

Package ‘joinet’

November 23, 2020

Version 0.0.6

Title Multivariate Elastic Net Regression

Description Implements high-dimensional multivariate regression by stacked generalisation (Wolpert 1992 <doi:10.1016/S0893-6080(05)80023-1>). For positively correlated outcomes, a single multivariate regression is typically more predictive than multiple univariate regressions. Includes functions for model fitting, extracting coefficients, outcome prediction, and performance measurement. If required, install MRCE from GitHub (<<https://github.com/cran/MRCE>>).

Depends R (>= 3.0.0)

Imports glmnet, palasso, cornet

Suggests knitr, rmarkdown, testthat, MASS

Enhances mice, earth, spls, MRCE, remMap, MultivariateRandomForest, SiER, mcn, GPM, RMTL, MTPS

VignetteBuilder knitr

License GPL-3

LazyData true

Language en-GB

RoxygenNote 7.1.1

URL <https://github.com/rauschenberger/joinet>

BugReports <https://github.com/rauschenberger/joinet/issues>

NeedsCompilation no

Author Armin Rauschenberger [aut, cre]

Maintainer Armin Rauschenberger <armin.rauschenberger@uni.lu>

Repository CRAN

Date/Publication 2020-11-23 10:50:16 UTC

R topics documented:

joinet-package	2
coef.joinet	3

cv.jointet	4
jointet	6
predict.jointet	8
weights.jointet	8

Index	10
--------------	-----------

jointet-package	<i>Multivariate Elastic Net Regression</i>
-----------------	--

Description

The R package `jointet` implements multivariate ridge and lasso regression using stacked generalisation. This multivariate regression typically outperforms univariate regression at predicting correlated outcomes. It provides predictive and interpretable models in high-dimensional settings.

Details

Use function `jointet` for model fitting. Type `library(jointet)` and then `?jointet` or `help("jointet")` to open its help file.

See the vignette for further examples. Type `vignette("jointet")` or `browseVignettes("jointet")` to open the vignette.

References

Armin Rauschenberger and Enrico Glaab (2020). "Predicting correlated outcomes from molecular data". *Manuscript in preparation*.

<armin.rauschenberger@uni.lu>

Examples

```
#--- data simulation ---
n <- 50; p <- 100; q <- 3
X <- matrix(rnorm(n*p), nrow=n, ncol=p)
Y <- replicate(n=q, expr=rnorm(n=n, mean=rowSums(X[,1:5])))
# n samples, p inputs, q outputs

if(!grepl('SunOS', Sys.info()['sysname'])){
#--- model fitting ---
object <- jointet(Y=Y, X=X)
# slot "base": univariate
# slot "meta": multivariate

#--- make predictions ---
y_hat <- predict(object, newx=X)
# n x q matrix "base": univariate
# n x q matrix "meta": multivariate

#--- extract coefficients ---
```

```

coef <- coef(object)
# effects of inputs on outputs
# q vector "alpha": intercepts
# p x q matrix "beta": slopes

#--- model comparison ---
loss <- cv.joinet(Y=Y,X=X)
# cross-validated loss
# row "base": univariate
# row "meta": multivariate
}

```

coef.joinet

Extract Coefficients

Description

Extracts pooled coefficients. (The meta learners linearly combines the coefficients from the base learners.)

Usage

```

## S3 method for class 'joinet'
coef(object, ...)

```

Arguments

object	joinet object
...	further arguments (not applicable)

Value

This function returns the pooled coefficients. The slot alpha contains the intercepts in a vector of length q , and the slot beta contains the slopes in a matrix with p rows (inputs) and q columns.

Examples

```

## Not run:
n <- 50; p <- 100; q <- 3
X <- matrix(rnorm(n*p), nrow=n, ncol=p)
Y <- replicate(n=q, expr=rnorm(n=n, mean=rowSums(X[, 1:5])))
object <- joinet(Y=Y,X=X)
coef <- coef(object)
## End(Not run)

```

 cv.joinet

 Model comparison

Description

Compares univariate and multivariate regression.

Usage

```
cv.joinet(
  Y,
  X,
  family = "gaussian",
  nfolds.ext = 5,
  nfolds.int = 10,
  foldid.ext = NULL,
  foldid.int = NULL,
  type.measure = "deviance",
  alpha.base = 1,
  alpha.meta = 1,
  compare = FALSE,
  mice = FALSE,
  cvpred = FALSE,
  times = FALSE,
  ...
)
```

Arguments

Y	outputs: numeric matrix with n rows (samples) and q columns (variables), with positive correlation (see details)
X	inputs: numeric matrix with n rows (samples) and p columns (variables)
family	distribution: vector of length 1 or q with entries "gaussian", "binomial" or "poisson"
nfolds.ext	number of external folds
nfolds.int	number of internal folds
foldid.ext	external fold identifiers: vector of length n with entries between 1 and nfolds.ext; or NULL
foldid.int	internal fold identifiers: vector of length n with entries between 1 and nfolds.int; or NULL
type.measure	loss function: vector of length 1 or q with entries "deviance", "class", "mse" or "mae" (see cv.glmnet)
alpha.base	elastic net mixing parameter for base learners: numeric between 0 (ridge) and 1 (lasso)

alpha.meta	elastic net mixing parameter for meta learner: numeric between 0 (ridge) and 1 (lasso)
compare	experimental arguments: character vector with entries "mnorm", "spls", "mrce", "sier", "mtps", "rmtl", "gpm" and others (requires packages spls, MRCE, SiER, MTPS, RMTL or GPM)
mice	missing data imputation: logical (mice=TRUE requires package mice)
cvpred	return cross-validated predictions: logical
times	measure computation time: logical
...	further arguments passed to <code>glmnet</code> and <code>cv.glmnet</code>

Value

This function returns a matrix with q columns, including the cross-validated loss from the univariate models (base), the multivariate models (meta), and the intercept-only models (none).

Examples

```
## Not run:
n <- 50; p <- 100; q <- 3
X <- matrix(rnorm(n*p), nrow=n, ncol=p)
Y <- replicate(n=q, expr=rnorm(n=n, mean=rowSums(X[, 1:5])))
cv.joinet(Y=Y, X=X)
## End(Not run)

## Not run:
# correlated features
n <- 50; p <- 100; q <- 3
mu <- rep(0, times=p)
Sigma <- 0.90*abs(col(diag(p))-row(diag(p)))
X <- MASS::mvrnorm(n=n, mu=mu, Sigma=Sigma)
mu <- rowSums(X[, sample(seq_len(p), size=5)])
Y <- replicate(n=q, expr=rnorm(n=n, mean=mu))
#Y <- t(MASS::mvrnorm(n=q, mu=mu, Sigma=diag(n)))
cv.joinet(Y=Y, X=X)
## End(Not run)

## Not run:
# other distributions
n <- 50; p <- 100; q <- 3
X <- matrix(rnorm(n*p), nrow=n, ncol=p)
eta <- rowSums(X[, 1:5])
Y <- replicate(n=q, expr=rbinom(n=n, size=1, prob=1/(1+exp(-eta))))
cv.joinet(Y=Y, X=X, family="binomial")
Y <- replicate(n=q, expr=rpois(n=n, lambda=exp(scale(eta))))
cv.joinet(Y=Y, X=X, family="poisson")
## End(Not run)

## Not run:
# uncorrelated outcomes
```

```

n <- 50; p <- 100; q <- 3
X <- matrix(rnorm(n*p), nrow=n, ncol=p)
y <- rnorm(n=n, mean=rowSums(X[, 1:5]))
Y <- cbind(y, matrix(rnorm(n*(q-1)), nrow=n, ncol=q-1))
cv.joint(Y=Y, X=X)
## End(Not run)

## Not run:
# sparse and dense models
n <- 50; p <- 100; q <- 3
X <- matrix(rnorm(n*p), nrow=n, ncol=p)
Y <- replicate(n=q, expr=rnorm(n=n, mean=rowSums(X[, 1:5])))
set.seed(1) # fix folds
cv.joint(Y=Y, X=X, alpha.base=1) # lasso
set.seed(1)
cv.joint(Y=Y, X=X, alpha.base=0) # ridge
## End(Not run)

```

joint

Multivariate Elastic Net Regression

Description

Implements multivariate elastic net regression.

Usage

```

joint(
  Y,
  X,
  family = "gaussian",
  nfolds = 10,
  foldid = NULL,
  type.measure = "deviance",
  alpha.base = 1,
  alpha.meta = 1,
  ...
)

```

Arguments

Y	outputs: numeric matrix with n rows (samples) and q columns (variables), with positive correlation (see details)
X	inputs: numeric matrix with n rows (samples) and p columns (variables)
family	distribution: vector of length 1 or q with entries "gaussian", "binomial" or "poisson"
nfolds	number of folds

foldid	fold identifiers: vector of length n with entries between 1 and nolds; or NULL (balance)
type.measure	loss function: vector of length 1 or q with entries "deviance", "class", "mse" or "mae" (see cv.glmnet)
alpha.base	elastic net mixing parameter for base learners: numeric between 0 (ridge) and 1 (lasso)
alpha.meta	elastic net mixing parameter for meta learner: numeric between 0 (ridge) and 1 (lasso)
...	further arguments passed to glmnet

Details

correlation: The q outcomes should be positively correlated. Avoid negative correlations by changing the sign of the variable.

elastic net: alpha.base controls input-output effects, alpha.meta controls output-output effects; lasso renders sparse models (alpha= 1), ridge renders dense models (alpha= 0)

Value

This function returns an object of class `joinet`. Available methods include [predict](#), [coef](#), and [weights](#). The slots `base` and `meta` each contain q [cv.glmnet](#)-like objects.

References

Armin Rauschenberger, Enrico Glaab (2020) "Predicting correlated outcomes from molecular data" *Manuscript in preparation*.

See Also

[cv.joinet](#), [vignette](#)

Examples

```
## Not run:
n <- 50; p <- 100; q <- 3
X <- matrix(rnorm(n*p), nrow=n, ncol=p)
Y <- replicate(n=q, expr=rnorm(n=n, mean=rowSums(X[, 1:5])))
object <- joinet(Y=Y, X=X)
## End(Not run)

## Not run:
browseVignettes("joinet") # further examples
## End(Not run)
```

predict.joinet *Make Predictions*

Description

Predicts outcome from features with stacked model.

Usage

```
## S3 method for class 'joinet'
predict(object, newx, type = "response", ...)
```

Arguments

object	joinet object
newx	covariates: numeric matrix with n rows (samples) and p columns (variables)
type	character "link" or "response"
...	further arguments (not applicable)

Value

This function returns predictions from base and meta learners. The slots base and meta each contain a matrix with n rows (samples) and q columns (variables).

Examples

```
## Not run:
n <- 50; p <- 100; q <- 3
X <- matrix(rnorm(n*p), nrow=n, ncol=p)
Y <- replicate(n=q, expr=rnorm(n=n, mean=rowSums(X[,1:5])))
Y[,1] <- 1*(Y[,1]>median(Y[,1]))
object <- joinet(Y=Y, X=X, family=c("binomial", "gaussian", "gaussian"))
predict(object, newx=X)
## End(Not run)
```

weights.joinet *Extract Weights*

Description

Extracts coefficients from the meta learner, i.e. the weights for the base learners.

Usage

```
## S3 method for class 'joinet'  
weights(object, ...)
```

Arguments

```
object          joinet object  
...            further arguments (not applicable)
```

Value

This function returns a matrix with $1 + q$ rows and q columns. The first row contains the intercepts, and the other rows contain the slopes, which are the effects of the outcomes in the row on the outcomes in the column.

Examples

```
## Not run:  
n <- 50; p <- 100; q <- 3  
X <- matrix(rnorm(n*p), nrow=n, ncol=p)  
Y <- replicate(n=q, expr=rnorm(n=n, mean=rowSums(X[, 1:5])))  
object <- joinet(Y=Y, X=X)  
weights(object)  
## End(Not run)
```

Index

* **documentation**

jointet-package, [2](#)

coef, [7](#)

coef.jointet, [3](#)

cv.glmnet, [4](#), [5](#), [7](#)

cv.jointet, [4](#), [7](#)

glmnet, [5](#), [7](#)

jointet, [2](#), [3](#), [6](#), [8](#), [9](#)

jointet-package, [2](#)

predict, [7](#)

predict.jointet, [8](#)

weights, [7](#)

weights.jointet, [8](#)