Package 'biglm'

June 12, 2024

Type Package

Title Bounded Memory Linear and Generalized Linear Models

Version 0.9-3

Author Thomas Lumley

Maintainer Thomas Lumley <t.lumley@auckland.ac.nz>

Description Regression for data too large to fit in memory.

License GPL

Suggests RSQLite, RODBC

Depends DBI, methods

Enhances leaps

NeedsCompilation yes

Repository CRAN

Date/Publication 2024-06-12 08:10:01 UTC

Contents

---

 bigglm   *Bounded memory linear regression*

Description

bigglm creates a generalized linear model object that uses only p^2 memory for p variables.

Usage

```
bigglm(formula, data, family=gaussian(),...)
## S3 method for class 'data.frame'
bigglm(formula, data,...,chunksize=5000)
## S3 method for class 'function'
bigglm(formula, data, family=gaussian(),
     weights=NULL, sandwich=FALSE, maxit=8, tolerance=1e-7,
     start=NULL,quiet=FALSE,...)
## S3 method for class 'RODBC'
bigglm(formula, data, family=gaussian(),
      tablename, ..., chunksize=5000)
## S4 method for signature 'ANY,DBIConnection'
bigglm(formula, data, family=gaussian(),
tablename, ..., chunksize=5000)
## S3 method for class 'bigglm'
vcov(object,dispersion=NULL, ...)
## S3 method for class 'bigglm'
deviance(object,...)
## S3 method for class 'bigglm'
family(object,...)
## S3 method for class 'bigglm'
AIC(object,...,k=2)
```

Arguments

formula	A model formula
data	See Details below. Method dispatch is on this argument
family	A glm family object
chunksize	Size of chunks for processng the data frame
weights	A one-sided, single term formula specifying weights
sandwich	TRUE to compute the Huber/White sandwich covariance matrix (uses p^4 memory rather than p^2)
maxit	Maximum number of Fisher scoring iterations
tolerance	Tolerance for change in coefficient (as multiple of standard error)
start	Optional starting values for coefficients. If NULL, maxit should be at least 2 as some quantities will not be computed on the first iteration
object	A bigglm object
dispersion	Dispersion parameter, or NULL to estimate
tablename	For the SQLiteConnection method, the name of a SQL table, or a string specifying a join or nested select
k	penalty per parameter for AIC
quiet	When FALSE, warn if the fit did not converge
...	Additional arguments

Details

The `data` argument may be a function, a data frame, or a `SQLiteConnection` or `RODBC` connection object.

When it is a function the function must take a single argument `reset`. When this argument is `FALSE` it returns a data frame with the next chunk of data or `NULL` if no more data are available. When`reset=TRUE` it indicates that the data should be reread from the beginning by subsequent calls. The chunks need not be the same size or in the same order when the data are reread, but the same data must be provided in total. The `bigglm.data.frame` method gives an example of how such a function might be written, another is in the Examples below.

The model formula must not contain any data-dependent terms, as these will not be consistent when updated. Factors are permitted, but the levels of the factor must be the same across all data chunks (empty factor levels are ok). Offsets are allowed (since version 0.8).

The `SQLiteConnection` and `RODBC` methods loads only the variables needed for the model, not the whole table. The code in the `SQLiteConnection` method should work for other `DBI` connections, but I do not have any of these to check it with.

Value

An object of class `bigglm`

References

Algorithm AS274 Applied Statistics (1992) Vol.41, No. 2

See Also

[biglm](), glm

Examples

```
data(trees)
ff<-log(Volume)~log(Girth)+log(Height)
a <- bigglm(ff,data=trees, chunksize=10, sandwich=TRUE)
summary(a)

gg<-log(Volume)~log(Girth)+log(Height)+offset(2*log(Girth)+log(Height))
b <- bigglm(gg,data=trees, chunksize=10, sandwich=TRUE)
summary(b)

## Not run:
## requires internet access
make.data<-function(urlname, chunksize,...){
     conn<-NULL
    function(reset=FALSE){
    if(reset){
      if(!is.null(conn)) close(conn)
      conn<<-url(urlname,open="r")
    } else{
      rval<-read.table(conn, nrows=chunksize,...)
```

```
        if (nrow(rval)==0) {
             close(conn)
             conn<<-NULL
             rval<-NULL
        }
        return(rval)
     }
   }
}

airpoll<-make.data("http://faculty.washington.edu/tlumley/NO2.dat",
         chunksize=150,
         col.names=c("logno2","logcars","temp","windsp",
                     "tempgrad","winddir","hour","day"))

b<-bigglm(exp(logno2)~logcars+temp+windsp,
          data=airpoll, family=Gamma(log),
          start=c(2,0,0,0),maxit=10)
summary(b)

## End(Not run)
```

---

biglm   *Bounded memory linear regression*

Description

biglm creates a linear model object that uses only p^2 memory for p variables. It can be updated with more data using update. This allows linear regression on data sets larger than memory.

Usage

```
biglm(formula, data, weights=NULL, sandwich=FALSE)
## S3 method for class 'biglm'
update(object, moredata,...)
## S3 method for class 'biglm'
vcov(object,...)
## S3 method for class 'biglm'
coef(object,...)
## S3 method for class 'biglm'
summary(object,...)
## S3 method for class 'biglm'
AIC(object,...,k=2)
## S3 method for class 'biglm'
deviance(object,...)
```

Arguments

`formula`	A model formula
`weights`	A one-sided, single term formula specifying weights
`sandwich`	TRUE to compute the Huber/White sandwich covariance matrix (uses p^4 memory rather than p^2)
`object`	A `biglm` object
`data`	Data frame that must contain all variables in `formula` and `weights`
`moredata`	Additional data to add to the model
`...`	Additional arguments for future expansion
`k`	penalty per parameter for AIC

Details

The model formula must not contain any data-dependent terms, as these will not be consistent when updated. Factors are permitted, but the levels of the factor must be the same across all data chunks (empty factor levels are ok). Offsets are allowed (since version 0.8).

Value

An object of class `biglm`

References

Algorithm AS274 Applied Statistics (1992) Vol.41, No. 2

See Also

lm

Examples

```
data(trees)
ff<-log(Volume)~log(Girth)+log(Height)

chunk1<-trees[1:10,]
chunk2<-trees[11:20,]
chunk3<-trees[21:31,]

a <- biglm(ff,chunk1)
a <- update(a,chunk2)
a <- update(a,chunk3)

summary(a)
deviance(a)
AIC(a)
```

---

predict.bigglm   *Predictions from a biglm/bigglm*

Description

Computes fitted means and standard errors at new data values after fitting a model with `biglm` or `bigglm`.

Usage

```
## S3 method for class 'bigglm'
predict(object, newdata, type = c("link", "response"),
se.fit = FALSE, make.function = FALSE, ...)
## S3 method for class 'biglm'
predict(object, newdata=NULL,  se.fit = FALSE, make.function = FALSE, ...)
```

Arguments

object	fitted model
newdata	data frame with variables for new values
type	`link` is on the linear predictor scale, `response` is the response
se.fit	Compute standard errors?
make.function	If `TRUE` return a prediction function, see Details below
...	not used

Details

When `make.function` is `TRUE`, the return value is either a single function that computes the fitted values or a list of two functions that compute the fitted values and standard errors. The input to these functions is the design matrix, without the intercept column. This allows the relatively time-consuming calls to `model.frame()` and `model.matrix()` to be avoided.

Value

Either a vector of predicted values or a data frame with predicted values and standard errors.

Author(s)

based on code by Christophe Dutang

References

~put references to the literature/web site here ~

See Also

predict.glm,biglm,bigglm

Examples

```
example(biglm)
predict(a,newdata=trees)
f<-predict(a,make.function=TRUE)
X<- with(trees, cbind(log(Girth),log(Height)))
f(X)
```

Index