Package 'LongituRF'

August 31, 2020

Title Random Forests for Longitudinal Data

Version 0.9

Description Random forests are a statistical learning method widely used in many areas of scientific research essentially for its ability to learn complex relationships between input and output variables and also its capacity to handle high-dimensional data. However, current random forests approaches are not flexible enough to handle longitudinal data. In this package, we propose a general approach of random forests for high-dimensional longitudinal data. It includes a flexible stochastic model which allows the covariance structure to vary over time. Furthermore, we introduce a new method which takes intra-individual covariance into consideration to build random forests. The method is fully detailled in Capitaine et.al. (2020) <doi:10.1177/0962280220946080> Random forests for high-dimensional longitudinal data.

License GPL-2

Imports stats, randomForest, rpart, mvtnorm, latex2exp

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Suggests testthat

NeedsCompilation no

Author Louis Capitaine [aut, cre] (<https://orcid.org/0000-0001-6800-2342>)

Maintainer Louis Capitaine <Louis.capitaine@u-bordeaux.fr>

Repository CRAN

Date/Publication 2020-08-31 09:10:07 UTC

R topics documented:

`DataLongGenerator` *Longitudinal data generator*

Description

Simulate longitudinal data according to the semi-parametric stochastic mixed-effects model given by:

$$Y_i(t) = f(X_i(t)) + Z_i(t)\beta_i + \omega_i(t) + \epsilon_i$$

with $Y_i(t)$ the output at time t for the ith individual; $X_i(t)$ the input predictors (fixed effects) at time t for the ith individual; $Z_i(t)$ are the random effects at time t for the ith individual; $\omega_i(t)$ is a Brownian motion with volatility $\gamma^2 = 0.8$ at time t for the ith individual; ϵ_i is the residual error with variance $\sigma^2 = 0.5$. The data are simulated according to the simulations in low dimensional in the low dimensional scheme of the paper <doi:10.1177/0962280220946080>

Usage

```
DataLongGenerator(n = 50, p = 6, G = 6)
```

Arguments

`n`	[numeric]: Number of individuals. The default value is `n=50`.
`p`	[numeric]: Number of predictors. The default value is `p=6`.
`G`	[numeric]: Number of groups of predictors with temporal behavior, generates `p-G` input variables with no temporal behavior.

Value

a list of the following elements:

- `Y:` vector of the output trajectories.
- `X :` matrix of the fixed-effects predictors.
- `Z:` matrix of the random-effects predictors.
- `id:` vector of the identifiers for each individual.
- `time:` vector the the time measurements for each individual.

Examples

```
oldpar <- par()
oldopt <- options()
data <- DataLongGenerator(n=17, p=6,G=6) # Generate the data
# Let's see the output :
w <- which(data$id==1)
plot(data$time[w],data$Y[w],type="l",ylim=c(min(data$Y),max(data$Y)), col="grey")
for (i in unique(data$id)){
  w <- which(data$id==i)
```

```
  lines(data$time[w],data$Y[w], col='grey')
}
# Let's see the fixed effects predictors:
par(mfrow=c(2,3), mar=c(2,3,3,2))
for (i in 1:ncol(data$X)){
  w <- which(data$id==1)
  plot(data$time[w],data$X[w,i], col="grey",ylim=c(min(data$X[,i]),
 max(data$X[,i])),xlim=c(1,max(data$time)),main=latex2exp::TeX(paste0("$X^{(",i,")}$")))
  for (k in unique(data$id)){
    w <- which(data$id==k)
    lines(data$time[w],data$X[w,i], col="grey")
  }
}
par(oldpar)
options(oldopt)
```

MERF *(S)MERF algorithm*

Description

(S)MERF is an adaptation of the random forest regression method to longitudinal data introduced by Hajjem et. al. (2014) <doi:10.1080/00949655.2012.741599>. The model has been improved by Capitaine et. al. (2020) <doi:10.1177/0962280220946080> with the addition of a stochastic process. The algorithm will estimate the parameters of the following semi-parametric stochastic mixed-effects model:

$$Y_i(t) = f(X_i(t)) + Z_i(t)\beta_i + \omega_i(t) + \epsilon_i$$

with $Y_i(t)$ the output at time t for the ith individual; $X_i(t)$ the input predictors (fixed effects) at time t for the ith individual; $Z_i(t)$ are the random effects at time t for the ith individual; $\omega_i(t)$ is the stochastic process at time t for the ith individual which model the serial correlations of the output measurements; ϵ_i is the residual error.

Usage

```
MERF(
  X,
  Y,
  id,
  Z,
  iter = 100,
  mtry = ceiling(ncol(X)/3),
  ntree = 500,
  time,
  sto,
  delta = 0.001
)
```

Arguments

X	[matrix]: A Nxp matrix containing the p predictors of the fixed effects, column codes for a predictor.
Y	[vector]: A vector containing the output trajectories.
id	[vector]: Is the vector of the identifiers for the different trajectories.
Z	[matrix]: A Nxq matrix containing the q predictor of the random effects.
iter	[numeric]: Maximal number of iterations of the algorithm. The default is set to iter=100
mtry	[numeric]: Number of variables randomly sampled as candidates at each split. The default value is p/3.
ntree	[numeric]: Number of trees to grow. This should not be set to too small a number, to ensure that every input row gets predicted at least a few times. The default value is ntree=500.
time	[vector]: Is the vector of the measurement times associated with the trajectories in Y,Z and X.
sto	[character]: Defines the covariance function of the stochastic process, can be either "none" for no stochastic process, "BM" for Brownian motion, OrnUhl for standard Ornstein-Uhlenbeck process, BBridge for Brownian Bridge, fbm for Fractional Brownian motion; can also be a function defined by the user.
delta	[numeric]: The algorithm stops when the difference in log likelihood between two iterations is smaller than delta. The default value is set to O.O01

Value

A fitted (S)MERF model which is a list of the following elements:

- forest: Random forest obtained at the last iteration.
- random_effects : Predictions of random effects for different trajectories.
- id_btilde: Identifiers of individuals associated with the predictions random_effects.
- var_random_effects: Estimation of the variance covariance matrix of random effects.
- sigma_sto: Estimation of the volatility parameter of the stochastic process.
- sigma: Estimation of the residual variance parameter.
- time: The vector of the measurement times associated with the trajectories in Y,Z and X.
- sto: Stochastic process used in the model.
- Vraisemblance: Log-likelihood of the different iterations.
- id: Vector of the identifiers for the different trajectories.
- OOB: OOB error of the fitted random forest at each iteration.

Examples

```
set.seed(123)
data <- DataLongGenerator(n=20) # Generate the data composed by n=20 individuals.
# Train a SMERF model on the generated data. Should take ~ 50 seconds
# The data are generated with a Brownian motion,
# so we use the parameter sto="BM" to specify a Brownian motion as stochastic process
smerf <- MERF(X=data$X,Y=data$Y,Z=data$Z,id=data$id,time=data$time,mtry=2,ntree=500,sto="BM")
smerf$forest # is the fitted random forest (obtained at the last iteration).
smerf$random_effects # are the predicted random effects for each individual.
smerf$omega # are the predicted stochastic processes.
plot(smerf$Vraisemblance) # evolution of the log-likelihood.
smerf$OOB # OOB error at each iteration.
```

MERT *(S)MERT algorithm*

Description

(S)MERT is an adaptation of the random forest regression method to longitudinal data introduced by Hajjem et. al. (2011) <doi:10.1016/j.spl.2010.12.003>. The model has been improved by Capitaine et. al. (2020) <doi:10.1177/0962280220946080> with the addition of a stochastic process. The algorithm will estimate the parameters of the following semi-parametric stochastic mixed-effects model:

$$Y_i(t) = f(X_i(t)) + Z_i(t)\beta_i + \omega_i(t) + \epsilon_i$$

with $Y_i(t)$ the output at time t for the ith individual; $X_i(t)$ the input predictors (fixed effects) at time t for the ith individual; $Z_i(t)$ are the random effects at time t for the ith individual; $\omega_i(t)$ is the stochastic process at time t for the ith individual which model the serial correlations of the output measurements; ϵ_i is the residual error.

Usage

```
MERT(X, Y, id, Z, iter = 100, time, sto, delta = 0.001)
```

Arguments

X	[matrix]: A Nxp matrix containing the p predictors of the fixed effects, column codes for a predictor.
Y	[vector]: A vector containing the output trajectories.
id	[vector]: Is the vector of the identifiers for the different trajectories.
Z	[matrix]: A Nxq matrix containing the q predictor of the random effects.
iter	[numeric]: Maximal number of iterations of the algorithm. The default is set to iter=100
time	[vector]: Is the vector of the measurement times associated with the trajectories in Y,Z and X.

sto	[character]: Defines the covariance function of the stochastic process, can be either "none" for no stochastic process, "BM" for Brownian motion, OrnUhl for standard Ornstein-Uhlenbeck process, BBridge for Brownian Bridge, fbm for Fractional Brownian motion; can also be a function defined by the user.
delta	[numeric]: The algorithm stops when the difference in log likelihood between two iterations is smaller than delta. The default value is set to O.O01

Value

A fitted (S)MERF model which is a list of the following elements:

- forest: Tree obtained at the last iteration.
- random_effects : Predictions of random effects for different trajectories.
- id_btilde: Identifiers of individuals associated with the predictions random_effects.
- var_random_effects: Estimation of the variance covariance matrix of random effects.
- sigma_sto: Estimation of the volatility parameter of the stochastic process.
- sigma: Estimation of the residual variance parameter.
- time: The vector of the measurement times associated with the trajectories in Y,Z and X.
- sto: Stochastic process used in the model.
- Vraisemblance: Log-likelihood of the different iterations.
- id: Vector of the identifiers for the different trajectories.

Examples

```
set.seed(123)
data <- DataLongGenerator(n=20) # Generate the data composed by n=20 individuals.
# Train a SMERF model on the generated data. Should take ~ 50 secondes
# The data are generated with a Brownian motion,
# so we use the parameter sto="BM" to specify a Brownian motion as stochastic process
smert <- MERF(X=data$X,Y=data$Y,Z=data$Z,id=data$id,time=data$time,sto="BM")
smert$forest # is the fitted random forest (obtained at the last iteration).
smert$random_effects # are the predicted random effects for each individual.
smert$omega # are the predicted stochastic processes.
plot(smert$Vraisemblance) #evolution of the log-likelihood.
```

---

predict.longituRF *Predict with longitudinal trees and random forests.*

Description

Predict with longitudinal trees and random forests.

Usage

```
## S3 method for class 'longituRF'
predict(object, X, Z, id, time, ...)
```

Arguments

object	: a longituRF output of (S)MERF; (S)REEMforest; (S)MERT or (S)REEMtree function.
X	[matrix]: matrix of the fixed effects for the new observations to be predicted.
Z	[matrix]: matrix of the random effects for the new observations to be predicted.
id	[vector]: vector of the identifiers of the new observations to be predicted.
time	[vector]: vector of the time measurements of the new observations to be predicted.
...	: low levels arguments.

Value

vector of the predicted output for the new observations.

Examples

```
set.seed(123)
data <- DataLongGenerator(n=20) # Generate the data composed by n=20 individuals.
REEMF <- REEMforest(X=data$X,Y=data$Y,Z=data$Z,id=data$id,time=data$time,mtry=2,ntree=500,sto="BM")
# Then we predict on the learning sample :
pred.REEMF <- predict(REEMF, X=data$X,Z=data$Z,id=data$id, time=data$time)
# Let's have a look at the predictions
# the predictions are in red while the real output trajectories are in blue:
par(mfrow=c(4,5),mar=c(2,2,2,2))
for (i in unique(data$id)){
  w <- which(data$id==i)
  plot(data$time[w],data$Y[w],type="l",col="blue")
  lines(data$time[w],pred.REEMF[w], col="red")
}
# Train error :
mean((pred.REEMF-data$Y)^2)

# The same function can be used with a fitted SMERF model:
smerf <-MERF(X=data$X,Y=data$Y,Z=data$Z,id=data$id,time=data$time,mtry=2,ntree=500,sto="BM")
pred.smerf <- predict(smerf, X=data$X,Z=data$Z,id=data$id, time=data$time)
# Train error :
mean((pred.smerf-data$Y)^2)
# This function can be used even on a MERF model (when no stochastic process is specified)
merf <-MERF(X=data$X,Y=data$Y,Z=data$Z,id=data$id,time=data$time,mtry=2,ntree=500,sto="none")
pred.merf <- predict(merf, X=data$X,Z=data$Z,id=data$id, time=data$time)
# Train error :
mean((pred.merf-data$Y)^2)
```

REEMforest *(S)REEMforest algorithm*

Description

(S)REEMforest algorithm

Usage

```
REEMforest(
  X,
  Y,
  id,
  Z,
  iter = 100,
  mtry,
  ntree = 500,
  time,
  sto,
  delta = 0.001
)
```

Arguments

X	[matrix]: A Nxp matrix containing the p predictors of the fixed effects, column codes for a predictor.
Y	[vector]: A vector containing the output trajectories.
id	[vector]: Is the vector of the identifiers for the different trajectories.
Z	[matrix]: A Nxq matrix containing the q predictor of the random effects.
iter	[numeric]: Maximal number of iterations of the algorithm. The default is set to iter=100
mtry	[numeric]: Number of variables randomly sampled as candidates at each split. The default value is p/3.
ntree	[numeric]: Number of trees to grow. This should not be set to too small a number, to ensure that every input row gets predicted at least a few times. The default value is ntree=500.
time	[time]: Is the vector of the measurement times associated with the trajectories in Y,Z and X.
sto	[character]: Defines the covariance function of the stochastic process, can be either "none" for no stochastic process, "BM" for Brownian motion, OrnUhl for standard Ornstein-Uhlenbeck process, BBridge for Brownian Bridge, fbm for Fractional Brownian motion; can also be a function defined by the user.
delta	[numeric]: The algorithm stops when the difference in log likelihood between two iterations is smaller than delta. The default value is set to O.O01

Details

(S)REEMforest is an adaptation of the random forest regression method to longitudinal data introduced by Capitaine et. al. (2020) <doi:10.1177/0962280220946080>. The algorithm will estimate the parameters of the following semi-parametric stochastic mixed-effects model:

$$Y_i(t) = f(X_i(t)) + Z_i(t)\beta_i + \omega_i(t) + \epsilon_i$$

with $Y_i(t)$ the output at time t for the ith individual; $X_i(t)$ the input predictors (fixed effects) at time t for the ith individual; $Z_i(t)$ are the random effects at time t for the ith individual; $\omega_i(t)$ is the stochastic process at time t for the ith individual which model the serial correlations of the output measurements; ϵ_i is the residual error.

Value

A fitted (S)REEMforest model which is a list of the following elements:

- `forest`: Random forest obtained at the last iteration.
- `random_effects` : Predictions of random effects for different trajectories.
- `id_btilde`: Identifiers of individuals associated with the predictions `random_effects`.
- `var_random_effects`: Estimation of the variance covariance matrix of random effects.
- `sigma_sto`: Estimation of the volatility parameter of the stochastic process.
- `sigma`: Estimation of the residual variance parameter.
- `time`: The vector of the measurement times associated with the trajectories in `Y`,`Z` and `X`.
- `sto`: Stochastic process used in the model.
- `Vraisemblance`: Log-likelihood of the different iterations.
- `id`: Vector of the identifiers for the different trajectories.
- `OOB`: OOB error of the fitted random forest at each iteration.

Examples

```
set.seed(123)
data <- DataLongGenerator(n=20) # Generate the data composed by n=20 individuals.
# Train a SREEMforest model on the generated data. Should take ~ 50 secondes
# The data are generated with a Brownian motion
#  so we use the parameter sto="BM" to specify a Brownian motion as stochastic process
SREEMF <- REEMforest(X=data$X,Y=data$Y,Z=data$Z,id=data$id,time=data$time,mtry=2,ntree=500,sto="BM")
SREEMF$forest # is the fitted random forest (obtained at the last iteration).
SREEMF$random_effects # are the predicted random effects for each individual.
SREEMF$omega # are the predicted stochastic processes.
plot(SREEMF$Vraisemblance) #evolution of the log-likelihood.
SREEMF$OOB # OOB error at each iteration.
```

REEMtree *(S)REEMtree algorithm*

Description

(S)REEMtree is an adaptation of the random forest regression method to longitudinal data introduced by Sela and Simonoff. (2012) <doi:10.1007/s10994-011-5258-3>. The algorithm will estimate the parameters of the following semi-parametric stochastic mixed-effects model:

$$Y_i(t) = f(X_i(t)) + Z_i(t)\beta_i + \omega_i(t) + \epsilon_i$$

with $Y_i(t)$ the output at time t for the ith individual; $X_i(t)$ the input predictors (fixed effects) at time t for the ith individual; $Z_i(t)$ are the random effects at time t for the ith individual; $\omega_i(t)$ is the stochastic process at time t for the ith individual which model the serial correlations of the output measurements; ϵ_i is the residual error.

Usage

```
REEMtree(X, Y, id, Z, iter = 10, time, sto, delta = 0.001)
```

Arguments

`X`	[matrix]: A `Nxp` matrix containing the `p` predictors of the fixed effects, column codes for a predictor.
`Y`	[vector]: A vector containing the output trajectories.
`id`	[vector]: Is the vector of the identifiers for the different trajectories.
`Z`	[matrix]: A `Nxq` matrix containing the `q` predictor of the random effects.
`iter`	[numeric]: Maximal number of iterations of the algorithm. The default is set to `iter=100`
`time`	[vector]: Is the vector of the measurement times associated with the trajectories in `Y`,`Z` and `X`.
`sto`	[character]: Defines the covariance function of the stochastic process, can be either `"none"` for no stochastic process, `"BM"` for Brownian motion, `OrnUhl` for standard Ornstein-Uhlenbeck process, `BBridge` for Brownian Bridge, `fbm` for Fractional Brownian motion; can also be a function defined by the user.
`delta`	[numeric]: The algorithm stops when the difference in log likelihood between two iterations is smaller than `delta`. The default value is set to O.O01

Value

A fitted (S)MERF model which is a list of the following elements:

- `forest`: Tree obtained at the last iteration.
- `random_effects` : Predictions of random effects for different trajectories.
- `id_btilde`: Identifiers of individuals associated with the predictions `random_effects`.

- `var_random_effects`: Estimation of the variance covariance matrix of random effects.
- `sigma_sto`: Estimation of the volatility parameter of the stochastic process.
- `sigma`: Estimation of the residual variance parameter.
- `time`: The vector of the measurement times associated with the trajectories in `Y`,`Z` and `X`.
- `sto`: Stochastic process used in the model.
- `Vraisemblance`: Log-likelihood of the different iterations.
- `id`: Vector of the identifiers for the different trajectories.

Examples

```
set.seed(123)
data <- DataLongGenerator(n=20) # Generate the data composed by n=20 individuals.
# Train a SREEMtree model on the generated data.
# The data are generated with a Brownian motion,
# so we use the parameter sto="BM" to specify a Brownian motion as stochastic process
X.fixed.effects <- as.data.frame(data$X)
sreemt <- REEMtree(X=X.fixed.effects,Y=data$Y,Z=data$Z,id=data$id,time=data$time,
sto="BM", delta=0.0001)
sreemt$forest # is the fitted random forest (obtained at the last iteration).
sreemt$random_effects # are the predicted random effects for each individual.
sreemt$omega # are the predicted stochastic processes.
plot(sreemt$Vraisemblance) #evolution of the log-likelihood.
```

Index