Package ‘FusionLearn’

January 20, 2025

Type Package

Title Fusion Learning

Version 0.2.1

Author Xin Gao, Yuan Zhong, and Raymond J. Carroll

Maintainer Yuan Zhong <aqua.zhong@gmail.com>

Description The fusion learning method uses a model selection algorithm to learn from multiple data sets across different experimental platforms through group penalization. The responses of interest may include a mix of discrete and continuous variables. The responses may share the same set of predictors, however, the models and parameters differ across different platforms. Integrating information from different data sets can enhance the power of model selection. Package is based on Xin Gao, Raymond J. Carroll (2017) <arXiv:1610.00667v1>.

License GPL (>= 2)

Suggests knitr, rmarkdown, MASS, ggplot2, mvtnorm

VignetteBuilder knitr

Encoding UTF-8

LazyData true

Depends R (>= 3.5.0)

NeedsCompilation no

Repository CRAN

Date/Publication 2022-04-24 22:10:02 UTC

Contents

FusionLearn-package *Fusion Learning*

Description

FusionLearn package implements a new learning algorithm to integrate information from different experimental platforms. The algorithm applies the grouped penalization method in the pseudolikelihood setting.

Details

In the context of fusion learning, there are k different data sets from k different experimental platforms. The data from each platform can be modeled by a different generalized linear model. Assume the same set of predictors $\{M_1, M_2, ..., M_j, ..., M_p\}$ are measured across k different experimental platforms.

Platforms	Formula	M_1	M_2	$\ldots$	M_j	$\ldots$	M_p
1	$y_1 : g_1(\mu_1) \sim$	$x_{11}\beta_{11}+$	$x_{12}\beta_{12}+$	$\ldots$	$x_{1j}\beta_{1j}+$	$\ldots$	$x_{1p}\beta_{1p}$
2	$y_2 : g_2(\mu_2) \sim$	$x_{21}\beta_{21}+$	$x_{22}\beta_{22}+$	$\ldots$	$x_{2j}\beta_{2j}+$	$\ldots$	$x_{2p}\beta_{2p}$
	...						
k	$y_k : g_k(\mu_k) \sim$	$x_{k1}\beta_{k1}+$	$x_{k2}\beta_{k2}+$	$\ldots$	$x_{kj}\beta_{kj}+$	$\ldots$	$x_{kp}\beta_{kp}$

Here x_{kj} represents the observation of the predictor M_j on the kth platform, and $\beta^{(j)}$ denotes the vector of regression coefficients for the predictor M_j.

Platforms	$\boldsymbol{M_j}$	$\boldsymbol{\beta^{(j)}}$
1	x_{1j}	β_{1j}
2	x_{2j}	β_{2j}
	...	...
k	x_{kj}	β_{kj}

Consider the following examples.

Example 1. Suppose k different types of experiments are conducted to study the genetic mechanism of a disease. The predictors in this research are different facets of individual genes, such as mRNA expression, protein expression, RNAseq expression and so on. The goal is to select the genes which affect the disease, while the genes are assessed in a number of ways through different measurement processes across k experimental platforms.

Example 2. The predictive models for three different financial indices are simultaneously built from a panel of stock index predictors. In this case, the predictor values across different models are the same, but the regression coefficients are different.

In the conventional approach, the model for each of the k platforms is analyzed separately. FusionLearn algorithm selects significant predictors through learning from multiple models. The overall objective is to minimize the function:

$$Q(\beta) = l_I(\beta) - n \sum_{j=1}^{p} \Omega_{\lambda_n} ||\beta^{(j)}||,$$

with p being the numbers of predictors, Ω_{λ_n} being the penalty functions, and $||\beta^{(j)}|| = (\sum_{i=1}^{k} \beta_{ij}^2)^{1/2}$ denoting the L_2-norm of the coefficients of the predictor M_j.

The user can specify the penalty function Ω_{λ_n} and the penalty values λ_n. This package also contains functions to provide the pseudolikelihood Bayesian information criterion:

$$pseu - BIC(s) = -2l_I(\hat{\beta}_I; Y) + d_s^* \gamma_n$$

with $-2l_I(\hat{\beta}_I; Y)$ denoting the pseudo loglikelihood, d_s^* measuring the model complexity and γ_n being the penalty on the model complexity.

The basic function `fusionbase` deals with continuous responses. The function `fusionbinary` is applied to binary responses, and the function `fusionmixed` is applied to a mix of continuous and binary responses.

Note

Here we provide two examples to illustrate the data structures. Assume X_I and X_{II} represent two sets of the predictors from 2 experimental platforms.

Example 1. If the observations from X_I and X_{II} are independent, the number of observations can be different. The order of the predictors $\{M_1, M_2, M_3, M_4\}$ in X_I matches with the predictors in X_{II}. If X_{II} does not include the predictor M_3, then the M_3 in X_{II} needs to be filled with `NA`.

$$X_I = \begin{array}{cccc} M_1 & M_2 & M_3 & M_4 \\ 0.1 & 0.3 & 0.5 & 20 \\ 0.3 & 0.1 & 0.5 & 7 \\ 0.1 & 0.9 & 1 & 0 \\ -0.3 & 1.2 & 2 & 40 \end{array} \quad X_{II} = \begin{array}{cccc} M_1 & M_2 & M_3 & M_4 \\ 100 & 8 & \texttt{NA} & 100 \\ 30 & 1 & \texttt{NA} & 2 \\ 43 & 19 & \texttt{NA} & -3 \end{array}$$

Example 2. If the observations from X_I and X_{II} are correlated, the number of observations must be the same. The ith row in X_I is correlatd with the ith row in X_{II}. The predictors of X_I and X_{II} should be matched in order. The predictors which are not measured need to be filled with `NA`.

$$X_I = \begin{array}{cccc} M_1 & M_2 & M_3 & M_4 \\ 0.1 & 0.3 & 0.5 & 20 \\ 0.3 & 0.1 & 0.5 & 70 \\ -0.1 & 0.9 & 1 & 0 \\ -0.3 & 1.2 & 2 & 40 \end{array} \quad X_{II} = \begin{array}{cccc} M_1 & M_2 & M_3 & M_4 \\ 0.3 & 0.8 & \texttt{NA} & 100 \\ 0.2 & 1 & \texttt{NA} & 20 \\ 0.43 & 1.9 & \texttt{NA} & -30 \\ -0.4 & -2 & \texttt{NA} & 40 \end{array}$$

In functions `fusionbase.fit`, `fusionbinary.fit`, and `fusionmixed.fit`, the option `depen` is used to specify whether observations from different platforms are correlated or independent.

Author(s)

Xin Gao, Yuan Zhong and Raymond J Carroll

Maintainer: Yuan Zhong <aqua.zhong@gmail.com>

References

Gao, X and Carroll, R. J. (2017) Data integration with high dimensionality. Biometrika, 104, 2, pp. 251-272

fusionbase *Fusion learning method for continuous responses*

Description

fusionbase conducts the group penalization to multiple linear models with a specified penalty value. fusionbase.fit can be used to search the best candidate model based on the pseudo Bayesian information criterion with a sequence of penalty values.

Usage

```
fusionbase(x, y, lambda, N, p, m, beta=0.1, thresh=0.05,
           maxiter=30, methods="scad",Complete=TRUE)

fusionbase.fit(x, y, lambda, N, p, m, beta=0.1, thresh=0.05,
               maxiter=30, methods="scad", Complete=TRUE, depen ="IND", a=1)
```

Arguments

x List. Listing matrices of the predictors from different platforms.

y List. A list of continuous responses vectors from different platforms following the same order as in x.

lambda Numeric or vector. For fusionbase, lambda is a numeric value for the penalty; for fusionbase.fit, lambda is a vector with a list of penalty values.

N Numeric or vector. If only one numeric value is provided, equal sample size will be assumed for each data set. If a vector is provided, then the elements are the sample sizes for all the platforms.

p Numeric. The number of predictors.

m Numeric. The number of platforms.

beta Numeric or Matrix. An initial value for the estimated parameters with dimensions nvars x nplatforms. The defaul value is 0.1.

thresh Numeric. The stopping criteria. The default value is 0.05.

maxiter Numeric. Maximum number of iterations. The default value is 30.

methods Character ("lass" or "scad"). lass: LASSO; scad: SCAD.

Complete Logic input. If Complete == TRUE, the predictors M_1,...,M_p are measured in all platforms. If Compelte == FALSE, in some platforms, not all of the predictors $\{M_1, M_2, ..., M_p\}$ are measured. The values of the corresponding estimated coefficients for the missing predictors will be NA.

depen Character. Input only for function fusionbase.fit. "IND" means the observations across different platforms are independent; "CORR" means the observations are correlated, and the sample sizes should be equal for different platforms.

a Numeric. Input only for function fusionbase.fit. The free multiplicative constant used in γ_n. The default value is 1.

Details

The basic fusion learning function to learn from multiple linear models with continuous responses. More details regarding the model assumptions and the algorithm can be found in FusionLearn.

Value

fusionbase returns a list that has components:

beta — A matrix (nvars x nplatforms) containing estimated coefficients of each linear model. If some data sets do not have the complete set of predictors, the corresponding coefficients are output as NA.

method — Penalty function LASSO or SCAD.

threshold — The numeric value shows the difference in the estimates between the successive updates upon convergence.

iteration — The numeric value shows the number of iterations upon convergence.

fusionbase.fit provides the results in a table:

lambda — The sequence of penalty values.

BIC — The pseudolikelihood Bayesian information criterion evaluated at the sequence of the penalty values.

-2Loglkh — Minus twice the pseudo loglikelihood of the chosen model.

Est_df — The estimated degrees of freedom quantifying the model complexity.

fusionbase.fit also returns a model selection plot showing the results above.

Note

The range of the penalty values should be carefully chosen. For some penalty values, the resulting models may have singular information matrix or the fitting of the glm cannot converge.

Author(s)

Xin Gao, Yuan Zhong, and Raymond J. Carroll

References

Gao, X and Carroll, R. J. (2017) Data integration with high dimensionality. Biometrika, 104, 2, pp. 251-272

Examples

```
##analysis of the stock index data
#Responses contain indices "VIX","GSPC", and "DJI"
y <- list(stockindexVIX[,1],stockindexGSPC[,1],stockindexDJI[,1])

#Predictors include 46 stocks
x <- list(stockindexVIX[,2:47],stockindexGSPC[,2:47],stockindexDJI[,2:47])
```

```
##Implementing the model selection algorithm based on the psuedolikelihood
##information criteria
model <- fusionbase.fit(x,y,seq(0.03,5,length.out = 10),232,46,3,depen="CORR")
lambda <- model[which.min(model[,2]),1]
result <- fusionbase(x,y,lambda,232,46,3)

##Identify the significant predictors for the three indices
id <- which(result$beta[,1]!=0)+1
colnames(stockindexVIX)[id]
```

fusionbinary *Fusion learning algorithm for binary responses*

Description

fusionbinary conducts the group penalization with a specified penalty value learning from multiple generalized linear models with binary responses. fusionbinary.fit can be used to search the best candidate model based on the pseudo Bayesian information criterion with a sequence of penalty values.

Usage

```
fusionbinary(x, y, lambda, N, p, m, beta=0.1, thresh=0.1,
             maxiter=100, methods="scad", link="logit", Complete=TRUE)

fusionbinary.fit(x, y, lambda, N, p, m, beta=0.1, thresh=0.1,
                 maxiter=100, methods="scad", link="logit", Complete=TRUE,
                 depen ="IND", a=1)
```

Arguments

x	List. Listing matrices of the predictors from different platforms.
y	List. A list of binary responses vectors from different platforms following the same order as in x.
lambda	Numeric or vector. For fusionbinary, lambda is a numeric value for the penalty; for fusionbinary.fit, lambda is a vector with a list of penalty values.
N	Numeric or vector. If only one numeric value is provided, equal sample size will be assumed for each data set. If a vector is provided, then the elements are the sample sizes for all the platforms.
p	Numeric. The number of predictors.
m	Numeric. The number of platforms.
beta	Numeric. An initial value for the estimated parameters with dimensions nvars x nplatforms.
thresh	Numeric. The stopping criteria. The default value is 0.1.
maxiter	Numeric. Maximum number of iterations. The default value is 100.

`methods` Character ("lass" or "scad"). `lass`: LASSO; `scad`: SCAD.

`link` Character ("logit" or "probit"). Link functions: logistic or probit.

`Complete` Logic input. If `Complete == TRUE`, the predictors M_1,...,M_p are measured in all platforms. If `Compelte == FALSE`, in some platforms, not all of the predictors $\{M_1, M_2, ..., M_p\}$ are measured. The values of the corresponding estimated coefficients for the missing predictors will be `NA`.

`depen` Character. Input only for function `fusionbinary.fit`. "IND" means the observations across different platforms are independent; "CORR" means the observations are correlated, and the sample sizes should be equal for different platforms.

`a` Numeric. Input only for function `fusionbinary.fit`. The free multiplicative constant used in γ_n. The default value is 1.

Details

The generalized fusion learning function to learn from multiple models with binary responses. More details regarding the algorithm can be found in `FusionLearn`.

Value

`fusionbinary` returns a list that has components:

`beta` A matrix (nvars x nplatforms) containing estimated coefficients of each linear model. If some data sets do not have the complete set of predictors, the corresponding coefficients are output as `NA`.

`method` Penalty function LASSO or SCAD.

`link` The link function used in the estimation.

`threshold` The numeric value shows the difference in the estimates between the successive updates upon convergence.

`iteration` The numeric value shows the number of iterations upon convergence.

`fusionbinary.fit` provides the results in a table:

`lambda` The sequence of penalty values.

`BIC` The pseudolikelihood Bayesian information criterion evaluated at the sequence of the penalty values.

`-2Loglkh` Minus twice the pseudo loglikelihood of the chosen model.

`Est_df` The estimated degrees of freedom quantifying the model complexity.

`fusionbinary.fit` also returns a model selection plot showing the results above.

Note

The range of the penalty values should be carefully chosen. For some penalty values, the resulting models may have singular information matrix or the fitting of the glm cannot converge.

Author(s)

Xin Gao, Yuan Zhong, and Raymond J. Carroll

References

Gao, X and Carroll, R. J. (2017) Data integration with high dimensionality. Biometrika, 104, 2, pp. 251-272

Examples

```
##Analysis of the gene data
y = list(mockgene1[,2],mockgene2[,2])           ## responses "status"
x = list(mockgene1[,3:502],mockgene2[,3:502])   ## 500 predictors


##Implementing fusion learning algorithm
result <- fusionbinary(x,y,0.3,N=c(98,286),500,2)
id <- which(result$beta[,1]!=0)+2
genename <- colnames(mockgene1)[id]
```

fusionmixed *Fusion learning algorithm for mixed data*

Description

fusionmixed conducts the group penalization with a specified penalty value learning from multiple generalized linear models with mixed continuous and binary responses. fusionmixed.fit can be used to search the best candidate model based on the pseudo Bayesian information criterion with a sequence of penalty values.

Usage

```
fusionmixed(x, y, lambda, N, p, m1, m2, beta=0.1, thresh=0.1,
            maxiter=100, methods="scad", link="logit", Complete=TRUE)

fusionmixed.fit(x, y, lambda, N, p, m1, m2, beta=0.1, thresh=0.1,
                maxiter=100, methods="scad",link="logit", Complete=TRUE,
                depen ="IND", a=1)
```

Arguments

x	List. Listing matrices of the predictors from different platforms. The first m1 data sets in the list are the ones of continuous responses, and the following m2 data sets are the ones of binary responses.
y	List. A list of the responses vectors from different platforms following the same order as in x. The values m1 and m2 must be specified.
lambda	Numeric or vector. For fusionmixed, lambda is a numeric value for the penalty; for fusionmixed.fit, lambda is a vector with a list of penalty values.

`N` Numeric or vector. If only one numeric value is provided, equal sample size will be assumed for each data set. If a vector is provided, then the elements are the sample sizes for all the platforms.

`p` Numeric. The number of predictors.

`m1` Numeric. Number of platforms whose response variables are continuous.

`m2` Numeric. Number of platforms whose response variables are binary.

`beta` Numeric. An initial value for the estimated parameters with dimensions nvars x nplatforms. The default value is 0.1.

`thresh` Numeric. The stopping criteria. The default value is 0.1.

`maxiter` Numeric. Maximum number of iterations. The default value is 100.

`methods` Character ("lass" or "scad"). `lass`: LASSO; `scad`: SCAD.

`link` Character ("logit" or "probit"). Link functions: logistic or probit.

`Complete` Logic input. If `Complete` == `TRUE`, the predictors M_1,...,M_p are measured in all platforms. If `Compelte` == `FALSE`, in some platforms, not all of the predictors $\{M_1, M_2, ..., M_p\}$ are measured. The values of the corresponding estimated coefficients for the missing predictors will be `NA`.

`depen` Character. Input only for function `fusionmixed.fit`. "IND" means the observations across different platforms are independent; "CORR" means the observations are correlated, and the sample sizes should be equal for different platforms.

`a` Numeric. Input only for function `fusionmixed.fit`. The free multiplicative constant used in γ_n. The default value is 1.

Details

`fusionmixed` is designed for a more complex data structure by aggregating information from continuous and binary responses. More details regarding the algorithm can be found in `FusionLearn`.

Value

`fusionmixed` returns a list that has components:

`beta` A matrix (nvars x nplatforms) containing estimated coefficients of each linear model. If some data sets do not have the complete set of predictors, the corresponding coefficients are output as `NA`.

`method` Penalty function LASSO or SCAD.

`link` The link function used in the estimation.

`threshold` The numeric value shows the difference in the estimates between the successive updates upon convergence.

`iteration` The numeric value shows the number of iterations upon convergence.

`fusionmixed.fit` provides the results in a table:

`lambda` The sequence of penalty values.

`BIC` The pseudolikelihood Bayesian information criterion evaluated at the sequence of the penalty values.

`-2Loglkh` Minus twice the pseudo loglikelihood of the chosen model.

`Est_df` The estimated degrees of freedom quantifying the model complexity.

`fusionmixed.fit` also returns a model selection plot showing the results above.

Note

The range of the penalty values should be carefully chosen. For some penalty values, the resulting models may have singular information matrix or the fitting of the glm cannot converge.

Author(s)

Xin Gao, Yuan Zhong, and Raymond J. Carroll

References

Gao, X and Carroll, R. J. (2017) Data integration with high dimensionality. Biometrika, 104, 2, pp. 251-272

See Also

`fusionbase`,`fusionbinary`,

Examples

```
##Analysis of the index data

#Responses contain indices "VIX","GSPC", and "DJI",
#"DJI" is dichotomized into "increasing" or "decreasing"
y <- list(stockindexVIX[,1],stockindexGSPC[,1],stockindexDJI[,1]>0)

#Predictors include 46 stocks
x <- list(stockindexVIX[,2:47],stockindexGSPC[,2:47],stockindexDJI[,2:47])
##Implementing the model selection based on psuedolikelihood
##information criteria
model <- fusionmixed.fit(x,y,seq(0.03,5,length.out = 10),232,46,2,1,depen="CORR")
lambda <- model[which.min(model[,2]),1]
result <- fusionmixed(x,y,lambda,232,46,2,1)

##Identify the significant predictors for three indices
id <- which(result$beta[,1]!=0)+1
colnames(stockindexVIX)[id]
```

mockgene *Mock Gene Data*

Description

This dataset is a mock version of two different microarray experiments on breast cancer cells.

Usage

```
data("mockgene1")
data("mockgene2")
```

Format

The first data "mockgene1" contains 98 subjects, and the second data "mockgene2" contains 286 subjects.

The first column for each data is ID number.

The second column is subjects' status. If the status is the estrogen-receptor-positive, $y = 1$; if the status is estrogen-receptor-negative, $y = 0$. Other columns record the gene expression values.

Details

This is an example to implement the `FusionLearn` algorithm for binary responses. In this case, the two experiments followed different protocols, and the two sets of gene expression profiles are different. The objective is to select a suitable subset gene predictors for the disease analysis based on both experiments.

Source

This data is a mock version of the original data. The original gene data contain over 20,000 profile expressions, and more details can be found on https://www.ncbi.nlm.nih.gov/ with series numbers GSE2034 and GSE22093.

stockindex *Finance Data*

Description

This is a dataset containing the log return on three financial market indices and 46 stocks between 2013 and 2015. The responses are the financial indices, "VIX", "SP500", and "DJI", and the predictors are 46 stocks from the market. The data are given in three-day gap from 700 trading days. We also provide the validation datasets of three indices.

Usage

```
data("stockindexVIX")
data("stockindexGSPC")
data("stockindexDJI")
data("validVIX")
data("validGSPC")
data("validDJI")
```

Details

This example is used to demonstrate the use of the functions `fusionbase` and `fusionmixed`. This dataset has correlated responses and the same predictors values for three models.

Source

This data is available on Yahoo Finance.

Index