
# plot(fit_high_SH$Estimate[[im[2] ]][[im[1]]])
#
#
```

```
# #Plot some matrices for different alpha values
# #Low alpha, higher penalty on groups
# plot(fit_high_SH$Estimate[[ 1 ]][[ 30 ]])
# #Medium alpha, equal penalty on groups and individual interactions
# plot(fit_high_SH$Estimate[[ 2 ]][[ 10 ]])
# #High alpha, more penalty on individual interactions
# plot(fit_high_SH$Estimate[[ 3 ]][[ 20 ]])
#
#
# #View Coefficients
# coef(fit_high_SH)[[im[2]]][[im[1]]]


############################## Uncomment code for EXAMPLE ###########################
# #Redefine lambdas
# lambdas<- seq(0.1,0.5,length = 50)
#
# #low Strong heredity with l_2 norm
# fit_low_SH<- FAMILY(X.tr, Z.tr, Y_low_SH, lambdas ,
#                       alphas, quad = TRUE,iter=500, verbose = TRUE )
# yhat_lSH<- predict(fit_low_SH, X.te, Z.te)
# mse_lSH <-apply(yhat_lSH,c(2,3), "-" ,Y_low_SH.te)
# mse_lSH<- apply(mse_lSH^2,c(2,3),sum)
#
# #Find optimal model and plot matrix
# im<- which(mse_lSH==min(mse_lSH),TRUE)
# plot(fit_low_SH$Estimate[[im[2] ]][[im[1]]])
#
#
# #Plot some matrices for different alpha values
# #Low alpha, higher penalty on groups
# plot(fit_low_SH$Estimate[[ 1 ]][[ 25 ]])
# #Medium alpha, equal penalty on groups and individual interactions
# plot(fit_low_SH$Estimate[[ 2 ]][[ 10 ]])
# #High alpha, more penalty on individual interactions
# plot(fit_low_SH$Estimate[[ 3 ]][[ 10 ]])
#
#
# #View Coefficients
# coef(fit_low_SH)[[im[2]]][[im[1]]]


#####################################################################################
#####################################################################################
############################### EXAMPLE - BINARY RESPONSE ###########################
#####################################################################################
#####################################################################################

############################## GENERATE DATA ########################################

#Generate data for logistic regression
Yp_high_SH<- as.vector((w.tr)%*%as.vector(B_high_SH))
Yp_high_SH.te<- as.vector((w.te)%*%as.vector(B_high_SH))
```

```
Yprobs_high_SH<- 1/(1+exp(-Yp_high_SH))
Yprobs_high_SH.te<- 1/(1+exp(-Yp_high_SH.te))

Yp_high_SH<- rbinom(100, size = 1, prob = Yprobs_high_SH)
Yp_high_SH.te<- rbinom(100, size = 1, prob = Yprobs_high_SH.te)

lambdas<- seq(0.01,0.15,length = 50)

############################## FIT SOME MODELS ########################################

#Fit glm via l_2 norm
fit_high_SH<- FAMILY(X.tr, Z.tr, Yp_high_SH, lambdas ,
                    alphas, quad = TRUE,iter=500, verbose = TRUE,
                    family = "binomial")
yhp_hSH<- predict(fit_high_SH, X.te, Z.te)
mse_high_SH <-apply(yhp_hSH,c(2,3), "-" ,Yp_high_SH.te)
mse_hSH<- apply(mse_high_SH^2,c(2,3),sum)
im<- which(mse_hSH==min(mse_hSH),TRUE)
plot(fit_high_SH$Estimate[[im[2] ]][[im[1]]])
roc( Yp_high_SH.te,yhp_hSH[,im[1],im[2]],plot = TRUE)

#View Coefficients
coef(fit_high_SH)[[im[2]]][[im[1]]]

############################## Uncomment code for EXAMPLE ###########################
# #Fit glm via l_infinity norm
# fit_high_SH<- FAMILY(X.tr, Z.tr, Yp_high_SH, lambdas , norm = "l_inf",
#                      alphas, quad = TRUE,iter=500, verbose = TRUE,
#                      family = "binomial")
# yhp_hSH<- predict(fit_high_SH, X.te, Z.te)
# mse_high_SH <-apply(yhp_hSH,c(2,3), "-" ,Yp_high_SH.te)
# mse_hSH<- apply(mse_high_SH^2,c(2,3),sum)
# im<- which(mse_hSH==min(mse_hSH),TRUE)
# plot(fit_high_SH$Estimate[[im[2] ]][[im[1]]])
# roc( Yp_high_SH.te,yhp_hSH[,im[1],im[2]],plot = TRUE)
#
# #View Coefficients
# coef(fit_high_SH)[[im[2]]][[im[1]]]

########################################################################################
########################################################################################
############################## EXAMPLE WHERE X=Z ####################################
######################## Uncomment Code for EXAMPLE #################################
########################################################################################

############################## GENERATE DATA ########################################
# #Redefine Lambdas
# lambdas<- seq(0.01,0.3,length = 50)
#
#
# #We consider the case X=Z now
# w.tr<- c()
# w.te<- c()
# X1<- cbind(1,X.tr)
# X2<- cbind(1,X.te)
#
# for(i in 1:11){
#   for(j in 1:11){
#     w.tr<- cbind(w.tr,X1[,j]*X1[,i])
#     w.te<- cbind(w.te, X2[,j]*X2[,i])
#   }
# }
#
# B<- matrix(0,ncol = 11,nrow = 11)
# rownames(B)<- c("inter" , paste("X",1:(nrow(B)-1),sep = ""))
# colnames(B)<- c("inter" , paste("X",1:(ncol(B)-1),sep = ""))
#
#
# B_high_SH<- B
# B_high_SH[1:6,1:6]<- 1
# #We exclude quadratic terms in this example
# diag(B_high_SH)[-1]<-0
# #View true coefficient matrix
# pheatmap(as.matrix(B_high_SH), scale="none",
#          cluster_rows=FALSE, cluster_cols=FALSE)
#
# #With high Strong heredity: all possible interactions
# Y_high_SH <- as.vector(w.tr%*%as.vector(B_high_SH))+rnorm(100)
# Y_high_SH.te <- as.vector(w.te%*%as.vector(B_high_SH))+rnorm(100)
#
# ############################## FIT SOME MODELS ########################################
#
# #high Strong heredity with l_2 norm
# fit_high_SH<- FAMILY(X.tr, X.tr, Y_high_SH, lambdas ,
#                      alphas, quad = FALSE,iter=500, verbose = TRUE )
# yhat_hSH<- predict(fit_high_SH, X.te, X.te)
# mse_hSH <-apply(yhat_hSH,c(2,3), "-" ,Y_high_SH.te)
# mse_hSH<- apply(mse_hSH^2,c(2,3),sum)
#
# #Find optimal model and plot matrix
# im<- which(mse_hSH==min(mse_hSH),TRUE)
# plot(fit_high_SH$Estimate[[im[2] ]][[im[1]]])
#
#
# #Plot some matrices for different alpha values
# #Low alpha, higher penalty on groups
# plot(fit_high_SH$Estimate[[ 1 ]][[ 50 ]])
# #Medium alpha, equal penalty on groups and individual interactions
# plot(fit_high_SH$Estimate[[ 2 ]][[ 50 ]])
# #High alpha, more penalty on individual interactions
# plot(fit_high_SH$Estimate[[ 3 ]][[ 50 ]])
#
#
# #View Coefficients
# coef(fit_high_SH,XequalZ = TRUE)[[im[2]]][[im[1]]]
```

Index