Package ‘sRDA’

December 14, 2017

Title Sparse Redundancy Analysis
Version 1.0.0
Date 2017-12-12
Author Attila Csala [aut, cre], Koos Zwinderman [ctb]
Maintainer Attila Csala <a@csala.me>
Description Sparse redundancy analysis
   for high dimensional (biomedical) data. Directional multivariate analysis
   to express the maximum variance in the predicted data set by a linear
   combination of variables of the predictive data set. Implemented in a
   partial least squares framework, for more de-
Depends R (>= 2.7), Matrix, doParallel, elasticnet, foreach, mvtnorm
License MIT + file LICENSE
Encoding UTF-8
LazyData true
RoxygenNote 6.0.1
NeedsCompilation no
Repository CRAN
Date/Publication 2017-12-14 13:36:54 UTC

R topics documented:

  generate_data .......................................................... 2
  sCCA ................................................................. 3
  sRDA ................................................................. 6

Index 10
generate_data

Generate data sets for sparse multivariate analysis

description

Generate two data sets with highly correlated and noise variables modeled in a multiple latent variable structure. The latent variables are orthogonal to each other thus capture a different portion of association between the involved data sets. The function generates data that can be used to verify sRDA's ability of finding the highly correlated variables across multiple latent variables.

usage

```r
generate_data(nr_LVs = 1, n = 50, nr_correlated_Xs = c(5),
nr_uncorrelated_Xs = 250, mean_reg_weights_assoc_X = c(0.7),
sd_reg_weights_assoc_X = c(0.05), Xnoise_min = -0.3, Xnoise_max = 0.3,
nr_correlated_Ys = c(5), nr_uncorrelated_Ys = 350,
mean_reg_weights_assoc_Y = c(0.7), sd_reg_weights_assoc_Y = c(0.05),
Ynoise_min = -0.3, Ynoise_max = 0.3)
```

arguments

- `nr_LVs`: The number of latent variables between the predictive and predicted data sets. The latent variables model the association between data sets.
- `n`: The number of observations (rows) in the data sets.
- `nr_correlated_Xs`: Number of variables of the predictive data set that are associated with the latent variables.
- `nr_uncorrelated_Xs`: Number of variables of the predictive data set that is not associated with the latent variables.
- `mean_reg_weights_assoc_X`: Mean of the regression weights of the predictive variables that are associated with the latent variables.
- `sd_reg_weights_assoc_X`: Standard deviation of the regression weights of the predictive variables that are associated with the latent variables.
- `Xnoise_min`: The lower bound of the uniform distribution that is used to sample the values for the regression weights of the predictive variables that are not associated with the latent variables.
- `Xnoise_max`: The upper bound of the uniform distribution that is used to sample the values for the regression weights of the predictive variables that are not associated with the latent variables.
- `nr_correlated_Ys`: Number of variables of the predictive data set that are associated with the latent variables.
nr_uncorrelated_Ys

Number of variables of the predicted data set that is not associated with the latent variables.

mean_reg_weights_assoc_Y

Mean of the regression weights of the predicted variables that are associated with the latent variables.

sd_reg_weights_assoc_Y

Standard deviation of the regression weights of the predicted variables that are associated with the latent variables.

ynoise_min

The lower bound of the uniform distribution that is used to sample the values for the regression weights of the predicted variables that are not associated with the latent variables.

ynoise_max

The upper bound of the uniform distribution that is used to sample the values for the regression weights of the predicted variables that are not associated with the latent variables.

Examples

# generate data with few highly correlated variables
dataXY <- generate_data(nr_LVs = 2,
                        n = 250,
                        nr_correlated_Xs = c(5,20),
                        nr_uncorrelated_Xs = 250,
                        mean_reg_weights_assoc_X = c(0.9,0.5),
                        sd_reg_weights_assoc_X = c(0.05, 0.05),
                        Xnoise_min = -0.3,
                        Xnoise_max = 0.3,
                        nr_correlated_Ys = c(10,15),
                        nr_uncorrelated_Ys = 350,
                        mean_reg_weights_assoc_Y = c(0.9,0.6),
                        sd_reg_weights_assoc_Y = c(0.05, 0.05),
                        Ynoise_min = -0.3,
                        Ynoise_max = 0.3)

# separate predictor and predicted sets
X <- dataXY$X
Y <- dataXY$Y

dim(X); dim(Y)
Description

Sparse Canonical Correlation analysis for high dimensional (biomedical) data. The function takes two datasets and returns a linear combination of maximally correlated canonical correlate pairs. Elastic net penalization (with its variants, UST, Ridge and Lasso penalization) is implemented for sparsity and smoothness with a built in cross validation procedure to obtain the optimal penalization parameters. It is possible to obtain multiple canonical variate pairs that are orthogonal to each other.

Usage

sCCA(predictor, predicted, penalization = "enet", ridge_penalty = 1, nonzero = 1, max_iterations = 100, tolerance = 1 * 10^-20, cross_validate = FALSE, parallel_cv = TRUE, nr_subsets = 10, multiple_LV = FALSE, nr_LVs = 1)

Arguments

predictor The n*p matrix of the predictor data set
predicted The n*q matrix of the predicted data set
penalization The penalization method applied during the analysis (none, enet or ust)
ridge_penalty The ridge penalty parameter of the predictor set’s latent variable used for enet or ust (an integer if cross_validate = FALSE, a list otherwise)
nonzero The number of non-zero weights of the predictor set’s latent variable (an integer if cross_validate = FALSE, a list otherwise)
max_iterations The maximum number of iterations of the algorithm
tolerance Convergence criteria
cross_validate K-fold cross validation to find best optimal penalty parameters (TRUE or FALSE)
parallel_cv Run the cross validation parallel (TRUE or FALSE)
nr_subsets Number of subsets for k-fold cross validation
multiple_LV Obtain multiple latent variable pairs (TRUE or FALSE)
nr_LVs Number of latent variables to be obtained

Value

An object of class "sRDA".

XI Predictor set’s latent variable scores
ETA Predictive set’s latent variable scores
ALPHA Weights of the predictor set’s latent variable
BETA Weights of the predicted set’s latent variable
nr_iterations Number of iterations ran before convergence (or max number of iterations)
SOLVE_XIXI Inverse of the predictor set’s latent variable variance matrix
iterations_crt The convergence criterion value (a small positive tolerance)
sum_absolute_betassum of the absolute values of beta weights
ridge_penalty
The ridge penalty parameter used for the model
nr_nonzeros
The number of nonzero alpha weights in the model
nr_latent_variables
The number of latent variable pairs in the model
CV_results
The detailed results of cross validations (if cross_validate is TRUE)

Author(s)
Attila Csala

Examples

```r
# generate data with few highly correlated variables
dataXY <- generate_data(nr_LVs = 2,
n = 250,
  nr_correlated_Xs = c(5,20),
  nr_uncorrelated_Xs = 250,
  mean_reg_weights_assoc_X =
    c(0.9,0.5),
  sd_reg_weights_assoc_X =
    c(0.05, 0.05),
  Xnoise_min = -0.3,
  Xnoise_max = 0.3,
  nr_correlated_Ys = c(10,15),
  nr_uncorrelated_Ys = 350,
  mean_reg_weights_assoc_Y =
    c(0.9,0.6),
  sd_reg_weights_assoc_Y =
    c(0.05, 0.05),
  Ynoise_min = -0.3,
  Ynoise_max = 0.3)

 # separate predictor and predicted sets
X <- dataXY$X
Y <- dataXY$Y

# run sRDA
CCA.res <- sCCA(predictor = X, predicted = Y, nonzero = 5,
  ridge_penalty = 1, penalization = "ust")

# check first 10 weights of X
CCA.res$ALPHA[1:10]

# Not run:
# run sRDA with cross-validation to determine best penalization parameters
CCA.res <- sCCA(predictor = X, predicted = Y, nonzero = c(5,10,15),
  ridge_penalty = c(0.1,1), penalization = "enet", cross_validate = TRUE,
  ...)```
sparse Redundancy Analysis (sRDA) to express the maximum variance in the predicted data set by a linear combination of variables (latent variable) of the predictive data set. Elastic net penalization (with its variants, UST, Ridge and Lasso penalization) is implemented for sparsity and smoothness with a built in cross validation procedure to obtain the optimal penalization parameters. It is possible to obtain multiple latent variables which are orthogonal to each other, thus each explaining a different portion of variance in the predicted data set. sRDA is implemented in a Partial Least Squares framework, for more details see Csala et al. (2017).

Usage

\[
\text{sRDA}(\text{predictor}, \text{predicted}, \text{penalization} = \text{"enet"}, \text{ridge}_\text{penalty} = 1, \\
\text{nonzero} = 1, \text{max}_\text{iterations} = 100, \text{tolerance} = 1 \times 10^{-20}, \\
\text{cross}_\text{validate} = \text{FALSE}, \text{parallel}_\text{CV} = \text{FALSE}, \text{nr}_\text{subsets} = 10, \\
\text{multiple}_\text{LV} = \text{FALSE}, \text{nr}_\text{LVs} = 1)
\]

Arguments

- \text{predictor} \quad : \text{The n*p matrix of the predictor data set}
- \text{predicted} \quad : \text{The n*q matrix of the predicted data set}
- \text{penalization} \quad : \text{The penalization method applied during the analysis (none, enet or ust)
### ridge_penalty
The ridge penalty parameter of the predictor set's latent variable used for enet
(an integer if cross_validate = FALSE, a list otherwise)

### nonzero
The number of non-zero weights of the predictor set's latent variable used for
enet or ust (an integer if cross_validate = FALSE, a list otherwise)

### max_iterations
The maximum number of iterations of the algorithm (integer)

### tolerance
Convergence criteria (number, a small positive tolerance)

### cross_validate
K-fold cross validation to find best optimal penalty parameters (TRUE or FALSE)

### parallel.CV
Run the cross validation parallel (TRUE or FALSE)

### nr_subsets
Number of subsets for k-fold cross validation (integer, the value for k)

### multiple.LV
Obtain multiple latent variable pairs (TRUE or FALSE)

### nr.LVs
Number of latent variable pairs (components) to be obtained (integer)

### Value
An object of class "sRDA".

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XI</td>
<td>Predictor set’s latent variable scores</td>
</tr>
<tr>
<td>ETA</td>
<td>Predictive set’s latent variable scores</td>
</tr>
<tr>
<td>ALPHA</td>
<td>Weights of the predictor set’s latent variable</td>
</tr>
<tr>
<td>BETA</td>
<td>Weights of the predicted set’s latent variable</td>
</tr>
<tr>
<td>nr_iterations</td>
<td>Number of iterations ran before convergence (or max number of iterations)</td>
</tr>
<tr>
<td>SOLVE.XIXI</td>
<td>Inverse of the predictor set’s latent variable variance matrix</td>
</tr>
<tr>
<td>iterations.crts</td>
<td>The convergence criterion value (a small positive tolerance)</td>
</tr>
<tr>
<td>sum_absolute.betas</td>
<td>Sum of the absolute values of beta weights</td>
</tr>
<tr>
<td>ridge.penalty</td>
<td>The ridge penalty parameter used for the model</td>
</tr>
<tr>
<td>nr.nonzeros</td>
<td>The number of nonzero alpha weights in the model</td>
</tr>
<tr>
<td>nr.latent.variables</td>
<td>The number of latent variable pairs (components) in the model</td>
</tr>
<tr>
<td>CV_results</td>
<td>The detailed results of cross validations (if cross_validate is TRUE)</td>
</tr>
</tbody>
</table>

### Author(s)
Attila Csala

### References
Examples

# generate data with few highly correlated variables
dataXY <- generate_data(nr_LVs = 2,
  n = 250,
  nr_correlated_Xs = c(5,20),
  nr_uncorrelated_Xs = 250,
  mean_reg_weights_assoc_X =
    c(0.9,0.5),
  sd_reg_weights_assoc_X =
    c(0.05, 0.05),
  Xnoise_min = -0.3,
  Xnoise_max = 0.3,
  nr_correlated_Ys = c(10,15),
  nr_uncorrelated_Ys = 350,
  mean_reg_weights_assoc_Y =
    c(0.9,0.6),
  sd_reg_weights_assoc_Y =
    c(0.05, 0.05),
  Ynoise_min = -0.3,
  Ynoise_max = 0.3)

# separate predictor and predicted sets
X <- dataXY$X
Y <- dataXY$Y

# run sRDA
RDA.res <- sRDA(predictor = X, predicted = Y, nonzero = 5,
                 ridge_penalty = 1, penalization = "ust")

# check first 10 weights of X
RDA.res$ALPHA[1:10]

## Not run:
# run sRDA with cross-validation to determine best penalization parameters
RDA.res <- sRDA(predictor = X, predicted = Y, nonzero = c(5,10,15),
                 ridge_penalty = c(0.1,1), penalization = "enet", cross_validate = TRUE,
                 parallel.CV = TRUE)

# check first 10 weights of X
RDA.res$ALPHA[1:10]

# check the Ridge parameter and the number of nonzeros included in the model
RDA.res$ridge_penalty
RDA.res$nr_nonzeros

# check how much time the cross validation did take
RDA.res$CV_results$time

# obtain multiple latent variables (components)
RDA.res <- sRDA(predictor = X, predicted = Y, nonzero = c(5,10,15),
    ridge_penalty = c(0.1,1), penalization = "enet", cross_validate = TRUE,
    parallel_CV = TRUE, multiple_LV = TRUE, nr_LVs = 2, max_iterations = 5)

# check first 20 weights of X in first two component
RDA.res$ALPHA[[1]][1:20]
RDA.res$ALPHA[[2]][1:20]

# components are orthogonal to each other
t(RDA.res$XI[[1]]) %*% RDA.res$XI[[2]]

## End(Not run)
Index

*Topic **RDA**
  sCCA, 3
  sRDA, 6
*Topic **Redundancy**
  sCCA, 3
  sRDA, 6
*Topic **data**
  generate_data, 2
*Topic **generate**
  generate_data, 2

generate_data, 2

sCCA, 3
sRDA, 6
sRDA-package (sRDA), 6