Package 'bigReg'

December 11, 2023

Type Package

Title Generalized Linear Models (GLM) for Large Data Sets

Version 0.1.5

Date 2023-12-09

Author Chibisi Chima-Okereke <chibisi@active-analytics.com>

Maintainer Chibisi Chima-Okereke <chibisi@active-analytics.com>

Description Allows the user to carry out GLM on very large data sets. Data can be created using the data_frame() function and appended to the object with object$append(data); data_frame and data_matrix objects are available that allow the user to store large data on disk. The data is stored as doubles in binary format and any character columns are transformed to factors and then stored as numeric (binary) data while a look-up table is stored in a separate .meta_data file in the same folder. The data is stored in blocks and GLM regression algorithm is modified and carries out a MapReduce-like algorithm to fit the model. The functions bglm(), and summary() and bglm_predict() are available for creating and post-processing of models. The library requires Armadillo installed on your system. It may not function on windows since multi-core processing is done using mclapply() which forks R on Unix/Linux type operating systems.

License GPL (>= 2)

Depends R (>= 3.2.0), Rcpp (>= 1.0.11), parallel, methods, stats, uuid (>= 0.1-2), MASS (>= 7.3-39)

LinkingTo Rcpp, RcppArmadillo (>= 0.5.200.1.0)

OS_type unix

Collate 'RcppExports.R' 'data_frame.r' 'data_matrix.r' 'family.r' 'map_reg.r'

RoxygenNote 7.2.3

NeedsCompilation yes

Repository CRAN

Date/Publication 2023-12-11 08:00:05 UTC

Encoding UTF-8

R topics documented:

`.control` *Function for creating control parameters for the GLM fit*

Description

Function for creating control parameters for the GLM fit

Usage

```
.control(epsilon = 1e-08, maxit = 25, trace = TRUE)
```

Arguments

`epsilon`	defaults to 1E-8
`maxit`	defaults 25 maximum number of iterations
`trace`	defaults to TRUE

`asInteger` *converts numeric vector to integer*

Description

converts numeric vector to integer

Usage

```
asInteger(x)
```

Arguments

`x`	numeric vector

bglm — *Function to carry out generalized linear regression on a data_frame data object*

Description

Function to carry out generalized linear regression on a data_frame data object

Usage

```
bglm(
  formula,
  family = gaussian_(),
  data,
  weights = NULL,
  offset = NULL,
  start = NULL,
  control = list(),
  etastart = NULL,
  mustart = NULL
)
```

Arguments

`formula`	formula that defines your regression model
`family`	family object from activeReg, e.g. .gaussian(), .binomial(), .poisson(), .quasipoisson(), .quasibinomial(), .Gamma(), .inverse.gaussian(), .quasi()
`data`	data_frame object containing data for linear regression
`weights`	weights for the model
`offset`	offsets for the model
`start`	starting values for the linear predictor
`control`	list of parameters for .control() function
`etastart`	starting values for the linear predictor
`mustart`	starting values for vector of means

Examples

```
require(parallel)
data("plasma", package = "bigReg")
data_dir = tempdir()
plasma1 <- plasma
plasma1 <- data_frame(plasma1, 10, path = data_dir, nCores = 1)
plasma_glm <- bglm(ESR ~ fibrinogen + globulin, data = plasma1, family = binomial_("logit"))
summary(plasma_glm)
```

`bglm_predict` *predict function for bglm object*

Description

predict function for bglm object

Usage

```
bglm_predict(
  mf = stop("mf: model frame must be supplied"),
  object = stop("object: bglm object must be supplied"),
  type = stop("type: either \"link\", \"response\", \"terms\"")
)
```

Arguments

`mf`	model frame
`object`	a bglm object
`type`	one of c("link", "response", "terms")

`binomial_` *binomial family function*

Description

binomial family function

Usage

```
binomial_(link = "logit")
```

Arguments

`link`	function character

blm *Function to carry out linear regression on a data_frame data object*

Description

Function to carry out linear regression on a data_frame data object

Usage

```
blm(
  formula = stop("formula: not supplied"),
  data = stop("data: data not supplied"),
  control = list(),
  weights = NULL,
  offset = NULL
)
```

Arguments

`formula`	formula that defines your regression model
`data`	data_frame object containing data for linear regression
`control`	list of parameters for control() function
`weights`	weights for the model
`offset`	offsets for the model

CreateFactor *creates factor from numeric vector and character vector as levels*

Description

The `CreateFactor` function creates a factor from a numeric vector and a character vector for levels

Usage

```
CreateFactor(x, levels)
```

Arguments

`x`	numeric vector containing the numeric indices of the levels
`levels`	character vector levels

`data_frame` *function to create a data_frame object*

Description

function to create a data_frame object. The data_frame object is an object that is held on disk. It is written to a folder `path` on disk where the data is written to in blocks or chunks. The data is written in binary format using a C++ function in purely numerical data and a mapping to the table is held in a ".meta_data" file in the folder. The table object accomodates numeric, factor, and character (converted to factor).

Usage

```
data_frame(
  data = stop("data must be supplied"),
  chunkSize = stop("chunkSize must be specified, a good number is 50000"),
  path = stop("path must be specified"),
  nCores = parallel::detectCores(),
  ...
)
```

Arguments

`data`	data.frame object to be converted into a data_frame object
`chunkSize`	number of rows to be used in each chunk
`path`	character to folder where the object will be created
`nCores`	the number of cores to use defaults to parallel::detectCores()
`...`	not currently used.

Details

Creates a data_frame object

Examples

```
irisA <- data_frame(iris[1:75,], 10, "irisA", nCores = 1)
irisA$append(iris[76:150,])
irisA$head()
irisA$tail(10)
irisA$delete(); rm(irisA)
```

data_matrix *function to create a data_frame object*

Description

function to create a data_matrix object. The data_matrix object is an object that is held on disk. It is written to a folder path on disk where the data is written to in blocks or chunks. The data is written in binary format using a C++ function in purely numerical data.

Usage

```
data_matrix(
  data = stop("data: matrix must be supplied"),
  chunkSize = stop("chunkSize must be specified, a good number is 50000"),
  path = stop("path must be specified"),
  nCores = parallel::detectCores(),
  ...
)
```

Arguments

data	object to be converted into a data_matrix object
chunkSize	number of rows to be used in each chunk
path	character to folder where the object will be created
nCores	the number of cores to use defaults to parallel::detectCores()
...	not used at the moment

Details

Creates a data_matrix object

family_ *family function*

Description

family function

Usage

```
family_(distr, link)
```

Arguments

distr	distr character one of "binomial", "poisson", "gaussian", "quasipoisson", "quasibinomial", "Gamma", "inverse.gaussian", "quasi"
link	function character

`Gamma_` *Gamma family function*

Description

Gamma family function

Usage

```
Gamma_(link = "inverse")
```

Arguments

`link`	function character

`gaussian_` *gaussian family function*

Description

gaussian family function

Usage

```
gaussian_(link = "identity")
```

Arguments

`link`	function character

`inverse.gaussian_` *inverse.gaussian family function*

Description

inverse.gaussian family function

Usage

```
inverse.gaussian_(link = "1/mu^2")
```

Arguments

`link`	function character

`load_data_frame` *function to load data_frame object*

Description

function to load data_frame object

Usage

```
load_data_frame(path = stop("path: to data_frame folder must be supplied"))
```

Arguments

`path`	character to folder containing object

`load_data_matrix` *function to load data_frame object*

Description

function to load data_frame object

Usage

```
load_data_matrix(path = stop("path: to data_matrix folder must be supplied"))
```

Arguments

`path`	character to folder containing object

`myIn` *finds whether x is in y*

Description

finds whether x is in y

Usage

```
myIn(x, y)
```

Arguments

`x`	item to be sought
`y`	vector to be matched against

mySeq *mySeq function to sequence integers*

Description

a function to create a sequence of integers

Usage

```
mySeq(start, end)
```

Arguments

`start`	integer from where sequence should start
`end`	integer where sequence should end

plasma *plasma data from the HSAUR package*

Description

Dataset from the HSAUR package

Usage

```
data(plasma)
```

Format

a data.frame

Details

...

Source

HSAUR package

References

HSAUR R package (HSAUR package)

Examples

```
data(plasma)
head(plasma)
```

poisson_ *poisson family function*

Description

poisson family function

Usage

```
poisson_(link = "log")
```

Arguments

link	function character

print.bglm *print function for the bglm object*

Description

print function for the bglm object

Usage

```
## S3 method for class 'bglm'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

x	bglm object to be displayed
digits	number of significant digits to use
...	not yet used

print.blm *print function for the blm object*

Description

print function for the blm object

Usage

```
## S3 method for class 'blm'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

x	blm object to be displayed
digits	number of significant digits to use
...	not yet used

print.data_frame *print function for a data_frame*

Description

print function for a data_frame

Usage

```
## S3 method for class 'data_frame'
print(x, ...)
```

Arguments

x	data_frame object to print
...	not used

print.data_matrix *print function for a data_matrix*

Description

print function for a data_matrix

Usage

```
## S3 method for class 'data_matrix'
print(x, ...)
```

Arguments

x	data_matrix object to print
...	not used

print.summary.bglm *Function to print the summary object from the bglm object*

Description

Function to print the summary object from the bglm object

Usage

```
## S3 method for class 'summary.bglm'
print(
  x,
  digits = max(3L, getOption("digits") - 3L),
  signif.stars = getOption("show.signif.stars"),
  ...
)
```

Arguments

x	summary blm object
digits	- the digits to be displayed
signif.stars	passed to printCoefmat
...	arguments passed to `printCoefmat()` function

`print.summary.blm` *Function to print the summary object from the blm object*

Description

Function to print the summary object from the blm object

Usage

```
## S3 method for class 'summary.blm'
print(
  x,
  digits = max(3L, getOption("digits") - 3L),
  signif.stars = getOption("show.signif.stars"),
  ...
)
```

Arguments

`x`	summary blm object
`digits`	- the digits to be displayed
`signif.stars`	passed to printCoefmat
`...`	arguments passed to `printCoefmat()` function

`process_bglm_block` *Function to print the summary object from the blm object*

Description

Function to print the summary object from the blm object

Usage

```
process_bglm_block(
  mf,
  formula,
  mmCall,
  family,
  offset,
  weights,
  start,
  niter,
  etastart,
  mustart
)
```

Arguments

`mf`	the data block to be processed
`formula`	the formula of for the model
`mmCall`	the call object of the model
`family`	the family object for the model
`offset`	the model offset
`weights`	the model weights
`start`	the starting coefficient estimates
`niter`	the current number of iterations
`etastart`	the start for eta
`mustart`	the start for mu

`quasibinomial_` *quasibinomial family function*

Description

quasibinomial family function

Usage

```
quasibinomial_(link = "logit")
```

Arguments

`link`	function character

`quasipoisson_` *quasipoisson family function*

Description

quasipoisson family function

Usage

```
quasipoisson_(link = "log")
```

Arguments

`link`	function character

quasi_ *quasi family function*

Description

quasi family function

Usage

```
quasi_(link = "identity", variance = "constant")
```

Arguments

`link`	function character
`variance`	choice character

readNumericVector *reads numeric vector to file*

Description

reads numeric vector to file

Usage

```
readNumericVector(size, filePath)
```

Arguments

`size`	the length of the numeric vector
`filePath`	dependent variable

read_df_block *read data frame block from file*

Description

read data frame block from file

Usage

```
read_df_block(size, filePath, df, ncol, factors, factor_indices)
```

Arguments

size	number of elements in the block
filePath	path to where the block is stored
df	an empty list having the same number of elements as columns in the table
ncol	number of columns in the dataframe block
factors	list containing factors
factor_indices	numeric vector containing the indicies that denote the factors

read_df_blocks *read multiple blocks of data frames from file*

Description

read multiple blocks of data frames from file

Usage

```
read_df_blocks(size, filePaths, df, ncols, factors, factor_indices)
```

Arguments

size	number of elements in each block
filePaths	path to where the blocks are stored
df	an empty list having the same number of elements as columns in the table
ncols	number of columns in the dataframe block
factors	list containing factors
factor_indices	numeric vector containing the indicies that denote the factors

`read_matrix_block` *read matrix block from file*

Description

read matrix block from file

Usage

```
read_matrix_block(filePath, size, ncol)
```

Arguments

`filePath`	path to file where matrix should be read from
`size`	total number of elements to be read
`ncol`	number of columns in the matrix

`read_matrix_blocks` *read matrix blocks from file*

Description

read matrix blocks from file

Usage

```
read_matrix_blocks(filePaths, size, ncols)
```

Arguments

`filePaths`	file paths from where the matrix blocks will be read
`size`	numeric vector containing the number of elements in each block
`ncols`	number of columns in the matrix

r_bind *row binding for benchmarking ...*

Description

row binding for benchmarking

Usage

```
r_bind(x, y)
```

Arguments

x	first matrix to be bound together
y	second matrix to be bound together

summary.bglm *summary function for the bglm object*

Description

summary function for the bglm object

Usage

```
## S3 method for class 'bglm'
summary(object, ...)
```

Arguments

object	bglm object to be summarized
...	not used

`summary.blm` *summary function for the blm object*

Description

summary function for the blm object

Usage

```
## S3 method for class 'blm'
summary(object, ...)
```

Arguments

`object`	blm object to be summarized
`...`	not used

`sum_bglm_block` *The reduction function for the algorithm*

Description

The reduction function for the algorithm

Usage

```
sum_bglm_block(x1, x2)
```

Arguments

`x1`	the first list object to be reduced
`x2`	the second list object to be reduced

SVD *Singular value decomposition of the aggregated list from XWXMatrix(W) functions*

Description

Singular value decomposition of the aggregated list from XWXMatrix(W) functions

Usage

```
SVD(out, epsilon)
```

Arguments

`out`	list containing requisite computed values
`epsilon`	either machine epsilon or user depermined epsilon

writeNumericVector *writes numeric vector to file*

Description

writes numeric vector to file

Usage

```
writeNumericVector(v, filePath)
```

Arguments

`v`	numeric vector
`filePath`	dependent variable

`write_numeric_vector` *writes numeric vector to file*

Description

writes numeric vector to file

Usage

```
write_numeric_vector(v, filePath)
```

Arguments

`v`	numeric vector to be written to file
`filePath`	path to file where the numeric vector should be written

`XWXMatrix` *Calculation of iterative regression components*

Description

Calculation of iterative regression components

Usage

```
XWXMatrix(X, y)
```

Arguments

`X`	design matrix
`y`	dependent variable

XWXMatrixW *Calculation of iterative regression components*

Description

Calculation of iterative regression components

Usage

```
XWXMatrixW(X, y, W)
```

Arguments

X	design matrix
y	dependent variable
W	weights

Index