Package ‘RPEGLMEN’

January 13, 2020

Type Package

Title Gamma and Exponential Generalized Linear Models with Elastic Net Penalty

Version 1.0.1

Date 2020-01-07

Author Anthony Christidis <anthony.christidis@stat.ubc.ca>, Xin Chen <chenx26@uw.edu>, Daniel Hanson <hansondj@uw.edu>

Maintainer Anthony Christidis <anthony.christidis@stat.ubc.ca>


License GPL (>= 2)

Encoding UTF-8

LazyData true

Imports Rcpp (>= 1.0.2), RPEIF, PerformanceAnalytics

LinkingTo Rcpp, RcppEigen

RoxygenNote 7.0.2

Suggests R.rsp, testthat

NeedsCompilation yes

Biarch true

SystemRequirements C++11

VignetteBuilder R.rsp

Repository CRAN

Date/Publication 2020-01-13 09:40:06 UTC
fit.glmGammaNet

Description

glm.glmGammaNet Fit glmnet model for Gamma distributed response data.

Usage

fit.glmGammaNet(
  A,
  b,
  exponential.dist = FALSE,
  alpha.EN = 0.5,
  num.lambda = 100L,
  glm_type = 1L,
  max.iter = 100L,
  abs.tol = 1e-04,
  rel.tol = 0.01,
  normalize_grad = FALSE,
  k.fold = 5L,
  has_intercept = TRUE,
  k.fold.iter = 5L,
  min.lambda.ratio = 1e-04,
  ...
)

Arguments

A The matrix of independent variables.
b The vector of response variables.
exponential.dist Parameter to determine whether we use the Exponential distribution (TRUE) or the Gamma distribution (FALSE).
alpha.EN The coefficient of elastic net regularizer (1 means lasso).
num.lambda Size of the lambda grid.
glm_type Type of glm model, 1 is exponential, 2 is gamma (not implemented yet).
max.iter Max number of iteration for the prox grad descent optimizer.
**Value**

vector of optimal coefficient for the glm model.

**Author(s)**

Anthony-Alexander Christidis, <anthony.christidis@stat.ubc.ca>

**Examples**

```r
# Function to return the periodogram of data series
myperiodogram <- function (data, max.freq = 0.5,
                           twosided = FALSE, keep = 1)
{
  data.fft <- fft(data)
  N <- length(data)
  tmp <- Mod(data.fft[2:floor(N/2)])^2/N
  freq <- ((1:(floor(N/2) - 1))/N)
  tmp <- tmp[1:floor(length(tmp) * keep)]
  freq <- freq[1:floor(length(freq) * keep)]
  if (twosided) {
    tmp <- c(rev(tmp), tmp)
    freq <- c(-rev(freq), freq)
  }
  return(list(spec = tmp, freq = freq))
}

# Function to compute the standard error based the periodogram of
# the influence functions time series
SE.Gamma <- function(data, d = 7, alpha = 0.5, keep = 1)
{
  N <- length(data)
  # Compute the periodograms
  my.periodogram <- myperiodogram(data)
  my.freq <- my.periodogram$freq
  my.periodogram <- my.periodogram$spec
  # Remove values of frequency 0 as it does not contain information
  # about the variance
  my.freq <- my.freq[-1]
  my.periodogram <- my.periodogram[-1]
  # Implement cut-off
  nfreq <- length(my.freq)
  ```
my.freq <- my.freq[1:floor(nfreq*keep)]
my.periodogram <- my.periodogram[1:floor(nfreq*keep)]

# GLM with BFGS optimization
# Create 1, x, x^2, ..., x^d
x.mat <- rep(1,length(my.freq))
for(col.iter in 1:d){
  x.mat <- cbind(x.mat,my.freq^col.iter)
}

# Fit the Exponential or Gamma model
res <- fit.glmGammaNet(x.mat, my.periodogram, alpha.EN = alpha)
# Return the estimated variance
return(sqrt(exp(res[1])/N))

# Loading hedge fund data from PA
data(edhec, package = "PerformanceAnalytics")
colnames(edhec)

# Computing the expected shortfall for the time series of returns
# library(RPEIF)
# test.mat <- apply(edhec, 2, IF.ES)
# test.mat <- apply(test.mat, 2, as.numeric)

# Returning the standard errors from the Gamma distribution fit
# apply(test.mat, 2, SE.Gamma)

glmnet_exp

---

**Elastic Net Penalized Exponentially Distributed Response Variables**

**Description**

*glmnet_exp* Fit glmnet model for exponential distributed response data.

**Usage**

```r
glmnet_exp(A,
          b,
          alpha.EN = 0.5,
          num_lambda = 100L,
          glm_type = 1L,
          max_iter = 100L,
          abs_tol = 1e-04,
          rel_tol = 0.01,
          normalize_grad = FALSE,
          k_fold = 5L,
          has_intercept = TRUE,
          k_fold_iter = 5L,
          ...)```

Arguments

A The matrix of independent variables.
b The vector of response variables.
alpha.EN The coefficient of elastic net regularizer (1 means lasso).
num_lambda Size of the lambda grid.
glm_type Type of glm model, 1 is exponential, 2 is gamma (not implemented yet).
max_iter Max number of iteration for the prox grad descent optimizer.
abs_tol Absolute error threshold for the pgd optimizer.
rel_tol Relative error threshold for the pgd optimizer (not used for vanilla PGD).
normalize_grad Switch for whether to normalize the gradient or not.
k_fold The number of folds for cross validation.
has_intercept Parameter to determine if there is an intercept (TRUE) or not (FALSE).
k_fold_iter The number of iterations for the cross-validation.

Value

Vector of optimal coefficient for the glm model.

Author(s)

Anthony-Alexander Christidis, <anthony.christidis@stat.ubc.ca>

Examples

```r
# Function to return the periodogram of data series
myperiodogram <- function (data, max.freq = 0.5, twosided = FALSE, keep = 1){
  data.fft <- fft(data)
  N <- length(data)
  tmp <- Mod(data.fft[2:floor(N/2)])^2/N
  freq <- ((1:(floor(N/2) - 1))/N)
  tmp <- tmp[1:floor(length(tmp) * keep)]
  freq <- freq[1:floor(length(freq) * keep)]
  if (twosided) {
    tmp <- c(rev(tmp), tmp)
    freq <- c(-rev(freq), freq)
  }
  return(list(spec = tmp, freq = freq))
}

# Function to compute the standard error based the periodogram of the influence functions time series
```
SE.Exponential <- function(data, d = 7, alpha = 0.5, keep = 1){
  N <- length(data)
  # Compute the periodograms
  my.periodogram <- myperiodogram(data)
  my.freq <- my.periodogram$freq
  my.periodogram <- my.periodogram$spec
  # Remove values of frequency 0 as it does not contain information
  # about the variance
  my.freq <- my.freq[-1]
  my.periodogram <- my.periodogram[-1]
  # Implement cut-off
  nfreq <- length(my.freq)
  my.freq <- my.freq[1:floor(nfreq*keep)]
  my.periodogram <- my.periodogram[1:floor(nfreq*keep)]
  # GLM with BFGS optimization
  # Create 1, x, x^2, ..., x^d
  x.mat <- rep(1, length(my.freq))
  for(col.iter in 1:d){
    x.mat <- cbind(x.mat, my.freq^col.iter)
  }
  # Fit the Exponential model
  res <- glmnet_exp(x.mat, my.periodogram, alpha.EN = alpha)
  # Return the estimated variance
  return(sqrt(exp(res[1])/N))
}

# Loading hedge fund data from PA
data(edhec, package = "PerformanceAnalytics")
colnames(edhec)

# Computing the expected shortfall for the time series of returns
# library(RPEIF)
# test.mat <- apply(edhec, 2, IF.ES)
# test.mat <- apply(test.mat, 2, as.numeric)

# Returning the standard errors from the Exponential distribution fit
# apply(test.mat, 2, SE.Exponential)
Index

fit.glmGammaNet, 2

glmnet_exp, 4