Package 'BFI'

July 18, 2023

Type Package

Title Bayesian Federated Inference

Version 0.4.2

Date 2023-07-02

Author Hassan Pazira [aut, cre],
Marianne Jonker [aut],
Anthony Coolen [aut]

Maintainer Hassan Pazira <hassan.pazira@radboudumc.nl>

Description Bayesian Federated Inference method combines data from different (medical) centers without sharing them. In this version of the package, the user can fit models specifying Gaussian and Binomial (Logistic) families.

Encoding UTF-8

Suggests knitr,
rmarkdown

VignetteBuilder knitr

R topics documented:

`bfi` *Bayesian Federated Inference*

Description

`bfi` function is used (in central server) to estimate the parameters in the GLM and Survival models by BFI method using the aggregated results form the local centers.

Usage

```
bfi(theta_hats = NULL, A_hats, Lambda,
    L = NULL, stratified = FALSE, nuisance = 1L)
```

Arguments

theta_hats	the p- or $(p+1)$-dimensional vector corresponding to the maximum a posteriori (MAP) estimation of the parameters
A_hats	the curvature matrix around the point theta_hat.
Lambda	the matrix used as the prior of the inverse variance-covariance matrix of Gaussian distribution.
L	number of locations/centers
stratified	logical flag for fitting an intercept. The intercept is fitted if intercept=TRUE (the default) or set to zero if intercept=FALSE.
nuisance	

Details

bfi function implements .....

Value

bfi returns a list containing the following components:

theta_hat	the p- or $(p+1)$-dimensional vector of estimates obtained by the 'BFI' method if stratified=FALSE. When stratified=TRUE, in this case dimension of this vector is $L+p-1$ (for binomial family) or $2*L+p-1$ (for gaussian family). See 'Details';
A_hat	the curvature matrix obtained by the 'BFI' method if stratified=FALSE. In this case, it's a $p \times p$- or $(p+1) \times (p+1)$-dimensional matrix depending on the family used. If stratified=TRUE, it's a list of L matrices corresponding to each center. This is not the same as A_hats in 'Arguments'. See 'Details';
sd	the p- or $(p+1)$-dimensional vector of standard deviation of estimates in theta_hat if stratified=FALSE, i.e., sqrt(diag(solve(A_hat))). If stratified=TRUE, it's a list of L vectors corresponding to each center. These vectors are standard deviation of parameter estimates obtained from the matrices in A_hat, i.e., sqrt(diag(solve(A_hat[[j]]))) where j refers to a center.

Author(s)

Hassan Pazira
Maintainer: Hassan Pazira <hassan.pazira@radboudumc.nl>

References

Jonker M.A., Pazira H. and Coolen A.C.C. (2023). *Bayesian Federated Inference for Statistical Models*, *Statistics in Medicine*, Vol. 0(0), 0-0. <https://doi.org/10.48550/arXiv.2302.07677>

See Also

MAP.estimation.

Examples

```
#-------------
# y ~ Binomial
#-------------
L <- 2    # L is the number of locations

##----------------
## Local center 1:
##----------------
set.seed(1123)
n1         <- 30
p          <- 4    # p (number of coefficients) is the same for all L locations.
X1         <- data.frame(matrix(rnorm(n1 * p1), n1, p1))
#true_beta <- c(1, 2, 0, 0, 0)
eta1       <- 1 + 2 * X1[,1]  ## with an intercept b0=1, b1=2, b2=b3=...=bp=0
mu1        <- binomial()$linkinv(eta1)
y1         <- rbinom(n1, 1, mu1)
lambda     <- 0.01
# we assume the same (inverse) covariance matrix for all locations.
Lambda      <- inv.prior.cov(X1, lambda, family=binomial)
fit1       <- MAP.estimation(y1, X1, family=binomial, Lambda)
theta_hat1 <- fit1$theta_hat # beta (intercept and coefficient) estimates
A_hat1     <- fit1$A_hat

##----------------
## Local center 2:
##----------------
n2         <- 50
p          <- 4    # p is the same for all L locations
X2         <- data.frame(matrix(rnorm(n2 * p2), n2, p2))
eta2       <- 1 + 2 * X2[,1]  ## with an intercept b0=1, b1=2,b2=b3=...=bp=0
mu2        <- binomial()$linkinv(eta2)
y2         <- rbinom(n2, 1, mu2)
fit2       <- MAP.estimation(y2, X2, family=binomial, Lambda)
theta_hat2 <- fit2$theta_hat # intercept and coefficient estimates
A_hat2     <- fit2$A_hat

##--------------
## Combined data
##--------------
y              <- c(y1, y2)
X              <- rbind(X1, X2)
fit_comb       <- MAP.estimation(y, X, family=binomial, Lambda)
theta_hat_comb <- fit_comb$theta_hat # beta estimates of combined data

###-----------------------------
### Bayesian Federated Inference
###-----------------------------
A_hats <- list(A_hat1, A_hat2)
theta_hats <- list(theta_hat1, theta_hat2)

# theta (intercept and coefficient) estimates by BFI
(theta_hat_bfi <- bfi(theta_hats, A_hats, Lambda)$theta_hat)

# Curvature matrix estimate by BFI
A_bfi  <- bfi(A_hats=A_hats, Lambda=Lambda)$A_hat # == bfi(theta_hats, A_hats=A_hats, Lambda=Lambda)$A_hat
```

```
# SD of the BFI estimates
sd_bfi <- bfi(A_hats=A_hats, Lambda=Lambda)$sd # == bfi(theta_hats, A_hats=A_hats, Lambda=Lambda)$sd

### Difference between BFI estimates and estimates with combined data
theta_hat_bfi-theta_hat_comb
```

`inv.prior.cov` *Creates an inverse covariance matrix for a Gaussian prior*

Description

`inv.prior.cov` builds a matrix which can be used as the inverse of a covariance matrix for a Gaussian prior distribution, to be used in the main functions `MAP.estimation()` and `bfi()`.

Usage

```
inv.prior.cov(X, lambda = 1, family = gaussian,
              intercept = TRUE, independ = TRUE,
              set_seed = NULL)
```

Arguments

`X`	design matrix of dimension $n \times p$, where p is the number of covariates (predictors) plus intercept.
`lambda`	a user supplied lambda sequence. Length of the vector `lambda` depends on the arguments `X`, `family`, and `intercept`. If `lambda` chosen by the user is an scalar, the function `inv.prior.cov()` consider `lambda` as a vector whose all elements are equal to that scalar value. Default is `lambda=1`. If the user choose a vector of two elements as an entry value, the `inv.prior.cov()` function set `lambda` as a vector whose all elements, except the last one, are equal to the first element of the entry, and the last element is set to the last element of the entry. Used when `independ=TRUE`
`family`	a description of the error distribution and link function used to specify the model. This can be a character string naming a family function or the result of a call to a family function (see `family` for details). By default the `gaussian` family (with identity link function) is used.
`intercept`	logical flag for fitting an intercept. The intercept is fitted if `intercept=TRUE` (the default) or set to zero if `intercept=FALSE`.
`independ`	logical flag for creating the prior covariance matrix with (in)dependent variables/parameters. If `TRUE` (the default), the result is a diagonal matrix which means the parameters are independent. If `FALSE`, the off-diagonal elements of the resulted matrix are generated from the standard normal distribution `rnorm()`, and the diagonal elements are the vector `lambda`.
`set_seed`	if `independ=FALSE`, this argument (which should be an integer) ensures consistent results. If `set_seed=NULL` (the default), the results are not consistent.

Details

`inv.prior.cov` creats a matrix whose dimension depends on the arguments `X`, `family`, and `intercept`. If we assume the parameters are independent (`independ=TRUE`), the function `inv.prior.cov()` returns a diagonal matrix with the vector `lambda` as its diagonal. If `independ=FALSE`, `inv.prior.cov` returns a matrix which is equal to `z%*%t(z)` where z is a matrix such that all elements are generated by standard normal distribution.

Value

`inv.prior.cov` returns a matrix.

Author(s)

Hassan Pazira
Maintainer: Hassan Pazira <hassan.pazira@radboudumc.nl>

References

Jonker M.A., Pazira H. and Coolen A.C.C. (2023). *Bayesian Federated Inference for Statistical Models*, *Statistics in Medicine*, Vol. 0(0), 0-0. <https://doi.org/10.48550/arXiv.2302.07677>

See Also

`MAP.estimation`.

Examples

```
#-------------
# y ~ Binomial
#-------------
X      <- data.frame(matrix(rnorm(50 * 4), 50, 4))
lambda <- 0.05
# We assume the same (inverse) covariance matrix for all
# locations and consider independency of prior parameters
(Lambda <- inv.prior.cov(X, lambda, family=binomial))
# No intercept
(Lambda <- inv.prior.cov(X, lambda, family="binomial", intercept = F))

#-------------
# y ~ Gaussian
#-------------
X         <- data.frame(matrix(rnorm(50 * 3), 50, 3))
lambda    <- 0.01
sigma2e   <- 0.5
# If we consider dependency for all prior parameters:
(Lambda   <- inv.prior.cov(X, family="gaussian", independ = F, set_seed=1123))
# No intercept
(Lambda  <- inv.prior.cov(X, family="gaussian", intercept = F, independ = F, set_seed=1123))
```

`MAP.estimation` *Maximum A Posteriori estimation*

Description

`MAP.estimation` function is used (in local centers) to estimate Maximum A Posterior (MAP) of the parameters for the GLM and Survival models.

Usage

```
MAP.estimation(y, X, family = gaussian, Lambda,
               intercept = TRUE, initial = NULL,
               control = list())
```

Arguments

`y`	response vector. When the `binomial` family is used, this argument can be a vector with entries 0 (failure) or 1 (success). Alternatively, the response can be a matrix where the first column is the number of "successes" and the second column is the number of "failures".
`X`	design matrix of dimension $n \times p$, where p is the number of covariates (predictors) plus intercept.
`family`	a description of the error distribution and link function used to specify the model. This can be a character string naming a family function or the result of a call to a family function (see `family` for details). By default the `gaussian` family (with identity link function) is used.
`Lambda`	the matrix used as the prior of the inverse variance-covariance matrix of Gaussian distribution.
`intercept`	logical flag for fitting an intercept. The intercept is fitted if `intercept=TRUE` (the default) or set to zero if `intercept=FALSE`.
`initial`	a vector specifying initial values for the parameters (intercept, coefficients and/or error variance) to be optimized over. For the `gaussian` family, it should be a $p+1$-dimensional vector, and for `binomial` the length of the vector should be p, where p is the number of covariates plus intercept. Since the `'L-BFGS-B'` method is used in the algorithm, these values should always be finite. Default is a vector of zeros.
`control`	a list of control parameters. See 'Details'.

Details

`MAP.estimation` function implements ....

The argument `control` is a list that can supply any of the following components:

`maxit`**:** is the maximum number of iterations. Default is 1e2;

`factr`**:** controls the convergence of the `'L-BFGS-B'` method. Convergence occurs when the reduction in the objective is within this factor of the machine tolerance. Default is 1e7, that is a tolerance of about 1e-8;

`pgtol`**:** helps control the convergence of the `'L-BFGS-B'` method. It is a tolerance on the projected gradient in the current search direction. Default is zero, when the check is suppressed;

`trace`: is a non-negative integer. If positive, tracing information on the progress of the optimization is produced. Higher values may produce more tracing information: for the method `'L-BFGS-B'` there are six levels of tracing. To understand exactly what these do see the source code of `optim` function in the **stats** package;

`REPORT`: is the frequency of reports for the `'L-BFGS-B'` method if `'control$trace'` is positive. Default is every 10 iterations;

`lmm`: is an integer giving the number of `BFGS` updates retained in the `'L-BFGS-B'` method. Default is 5.

Value

`MAP.estimation` returns a list containing the following components:

`theta_hat`	the p- or $(p+1)$-dimensional vector corresponding to the maximum a posteriori (MAP) estimation of the parameters;
`A_hat`	the curvature matrix around the point `theta_hat`;
`sd`	the p- or $(p+1)$-dimensional vector of standard deviation of estimates in `theta_hat`, i.e., `sqrt(diag(solve(A_hat)))`;
`Lambda`	the matrix used as the prior of the inverse variance-covariance matrix;
`formula`	the formula of the model;
`n`	sample size;
`np`	the number of coefficients/regression parameters. In other words, number of predictors plus the intercept if `intercept=TRUE`, or without intercept if `intercept=FALSE`;
`value`	the value of the 'negative' loglikelihood function corresponding to `theta_hat`;
`family`	a description of the error distribution used in the model;
`convergence`	an integer value used to encode the warnings and the errors related to the algorithm used to fit the model. The values returned are: **0** algorithm has converged; **1** maximum number of iterations ('`maxit`') has been reached; **2** Warning from the 'L-BFGS-B' method. See the message after this value;
`control`	the list of control parameters used to compute the MAP estimates.

Author(s)

Hassan Pazira
Maintainer: Hassan Pazira <hassan.pazira@radboudumc.nl>

References

Jonker M.A., Pazira H. and Coolen A.C.C. (2023). *Bayesian Federated Inference for Statistical Models*, *Statistics in Medicine*, Vol. 0(0), 0-0. <https://doi.org/10.48550/arXiv.2302.07677>

See Also

`bfi` and `summary.bfi`.

Examples

```
#-------------
# y ~ Gaussian
#-------------

set.seed(11235813)
n      <- 30
p      <- 3    # number of coefficients (without intercept)
X      <- data.frame(matrix(rnorm(n * p), n, p))
eta    <- 1 + 2 * X[,1]    # with an intercept
mu     <- gaussian()$linkinv(eta)
sigma2 <- 1.5
# the true theta is c(1, 2, 0, 0, sigma2)
y      <- rnorm(n, mu, sd=sqrt(sigma2))
lambda <- 0.01
# inverse of covariance matrix:
Lambda <- inv.prior.cov(X, lambda=c(lambda,sigma2), family=gaussian)

# MAP estimates of the parameters of interest (including 'intercept') and curvature matrix
(fit <- MAP.estimation(y, X, family=gaussian, Lambda))
class(fit)

# MAP estimates without 'intercept'
Lambda <- inv.prior.cov(X, lambda=c(lambda,sigma2), family=gaussian,  intercept = F)
(fit1 <- MAP.estimation(y, X, family=gaussian, Lambda, intercept = F))
```

`Nurses` *Nurses' stress in different hospitals*

Description

This data set contains three-level simulated data from a hypothetical study on stress in hospitals. The data are from nurses working in wards nested within hospitals. It is a cluster-randomized experiment. In each of 25 hospitals, four wards are selected and randomly assigned to an experimental and a control condition. In the experimental condition, a training program is offered to all nurses to cope with job-related stress. After the program is completed, a sample of about 10 nurses from each ward is given a test that measures job-related stress. Additional variables are: nurse age (years), nurse experience (years), nurse gender (`0` = `male`, `1` = `female`), type of ward (`0` = `general care`, `1` = `special care`), and hospital size (`0` = `small`, `1` = `medium`, `2` = `large`).

Usage

```
data(Nurses)
```

References

Hox, J., Moerbeek, M., and van de Schoot, R. (2010). *Multilevel Analysis: Techniques and Applications*, Second Edition (2nd ed.). *Routledge*. <https://doi.org/10.4324/9780203852279>

summary.bfi *Summarizing BFI Fits*

Description

Summary method for an object with class 'bfi' created by the MAP.estimation function.

Usage

```
## S3 method for class 'bfi'
summary(object, curmat = FALSE,
        digits = max(3, getOption("digits") - 3))
```

Arguments

object	fitted bfi object.
curmat	logical; if TRUE, the curvature matrix around the estimated parameters is returned and printed. Default is FALSE.
digits	significant digits in printout.

Details

summary.bfi gives information about the MAP estimates of parameters of the model. It can be used for bfi objects built by the MAP.estimation function.

The output of the summary method shows the details of the model, i.e. formula, family and link function used to specify the generalized linear model, followed by information about the estimates, standard deviations and confidence intervals. Information about the log-likelihood and convergence status are also provided.

By default, summary.bfi function does not return the curvature matrix, but the user can use curmat=TRUE to print it.

Value

summary.bfi returns an object of class summary.bfi, a list with the following components:

theta_hat	the component from object. The last element of this vector is the estimate of the dispersion parameter (sigma2). See the MAP.estimation function.
A_hat	the component from object. See the MAP.estimation function.
sd	the component from object. The last element of this vector is the square root of the estimated dispersion. See the MAP.estimation function.
Lambda	the component from object. See the MAP.estimation function.
formula	the component from object. See the MAP.estimation function.
n	the component from object. See the MAP.estimation function.
np	the component from object. See the MAP.estimation function.
value	the component from object. See the MAP.estimation function.
family	the component from object. See the MAP.estimation function.
convergence	the component from object. See the MAP.estimation function.

control	the component from object. See the MAP.estimation function.
logLik	the value of the loglikelihood function corresponding to estimates (theta_hat). This is the minus of the value component.
link	the link function. By default the gaussian family with identity link function and the binomial family with logit link function are used.
dispersion	the estimated variance of the random error, i.e., sigma2. The dispersion is taken as 1 for the binomial family.
se	the standard error. se is calculated by dividing sd (standard deviation) by the square root of the sample size.
CI	a 95% confidence interval.

Author(s)

Hassan Pazira
Maintainer: Hassan Pazira <hassan.pazira@radboudumc.nl>

See Also

MAP.estimation and bfi functions.

Examples

```
#-------------
# y ~ Gaussian
#-------------

set.seed(1123581)
n       <- 30
p       <- 4
X       <- data.frame(matrix(rnorm(n * p), n, p))
b       <- 1:2
eta     <- b[1] + X[, 1] * b[2]
mu      <- gaussian()$linkinv(eta)
sigma2e <- 0.5
y       <- rnorm(n, mu, sd=sqrt(sigma2e))
lambda  <- 0.1
Lambda  <- inv.prior.cov(X, lambda=c(lambda,sigma2e), family=gaussian)
fit     <- MAP.estimation(y, X, family=gaussian, Lambda)
class(fit)

summary(fit)
sumfit  <- summary(fit, curmat = T)
sumfit$logLik
sumfit$dispersion
sumfit$CI
class(sumfit)
```

trauma *Trauma patients from different hospitals*

Description

This data set consists of data of 371 trauma patients from three hospitals. The binary variable `mortality` is used as an outcome, and variables `age`, `sex`, the Injury Severity Score (`ISS`, ranging from 1 (low) to 75 (high)) and the Glasgow Coma Scale (`GCS`, which expresses the level of consciousness, ranging from 3 (low) to 15 (high)) are used as covariates. The data originate from multiple hospitals which can be categorised in three groups as: peripheral hospital without a neuro-surgical unit (`Status = 1`), peripheral hospital with a neuro-surgical unit (`Status = 2`), and academic medical centre (`Status = 3`).

Usage

```
data(trauma)
```

References

Jonker M.A., Pazira H. and Coolen A.C.C. (2023). *Bayesian Federated Inference for Statistical Models*, *Statistics in Medicine*, Vol. 0(0), 0-0. <https://doi.org/10.48550/arXiv.2302.07677>

Draaisma J.M.Th, de Haan A.F.J., Goris R.J.A. (1989). *Preventable Trauma Deaths in the Netherlands - A prospective Multicentre Study*, *The journal of Trauma*, Vol. 29(11), 1552-1557.

Index