Package 'FADA'

December 10, 2019

Type Package

Title Variable Selection for Supervised Classification in High Dimension

Version 1.3.5

Date 2019-12-10

Author Emeline Perthame (Institut Pasteur, Paris, France), Chloe Friguet (Universite de Bretagne Sud, Vannes, France) and David Causeur (Agrocampus Ouest, Rennes, France)

Maintainer David Causeur <david.causeur@agrocampus-ouest.fr>

Description The functions provided in the FADA (Factor Adjusted Discriminant Analysis) package aim at performing supervised classification of high-dimensional and correlated profiles. The procedure combines a decorrelation step based on a factor modeling of the dependence among covariates and a classification method. The available methods are Lasso regularized logistic model (see Friedman et al. (2010)), sparse linear discriminant analysis (see Clemmensen et al. (2011)), shrinkage linear and diagonal discriminant analysis (see M. Ahdesmaki et al. (2010)). More methods of classification can be used on the decorrelated data provided by the package FADA.

License GPL (>= 2)

Depends MASS, elasticnet

Imports sparseLDA,sda,glmnet,mnormt,crossval,corpcor, matrixStats,methods

NeedsCompilation no

Repository CRAN

Date/Publication 2019-12-10 15:30:05 UTC

R topics documented:

`FADA-package` *Variable selection for supervised classification in high dimension*

Description

The functions provided in the FADA (Factor Adjusted Discriminant Analysis) package aim at performing supervised classification of high-dimensional and correlated profiles. The procedure combines a decorrelation step based on a factor modeling of the dependence among covariates and a classification method. The available methods are Lasso regularized logistic model (see Friedman et al. (2010)), sparse linear discriminant analysis (see Clemmensen et al. (2011)), shrinkage linear and diagonal discriminant analysis (see M. Ahdesmaki et al. (2010)). More methods of classification can be used on the decorrelated data provided by the package FADA.

Details

Package:	FADA
Type:	Package
Version:	1.2
Date:	2014-10-08
License:	GPL (>= 2)

The functions available in this package are used in this order:

- Step 1: Decorrelation of the training dataset using a factor model of the covariance by the `decorrelate.train` function. The number of factors of the model can be estimated or forced.
- Step 2: If needed, decorrelation of the testing dataset by using the `decorrelate.test` function and the estimated factor model parameters provided by `decorrelate.train`.
- Step 3: Estimation of a supervised classification model using the decorrelated training dataset by the `FADA` function. One can choose among several classification methods (more details in the manual of `FADA` function).
- Step 4: If needed, computation of the error rate by the `FADA` function, either using a supplementary test dataset or by K-fold cross-validation.

Author(s)

Emeline Perthame (Agrocampus Ouest, Rennes, France), Chloe Friguet (Universite de Bretagne Sud, Vannes, France) and David Causeur (Agrocampus Ouest, Rennes, France)

Maintainer: David Causeur, http://math.agrocampus-ouest.fr/infoglueDeliverLive/membres/david.causeur, mailto: david.causeur@agrocampus-ouest.fr

References

Ahdesmaki, M. and Strimmer, K. (2010), Feature selection in omics prediction problems using cat scores and false non-discovery rate control. Annals of Applied Statistics, 4, 503-519.

Clemmensen, L., Hastie, T. and Witten, D. and Ersboll, B. (2011), Sparse discriminant analysis. Technometrics, 53(4), 406-413.

Friedman, J., Hastie, T. and Tibshirani, R. (2010), Regularization paths for generalized linear models via coordinate descent. Journal of Statistical Software, 33, 1-22.

Friguet, C., Kloareg, M. and Causeur, D. (2009), A factor model approach to multiple testing under dependence. Journal of the American Statistical Association, 104:488, 1406-1415.

Perthame, E., Friguet, C. and Causeur, D. (2015), Stability of feature selection in classification issues for high-dimensional correlated data, Statistics and Computing.

Examples

```
### Not run
### example of an entire analysis with FADA package if a testing data set is available
### loading data
# data(data.train)
# data(data.test)

# dim(data.train$x) # 30 250
# dim(data.test$x) # 1000 250

### decorrelation of the training data set
# res = decorrelate.train(data.train) # Optimal number of factors is 3
### decorrelation of the testing data set afterward
# res2 = decorrelate.test(res,data.test)

### classification step with sda, using local false discovery rate for variable selection
### linear discriminant analysis
# FADA.LDA = FADA(res2,method="sda",sda.method="lfdr")

### diagonal discriminant analysis
# FADA.DDA =  FADA(res2, method="sda",sda.method="lfdr",diagonal=TRUE)


### example of an entire analysis with FADA package if no testing data set is available
### loading data

### decorrelation step
# res = decorrelate.train(data.train) # Optimal number of factors is 3

### classification step with sda, using local false discovery rate for variable selection
### linear discriminant analysis, error rate is computed by 10-fold CV (20 replications of the CV)
# FADA.LDA = FADA(res,method="sda",sda.method="lfdr")
```

`data.test` *Test dataset simulated with the same distribution as the training dataset data.train.*

Description

The test dataset has the same list structure as the training dataset dta. Only the numbers of rows of the x component and length of the y component are different since the test sample size is 1000.

Usage

```
data(data.test)
```

Format

List with 2 components: `x`, the 1000x250 matrix of simulated explanatory variables and `y`, the 1000x1 grouping variable (coded 1 and 2).

Examples

```
data(data.test)
dim(data.test$x) # 1000 250
data.test$y # 2 levels
```

`data.train` *Training data*

Description

Simulated training dataset. The x component is a matrix of explanatory variables, with 30 rows and 250 columns. Each row is simulated according to a multinormal distribution which mean depends on a group membership given by the y component. The variance matrix is the same within each group.

Usage

```
data(data.train)
```

Format

A list with 2 components. `x` is a 30x250 matrix of simulated explanatory variables. `y` is a 30x1 grouping variable (coded 1 and 2).

Examples

```
data(data.train)
dim(data.train$x) # 30 250
data.train$y # 2 levels
hist(cor(data.train$x[data.train$y==1,])) # high dependence
hist(cor(data.train$x[data.train$y==2,]))
```

`decorrelate.test` *Factor Adjusted Discriminant Analysis 2: Decorrelation of a testing data set after running the* `decorrelate.train` *function on a training data set*

Description

This function decorrelates the test dataset by adjusting data for the effects of latent factors of dependence, after running the `decorrelate.train` function on a training data set.

Usage

```
decorrelate.test(faobject,data.test)
```

Arguments

`faobject`	An object returned by function `decorrelate.train`.
`data.test`	A list containing the testing dataset, with the following component: `x` is a n x p matrix of explanatory variables, where n stands for the testing sample size and p for the number of explanatory variables.

Value

Returns a list with the following elements:

`meanclass`	Group means estimated after iterative decorrelation
`fa.training`	Decorrelated training data
`fa.testing`	Decorrelated testing data
`Psi`	Estimation of the factor model parameters: specific variance
`B`	Estimation of the factor model parameters: loadings
`factors.training`	Scores of the trainings individuals on the factors
`factors.testing`	Scores of the testing individuals on the factors
`groups`	Recall of group variable of training data
`proba.training`	Internal value (estimation of individual probabilities for the training dataset)
`proba.testing`	Internal value (estimation of individual probabilities for the testing dataset)
`mod.decorrelate.test`	Internal value (classification model)

Author(s)

Emeline Perthame, Chloe Friguet and David Causeur

References

Friedman, J., Hastie, T. and Tibshirani, R. (2010), Regularization paths for generalized linear models via coordinate descent. Journal of Statistical Software, 33, 1-22.

Friguet, C., Kloareg, M. and Causeur, D. (2009), A factor model approach to multiple testing under dependence. Journal of the American Statistical Association, 104:488, 1406-1415.

Perthame, E., Friguet, C. and Causeur, D. (2015), Stability of feature selection in classification issues for high-dimensional correlated data, Statistics and Computing.

See Also

FADA-package FADA glmnet-package

Examples

```
data(data.train)
data(data.test)
fa = decorrelate.train(data.train)
fa2 = decorrelate.test(fa,data.test)
names(fa2)
```

`decorrelate.train` *Factor Adjusted Discriminant Analysis 1: Decorrelation of the training data*

Description

This function decorrelates the training dataset by adjusting data for the effects of latent factors of dependence.

Usage

```
decorrelate.train(data.train, nbf = NULL, maxnbfactors=12, diagnostic.plot = FALSE,
min.err = 0.001, verbose = TRUE,EM = TRUE, maxiter = 15,...)
```

Arguments

`data.train`	A list containing the training dataset with the following components: `x` is the n x p matrix of explanatory variables, where n stands for the training sample size and p for the number of explanatory variables ; `y` is a numeric vector giving the group of each individual numbered from 1 to K.
`nbf`	Number of factors. If `nbf = NULL`, the number of factors is estimated. `nbf` can also be set to a positive integer value. If `nbf = 0`, the data are not factor-adjusted.

`maxnbfactors` The maximum number of factors. Default is `maxnbfactors=12`.

`diagnostic.plot`
If `diagnostic.plot =TRUE`, the values of the variance inflation criterion are plotted for each number of factors. Default is `diagnostic.plot =FALSE`. This option might be helpful to manually determine the optimal number of factors.

`min.err` Threshold of convergence of the algorithm criterion. Default is min.err=0.001.

`verbose` Print out number of factors and values of the objective criterion along the iterations. Default is `TRUE`.

`EM` The method used to estimate the parameters of the factor model. If `EM=TRUE`, parameters are estimated by an EM algorithm. Setting `EM=TRUE` is recommended when the number of covariates exceeds the number of observations. If `EM=FALSE`, the parameters are estimated by maximum-likelihood using `factanal`. Default is `EM=TRUE`

`maxiter` Maximum number of iterations for estimation of the factor model.

`...` Other arguments that can be passed in the `cv.glmnet` and `glmnet` functions from glmnet package. These functions are used to estimate individual group probabilities. Modifying these parameters should not affect the decorrelation procedure. However, the argument `nfolds` in `cv.glmnet` is set to 10 by default and should be reduced (minimum 3) for large datasets, in order to decrease the computation time of `decorrelation.train`.

Value

Returns a list with the following elements:

`meanclass` Group means estimated after iterative decorrelation

`fa.training` Decorrelated training data

`Psi` Estimation of the factor model parameters: specific variance

`B` Estimation of the factor model parameters: loadings

`factors.training`
Scores of the trainings individuals on the factors

`groups` Recall of group variable of training data

`proba.training` Internal value (estimation of individual probabilities for the training dataset)

Author(s)

Emeline Perthame, Chloe Friguet and David Causeur

References

Friedman, J., Hastie, T. and Tibshirani, R. (2010), Regularization paths for generalized linear models via coordinate descent. Journal of Statistical Software, 33, 1-22.

Friguet, C., Kloareg, M. and Causeur, D. (2009), A factor model approach to multiple testing under dependence. Journal of the American Statistical Association, 104:488, 1406-1415.

Perthame, E., Friguet, C. and Causeur, D. (2015), Stability of feature selection in classification issues for high-dimensional correlated data, Statistics and Computing.

See Also

FADA-package FADA glmnet-package factanal

Examples

```
data(data.train)

res0 = decorrelate.train(data.train,nbf=3) #  when the number of factors is forced

res1 = decorrelate.train(data.train) #  when the optimal number of factors is unknown
```

FADA *Factor Adjusted Discriminant Analysis 3-4 : Supervised classification on decorrelated data*

Description

This function performs supervised classification on factor-adjusted data.

Usage

```
FADA(faobject, K=10,B=20, nbf.cv = NULL,method = c("glmnet",
    "sda", "sparseLDA"), sda.method = c("lfdr", "HC"), alpha=0.1, ...)
```

Arguments

faobject	An object returned by function decorrelate.train or decorrelate.test.
K	Number of folds to estimate classification error rate, only when no testing data is provided. Default is K=10.
B	Number of replications of the cross-validation. Default is B=20.
nbf.cv	Number of factors for cross validation to compute error rate, only when no testing data is provided. By default, nbf = NULL and the number of factors is estimated for each fold of the cross validation. nbf can also be set to a positive integer value. If nbf = 0, the data are not factor-adjusted.
method	The method used to perform supervised classification model. 3 options are available. If method = "glmnet", a Lasso penalized logistic regression is performed using **glmnet** R package. If method = "sda", a LDA or DDA (see diagonal argument) is performed using Shrinkage Discriminant Analysis using **sda** R package. If method = "sparseLDA", a Lasso penalized LDA is performed using **SparseLDA** R package.
sda.method	The method used for variable selection, only if method="sda". If sda.method="lfdr", variables are selected through CAT scores and False Non Discovery Rate control. If sda.method="HC", the variable selection method is Higher Cristicism Thresholding.
alpha	The proportion of the HC objective to be observed, only if method="sda" and sda.method="HC". Default is 0.1.

`...` Some arguments to tune the classification method. See the documentation of the chosen method (glmnet, sda or sda) for more informations about these parameters.

Value

Returns a list with the following elements:

`method`	Recall of the classification method
`selected`	A vector containing index of the selected variables
`proba.train`	A matrix containing predicted group frequencies of training data.
`proba.test`	A matrix containing predicted group frequencies of testing data, if a testing data set has been provided
`predict.test`	A matrix containing predicted classes of testing data, if a testing data set has been provided
`cv.error`	A numeric value containing the average classification error, computed by cross validation, if no testing data set has been provided
`cv.error.se`	A numeric value containing the standard error of the classification error, computed by cross validation, if no testing data set has been provided
`mod`	The classification model performed. The class of this element is the class of a model returned by the chosen method. See the documentation of the chosen method for more details.

Author(s)

Emeline Perthame, Chloe Friguet and David Causeur

References

Ahdesmaki, M. and Strimmer, K. (2010), Feature selection in omics prediction problems using cat scores and false non-discovery rate control. Annals of Applied Statistics, 4, 503-519.

Clemmensen, L., Hastie, T. and Witten, D. and Ersboll, B. (2011), Sparse discriminant analysis. Technometrics, 53(4), 406-413.

Friedman, J., Hastie, T. and Tibshirani, R. (2010), Regularization paths for generalized linear models via coordinate descent. Journal of Statistical Software, 33, 1-22.

Friguet, C., Kloareg, M. and Causeur, D. (2009), A factor model approach to multiple testing under dependence. Journal of the American Statistical Association, 104:488, 1406-1415.

Perthame, E., Friguet, C. and Causeur, D. (2015), Stability of feature selection in classification issues for high-dimensional correlated data, Statistics and Computing.

See Also

`FADA`, `decorrelate.train`, `decorrelate.test`, `sda`, `sda-package`, `glmnet-package`

Examples

```
data(data.train)
data(data.test)

# When testing data set is provided
res = decorrelate.train(data.train)
res2 = decorrelate.test(res, data.test)
classif = FADA(res2,method="sda",sda.method="lfdr")

### Not run
# When no testing data set is provided
# Classification error rate is computed by a K-fold cross validation.
# res = decorrelate.train(data.train)
# classif = FADA(res, method="sda",sda.method="lfdr")
```

Index