Package 'logitr'

May 5, 2017

Type Package

Title Penalized Logistic Regression

Version 0.1.0

Description This is an R package for linear and logistic regression with optional ridge and bridge regularization penalties.

URL https://github.com/MGallow/logitr

BugReports https://github.com/MGallow/logitr/issues

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports Rcpp (>= 0.12.10), dplyr

LinkingTo Rcpp

RoxygenNote 5.0.1

R topics documented:

`gradient_IRLS_logistic` *Gradient of Logistic Regression (IRLS)*

Description

Computes the gradient of logistic regression (optional ridge regularization term). We use this to determine if the KKT conditions are satisfied. This function is to be used with the 'IRLS' function.

Usage

```
gradient_IRLS_logistic(betas, X, y, lam = 0, vec)
```

Arguments

`betas`	beta estimates (includes intercept)
`X`	matrix or data frame
`y`	response vector of 0,1
`lam`	tuning parameter for ridge regularization term
`vec`	vector to specify which coefficients will be penalized

Value

returns the gradient

Examples

```
gradient_IRLS_logistic(betas, X, y, lam = 0.1, penalty = 'ridge')
```

`gradient_linear` *Gradient of Linear Regression*

Description

Computes the gradient of linear regression (optional ridge regularization term). This function is to be used with the 'Linearr' function.

Usage

```
gradient_linear(betas, X, y, lam = 0, weights = NULL, vec)
```

Arguments

`betas`	beta estimates (includes intercept)
`X`	matrix or data frame
`y`	response vector of 0,1
`lam`	tuning parameter for ridge regularization term
`weights`	option vector of weights for weighted least squares
`vec`	vector to specify which coefficients will be penalized

Value

returns the gradient

Examples

```
gradient_linear(betas, X, y, lam = 0.1)
```

---

gradient_MM_logistic *Gradient of Logistic Regression (MM)*

Description

Computes the gradient of logistic regression (optional ridge regularization term). We use this to determine if the KKT conditions are satisfied. This function is to be used with the 'MM' function.

Usage

```
gradient_MM_logistic(betas, X, y, lam = 0, alpha = 1.5, gamma = 1, vec)
```

Arguments

betas	beta estimates (includes intercept)
X	matrix or data frame
y	response vector of 0,1
lam	tuning parameter for ridge regularization term
alpha	optional tuning parameter for bridge regularization term. Defaults to 'alpha = 1.5'
gamma	indicator function. 'gamma = 1' for ridge, 'gamma = 0' for bridge. Defaults to 'gamma = 1'
vec	vector to specify which coefficients will be penalized

Value

returns the gradient

Examples

```
gradient_MM_logistic(betas, X, y, lam = 0.1, alpha = 1.5, penalty = 'bridge')
```

IRLS *Iterative Re-Weighted Least Squares*

Description

Computes the logistic regression coefficient estimates using the iterative re-weighted least squares (IRLS) algorithm. This function is to be used with the 'logisticr' function.

Usage

```
IRLS(X, y, lam = 0, intercept = TRUE, tol = 10^(-5), maxit = 1e+05, vec)
```

Arguments

X	matrix or data frame
y	matrix or vector of response 0,1
lam	tuning parameter for regularization term
intercept	Defaults to TRUE
tol	tolerance - used to determine algorithm convergence
maxit	maximum iterations
vec	optional vector to specify which coefficients will be penalized
betas	beta estimates (includes intercept)

Value

returns beta estimates (includes intercept), total iterations, and gradients.

Examples

```
IRLS(X, y, n.list = c(rep(1, n)), lam = 0.1, alpha = 1.5)
```

---

linearr *Linear*

Description

Computes the linear regression coefficient estimates (ridge-penalization and weights, optional)

Usage

```
linearr(X, y, lam = 0, weights = NULL, intercept = TRUE, kernel = FALSE)
```

Arguments

`X`	matrix or data frame
`y`	matrix or data frame of response values
`lam`	optional tuning parameter for ridge regularization term. Defaults to 'lam = 0'
`weights`	optional vector of weights for weighted least squares
`intercept`	add column of ones if not already present. Defaults to TRUE
`kernel`	use linear kernel to compute ridge regression coefficeients. Defaults to TRUE when p » n

Value

returns the coefficient estimates

Examples

```
Weighted ridge regression
library(dplyr)
X = dplyr::select(iris, -c(Species, Sepal.Length))
y = dplyr::select(iris, Sepal.Length)
linearr(X, y, lam = 0.1, weights = rep(1:150))

Kernelized ridge regression
linearr(X, y, lam = 0.1, kernel = T)
```

`logisticr` *Logistic Regression*

Description

Computes the coefficient estimates for logistic regression. ridge regularization and bridge regularization optional.

Usage

```
logisticr(X, y, lam = 0, alpha = 1.5, penalty = "none",
  intercept = TRUE, method = "IRLS", tol = 10^(-5), maxit = 10^(5),
  vec = NULL)
```

Arguments

`X`	matrix or data frame
`y`	matrix or vector of response values 0,1
`lam`	optional tuning parameter for ridge regularization term. Defaults to 'lam = 0'
`alpha`	optional tuning parameter for bridge regularization term. Defaults to 'alpha = 1.5'
`penalty`	choose from c('none', 'ridge', 'bridge'). Defaults to 'none'
`intercept`	Defaults to TRUE

`method`	optimization algorithm. Choose from 'IRLS' or 'MM'. Defaults to 'IRLS'
`tol`	tolerance - used to determine algorithm convergence. Defaults to 10^-5
`maxit`	maximum iterations. Defaults to 10^5
`vec`	optional vector to specify which coefficients will be penalized

Value

returns beta estimates (includes intercept), total iterations, and gradients.

Examples

```
Logistic Regression
library(dplyr)
X = dplyr::select(iris, -Species)
y = dplyr::select(iris, Species)
y$Species = ifelse(y$Species == 'setosa', 1, 0)
logisticr(X, y)

ridge Logistic Regression with IRLS
logistir(X, y, lam = 0.1, penalty = 'ridge')

ridge Logistic Regression with MM
logisticr(X, y, lam = 0.1, penalty = 'ridge', method = 'MM')

bridge Logistic Regression
(Defaults to MM -- IRLS will return error)
logisticr(X, y, lam = 0.1, alpha = 1.5, penalty = 'bridge')
```

---

logitr *Logit*

Usage

```
logitr(u)
```

Arguments

`u`	some number. Ex: X returns the logit of u Computes the logit for u logit(X %*% beta)

MM *Majorize-Minimization function*

Description

This function utilizes the MM algorithm. It will be used to compute the logistic regression coefficient estimates. This function is to be used with the 'logisticr' function.

Usage

```
MM(X, y, lam = 0, alpha = 1.5, gamma = 1, intercept = TRUE,
  tol = 10^(-5), maxit = 1e+05, vec = NULL)
```

Arguments

`X`	matrix or data frame
`y`	matrix or vector of response 0,1
`lam`	optional tuning parameter for ridge regularization term. Defaults to 'lam = 0'
`alpha`	optional tuning parameter for bridge regularization term. Defaults to 'alpha = 1.5'
`gamma`	gamma indicator function. 'gamma = 1' for ridge, 'gamma = 0' for bridge. Defaults to 'gamma = 1'
`intercept`	defaults to TRUE
`tol`	tolerance - used to determine algorithm convergence
`maxit`	maximum iterations
`vec`	optional vector to specify which coefficients will be penalized

Value

returns beta estimates (includes intercept), total iterations, and gradients.

Examples

```
MM(X, y)
```

predict_linearr *Predict Linear Regression*

Description

Generates prediction for linear regression. Note that one can either input a 'linearr' object or a matrix of beta coefficients.

Usage

```
predict_linearr(object, X, y = NULL)
```

Arguments

object	'linearr' object or matrix of betas
X	matrix or data frame of (new) observations
y	optional, matrix or vector of response values

Value

predictions and loss metrics

Examples

```
fitted = linearr(X, y, lam = 0.1)
predict_linearr(fitted, X)
```

`predict_logisticr` *Predict Logistic Regression*

Description

Generates prediction for logistic regression. Note that one can either input a 'logisticr' object or a matrix of beta coefficients.

Usage

```
predict_logisticr(object, X, y = NULL)
```

Arguments

object	'logisticr' object or matrix of betas
X	matrix or data frame of (new) observations
y	optional, matrix or vector of response values 0,1

Value

predictions and loss metrics

Examples

```
fitted = logisticr(X, y, lam = 0.1, penalty = 'ridge', method = 'MM')
predict_logisticr(fitted, X)
```

Index