Package ‘xtune’

May 24, 2019

Type Package

Title Regularized Regression with Differential Penalties Integrating
External Information

Version 0.1.0

Author Chubing Zeng

Maintainer Chubing Zeng <chubingz@usc.edu>

Description Extends standard penalized regression (Lasso and Ridge) to allow differential shrinkage based on external information with the goal of achieving a better prediction accuracy. Examples of external information include the grouping of predictors, prior knowledge of biological importance, external p-values, function annotations, etc. The choice of multiple tuning parameters is done using an Empirical Bayes approach. A majorization-minimization algorithm is employed for implementation.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports glmnet, stats, selectiveInference

Depends R (>= 2.10)

RoxygenNote 6.0.1

Suggests knitr, numDeriv, lbfgs, rmarkdown, testthat, covr

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2019-05-24 09:00:03 UTC

R topics documented:

`coef.xtune` *Extract model coefficients from fitted* `xtune` *object*

Description

`coef.xtune` extracts model coefficients from objects returned by `xtune` object.

Usage

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

Arguments

`object`	Fitted 'xtune' model object.
`...`	Not used

Details

`coef` and `predict` methods are provided as a convenience to extract coefficients and make prediction. `coef.xtune` simply extracts the estimated coefficients returned by `xtune`.

Value

Coefficients extracted from the fitted model.

See Also

`xtune`, `predict.xtune`

Examples

```
## see examples in predict.xtune()
```

`diet` *Simulated diet data to predict weight loss*

Description

The simulated `diet` data contains 100 observations, 14 predictors, and an binary outcome, weightloss. The external information Z is the nutrition fact about the dietary items. Z contains three external information variables: Calories, protein and carbohydrates.

Usage

```
data(diet)
```

Format

The `diet` object is a list containing three elements:

- DietItems: Matrix of predictors.
- weightloss: 0: no weight loss; 1: weight loss
- nutritionFact: External information of the predictors

References

S. Witte, John & Greenland, Sander & W. Haile, Robert & L. Bird, Cristy. (1994). Hierarchical Regression Analysis Applied to a Study of Multiple Dietary Exposures and Breast Cancer. Epidemiology (Cambridge, Mass.). 5. 612-21. 10.1097/00001648-199411000-00009.

See Also

`example`

Examples

```
data(diet)
X <- diet$DietItems
Y <- diet$weightloss
Z <- diet$nutritionFact
fit <- xtune(X,Y,Z)
fit$penalty.vector
```

estimateVariance *Estimate noise variance given predictor X and response Y.*

Description

estimateVariance estimate noise variance.

Usage

```
estimateVariance(X, Y, n_rep = 5)
```

Arguments

X	predictor matrix of dimension n by p.
Y	continuous outcome vector of length n.
n_rep	number of repeated estimation. Default is 10.

Details

The estimateSigma function from selectiveInference is used repeatedly to estimate noise variance.

References

Stephen Reid, Jerome Friedman, and Rob Tibshirani (2014). A study of error variance estimation in lasso regression. arXiv:1311.5274.

See Also

selectiveInference

Examples

```
## simulate some data
set.seed(9)
n = 30
p = 10
sigma.square = 1
X = matrix(rnorm(n*p),n,p)
beta = c(2,-2,1,-1,rep(0,p-4))
Y = X%*%beta + rnorm(n,0,sqrt(sigma.square))

## estimate sigma square
sigma.square.est = estimateVariance(X,Y)
sigma.square.est
```

`example` *An simulated example dataset*

Description

The simulated `example` data contains 100 observations, 200 predictors, and an continuous outcome. Z contains 3 columns, each column is indicator variable (can be viewed as the grouping of predictors).

Usage

```
data(example)
```

Format

The `example` object is a list containing three elements:

- X: A simulated 100 by 200 matrix
- Y: Continuous response vector of length 100
- Z: A 200 by 3 matrix. Z_jk indicates whether predictor X_j has external variable Z_k or not.

Examples

```
data(example)
X <- example$X
Y <- example$Y
Z <- example$Z
xtune(X,Y,Z)
```

`gene` *Simulated gene data to predict weight loss*

Description

The simulated gene data contains 50 observations, 200 predictors, and an continuous outcome, bone mineral density. The external information Z is four previous study results that identifies the biological importance of genes.

Usage

```
data(gene)
```

Format

The gene object is a list containing three elements:

- GeneExpression: Matrix of gene expression predictors.
- bonedensity: Continuous outcome variable
- PreviousStudy: Whether each gene is identified by previous study results.

See Also

`diet`

Examples

```
data(gene)
X <- gene$geneItems
Y <- gene$weightloss
Z <- gene$NuitritionFact
fit <- xtune(X,Y,Z)
fit$penalty.vector
```

`misclassification` *Calculate misclassification error*

Description

`misclassification` calculate misclassification error between predicted class and true class

Usage

```
misclassification(pred, true)
```

Arguments

`pred`	Predicted class
`true`	Actual class

Value

misclassification error

Examples

```
Y1 <- rbinom(10,1,0.5)
Y2 <- rnorm(10,1,0.5)
misclassification(Y1,Y2)
```

mse *Calculate mean square error*

Description

`mse` calculate mean square error (MSE) between prediction values and true values

Usage

```
mse(pred, true)
```

Arguments

`pred`	Prediction values vector
`true`	Actual values vector

Value

mean square error

Examples

```
Y1 <- rnorm(10,0,1)
Y2 <- rnorm(10,0,1)
mse(Y1,Y2)
```

predict.xtune *Model predictions based on fitted `xtune` object*

Description

`predict.xtune` produces predicted values fitting an xtune model to a new dataset

Usage

```
## S3 method for class 'xtune'
predict(object, newX, type = c("response", "class"),
  X = NULL, Y = NULL, ...)
```

Arguments

`object`	Fitted 'xtune' model object.
`newX`	Matrix of values at which predictions are to be made.
`type`	Type of prediction required. For "linear" models it gives the fitted values. Type "response" gives the fitted probability scores for "binary" outcome. Type "class" applies only to "binary" models, and produces the class label corresponding to the maximum probability. Note that with type = "class", it is required to supply the original X = and Y = as additional arguments to `predict()`.
`X`	Passing arguments `X=` when `type = class`
`Y`	Passing arguments `Y=` when `type = class`
`...`	Not used

Details

`coef` and `predict` methods are provided as a convenience to extract coefficients and make prediction. `predict.xtune` simply calculate the predicted value using the estimated coefficients returned by `xtune`.

Value

A vector of predictions

See Also

`xtune`, `coef.xtune`

Examples

```
## simulate data
set.seed(9)
data(example)
X <- example$X
Y <- example$Y
Z <- example$Z

## If no Z provided, perform Empirical Bayes tuning
# fit.eb <- xtune(X,Y)
## Coef and predict methods
#coef(fit.eb)
# predict(fit.eb,X)

## Differential shrinkage based on external information Z:
fit.diff <- xtune(X,Y,Z)
## Coef and predict methods
coef(fit.diff)
predict(fit.diff,X)
```

xtune *Tuning differential shrinkage parameters in penalized regression based on external information.*

Description

xtune uses an Empirical Bayes approach to integrate external information into penalized linear regression models. It fits models with differential amount of shrinkage for each regression coefficient based on external information.

Usage

```
xtune(X, Y, Z = NULL, family = c(”linear”, "binary"), sigma.square = NULL,
  method = c(”lasso”, "ridge"), message = TRUE, control = list())
```

Arguments

X Numeric design matrix of explanatory variables (n observations in rows, p predictors in columns), without an intercept. xtune includes an intercept by default.

Y Outcome vector of dimension n. Quantitative for family="linear", or family="binary" for a 0/1 binary outcome variable.

Z Numeric information matrix about the predictors (p rows, each corresponding to a predictor in X; q columns of external information about the predictors, such as prior biological importance). If Z is the grouping of predictors, it is best if user codes it as a dummy variable (i.e. each column indicating whether predictors belong to a specific group)

family Response type. "linear" for continuous outcome, "binary" for 0/1 binary outcome.

sigma.square A user-supplied noise variance estimate. Typically, this is left unspecified, and the function automatically computes an estimated sigma square values using R package selectiveinference.

method The type of regularization applied in the model. method = ’lasso’ for Lasso regression, method = ’ridge’ for Ridge regression

message Generates diagnostic message in model fitting. Default is TRUE.

control Specifies xtune control object. See xtune.control for more details.

Details

xtune has two main usages:

- The basic usage of it is to choose the tuning parameter λ in Lasso and Ridge regression using an Empirical Bayes approach, as an alternative to the widely-used cross-validation. This is done by calling xtune without specifying external information matrix Z.

- More importantly, if an external information Z about the predictors X is provided, `xtune` can allow differential shrinkage parameters for regression coefficients in penalized regression models. The idea is that Z might be informative for the effect-size of regression coefficients, therefore we can guide the penalized regression model using Z.

Please note that the number of rows in Z should match with the number of columns in X. Since each column in Z is a feature about X. See here for more details on how to specify Z.

A majorization-minimization procedure is employed to fit `xtune`.

Value

An object with S3 class `xtune` containing:

`beta.est` The fitted vector of coefficients.

`penalty.vector` The estimated penalty vector applied to each regression coefficient. Similar to the `penalty.factor` argument in glmnet.

`lambda` The estimated λ value. Note that the lambda value is calculated to reflect that the fact that penalty factors are internally rescaled to sum to nvars in glmnet. Similar to the `lambda` argument in glmnet.

`n_iter` Number of iterations used until convergence.

`method` Same as in argument above

`sigma.square` The estimated sigma square value using `estimateVariance`, if `sigma.square` is left unspecified.

`family` same as above

`likelihood` A vector containing the marginal likelihood value of the fitted model at each iteration.

Author(s)

Chubing Zeng

See Also

predict.xtune, as well as glmnet.

Examples

```
## use simulated example data
set.seed(9)
data(example)
X <- example$X
Y <- example$Y
Z <- example$Z

## Empirical Bayes tuning to estimate tuning parameter, as an alternative to cross-validation:
fit.eb <- xtune(X,Y)
fit.eb$lambda

### compare with tuning parameter choosen by cross-validation, using glmnet
```

```
## Not run:
fit.cv <- cv.glmnet(X,Y,alpha = 1)
fit.cv$lambda.min

## End(Not run)
## Differential shrinkage based on external information Z:
fit.diff <- xtune(X,Y,Z)
fit.diff$penalty.vector
```

xtune.control *Control function for xtune fitting*

Description

Control function for xtune fitting.

Usage

```
xtune.control(alpha.init = NULL, maxstep = 100, tolerance = 0.001,
  maxstep_inner = 50, tolerance_inner = 0.1, compute.likelihood = FALSE,
  verbosity = FALSE, standardize = TRUE, intercept = TRUE)
```

Arguments

alpha.init	initial values of alpha vector supplied to the algorithm. alpha values are the hyper-parameters for the double exponential prior of regression coefficients, and it controls the prior variance of regression coefficients. Default is a vector of 0 with length p.
maxstep	Maximum number of iterations. Default is 100.
tolerance	Convergence threshold. Default is 1e-4.
maxstep_inner	Maximum number of iterations for the inner loop of the majorization-minimization algorithm.
tolerance_inner	Convergence threshold for the inner loop of the majorization-minimization algorithm.
compute.likelihood	Should the function compute the marginal likelihood for hyper-parameters at each step of the update? Default is TRUE.
verbosity	Track algorithm update process? Default is FALSE.
standardize	Standardize X or not, same as the standardized option in glmnet
intercept	Should intercept(s) be fitted (default=TRUE) or set to zero (FALSE), same as the intercept option in glmnet

Index