Package ‘misaem’

January 15, 2019

Title Logistic Regression with Missing Covariates

Version 0.9.1

Date 2019-01-15

Description Estimate parameters of logistic regression with missing data and perform model selection, using algorithm Stochastic Approximation EM.

Depends R (>= 3.4.0)

Encoding UTF-8

License GPL-3

URL https://github.com/wjiang94/misaem.git

Imports mvtnorm, stats, MASS

Suggests knitr, rmarkdown

LazyData true

VignetteBuilder knitr

RoxygenNote 6.1.0

NeedsCompilation no

Author Wei Jiang [aut, cre]

Maintainer Wei Jiang <wei.jiang@polytechnique.edu>

Repository CRAN

Date/Publication 2019-01-15 12:40:03 UTC

R topics documented:

`combinations` *combinations*

Description

Given all the possible patterns of missingness.

Usage

```
combinations(p)
```

Arguments

`p`	Dimension of covariates.

Value

A matrix containing all the possible missing patterns. Each row indicates a pattern of missingness. "1" means "observed", 0 means "missing".

Examples

```
comb = combinations(5)
```

`likelihood_saem` *likelihood_saem*

Description

Used in main function miss.saem. Caculate the observed log-likelihood for logistic regression model with missing data, using Monte Carlo version of Louis formula.

Usage

```
likelihood_saem(beta, mu, Sigma, Y, X.obs,
  rindic = as.matrix(is.na(X.obs)),
  whichcolXmissing = (1:ncol(rindic))[apply(rindic, 2, sum) > 0],
  mc.size = 2)
```

Arguments

`beta` Estimated parameter of logistic regression model.

`mu` Estimated parameter μ.

`Sigma` Estimated parameter Σ.

`Y` Response vector $N \times 1$

`X.obs` Design matrix with missingness $N \times p$

`rindic` Missing pattern of X.obs. If a component in X.obs is missing, the corresponding position in rindic is 1; else 0.

`whichcolXmissing` The column index in covariate containing at least one missing observation.

`mc.size` Monte Carlo sampling size.

Value

Observed log-likelihood.

Examples

```
# Generate dataset
N <- 50  # number of subjects
p <- 3     # number of explanatory variables
mu.star <- rep(0,p)  # mean of the explanatory variables
Sigma.star <- diag(rep(1,p)) # covariance
beta.star <- c(1, 1,  0) # coefficients
beta0.star <- 0 # intercept
beta.true = c(beta0.star,beta.star)
X.complete <- matrix(rnorm(N*p), nrow=N)%*%chol(Sigma.star) +
              matrix(rep(mu.star,N), nrow=N, byrow = TRUE)
p1 <- 1/(1+exp(-X.complete%*%beta.star-beta0.star))
y <- as.numeric(runif(N)<p1)
# Generate missingness
p.miss <- 0.10
patterns <- runif(N*p)<p.miss #missing completely at random
X.obs <- X.complete
X.obs[patterns] <- NA

# Observed log-likelihood
ll_obs = likelihood_saem(beta.true,mu.star,Sigma.star,y,X.obs)
```

log_reg *log_reg*

Description

Caculate the likelihood or log-likelihood for one observation of logistic regression model .

Usage

```
log_reg(y, x, beta, iflog = TRUE)
```

Arguments

`y`	Response value (0 or 1).
`x`	Covariate vector of dimension $p \times 1$.
`beta`	Estimated parameter of logistic regression model.
`iflog`	If TRUE, log_reg calculate the log-likelihood; else likelihood.

Value

Likelihood or log-likelihood.

Examples

```
res = log_reg(1,c(1,2,3),c(1,-1,1))
```

louis_lr_saem *louis_lr_saem*

Description

Used in main function miss.saem. Caculate the variance of estimated parameters for logistic regression model with missing data, using Monte Carlo version of Louis formula.

Usage

```
louis_lr_saem(beta, mu, Sigma, Y, X.obs, pos_var = 1:ncol(X.obs),
  rindic = as.matrix(is.na(X.obs)),
  whichcolXmissing = (1:ncol(rindic))[apply(rindic, 2, sum) > 0],
  mc.size = 2)
```

Arguments

`beta`	Estimated parameter of logistic regression model.
`mu`	Estimated parameter μ.
`Sigma`	Estimated parameter Σ.
`Y`	Response vector $N \times 1$
`X.obs`	Design matrix with missingness $N \times p$
`pos_var`	Index of selected covariates.
`rindic`	Missing pattern of X.obs. If a component in X.obs is missing, the corresponding position in rindic is 1; else 0.
`whichcolXmissing`	The column index in covariate containing at least one missing observation.
`mc.size`	Monte Carlo sampling size.

Value

Variance of estimated β.

Examples

```
# Generate dataset
N <- 50  # number of subjects
p <- 3     # number of explanatory variables
mu.star <- rep(0,p)  # mean of the explanatory variables
Sigma.star <- diag(rep(1,p)) # covariance
beta.star <- c(1, 1,  0) # coefficients
beta0.star <- 0 # intercept
beta.true = c(beta0.star,beta.star)
X.complete <- matrix(rnorm(N*p), nrow=N)%*%chol(Sigma.star) +
              matrix(rep(mu.star,N), nrow=N, byrow = TRUE)
p1 <- 1/(1+exp(-X.complete%*%beta.star-beta0.star))
y <- as.numeric(runif(N)<p1)
# Generate missingness
p.miss <- 0.10
patterns <- runif(N*p)<p.miss #missing completely at random
X.obs <- X.complete
X.obs[patterns] <- NA

# Louis formula to obtain variance of estimates
V_obs = louis_lr_saem(beta.true,mu.star,Sigma.star,y,X.obs)
```

---

miss.saem *miss.saem*

Description

This function uses algorithm SAEM to fit the logistic regression model with missing data.

Usage

```
miss.saem(X.obs, y, pos_var = 1:ncol(X.obs), maxruns = 500,
  tol_em = 1e-07, nmcmc = 2, tau = 1, k1 = 50, seed = 200,
  print_iter = TRUE, var_cal = FALSE, ll_obs_cal = FALSE)
```

Arguments

X.obs	Design matrix with missingness $N \times p$
y	Response vector $N \times 1$
pos_var	Index of selected covariates. The default is pos_var = 1:ncol(X.obs).
maxruns	Maximum number of iterations. The default is maxruns = 500.
tol_em	The tolerance to stop SAEM. The default is tol_em = 1e-7.
nmcmc	The MCMC length. The default is nmcmc = 2.

tau Rate τ in the step size $(k-k_1)^{-\tau}$. The default is tau = 1.

k1 Number of first iterations k_1 in the step size $(k-k_1)^{-\tau}$. The default is k1=50.

seed An integer as a seed set for the radom generator. The default value is 200.

print_iter If TRUE, miss.saem will print the estimated parameters in each iteration of SAEM.

var_cal If TRUE, miss.saem will calculate the variance of estimated parameters.

ll_obs_cal If TRUE, miss.saem will calculate the observed log-likelihood.

Value

A list with components

mu Estimated μ.

sig2 Estimated Σ.

beta Estiamated β.

time_run Execution time.

seqbeta Sequence of β estimated in each iteration.

seqbeta_avg Sequence of β with averaging in each iteration.

ll Observed log-likelihood.

var_obs Estimated variance for estimated parameters.

std_obs Estimated standard error for estimated parameters.

Examples

```
# Generate dataset
N <- 100  # number of subjects
p <- 3     # number of explanatory variables
mu.star <- rep(0,p)  # mean of the explanatory variables
Sigma.star <- diag(rep(1,p)) # covariance
beta.star <- c(1, 1,  0) # coefficients
beta0.star <- 0 # intercept
beta.true = c(beta0.star,beta.star)
X.complete <- matrix(rnorm(N*p), nrow=N)%*%chol(Sigma.star) +
              matrix(rep(mu.star,N), nrow=N, byrow = TRUE)
p1 <- 1/(1+exp(-X.complete%*%beta.star-beta0.star))
y <- as.numeric(runif(N)<p1)
# Generate missingness
p.miss <- 0.10
patterns <- runif(N*p)<p.miss #missing completely at random
X.obs <- X.complete
X.obs[patterns] <- NA

# SAEM
list.saem = miss.saem(X.obs,y)
print(list.saem$beta)
```

`model_selection` *model_selection*

Description

Model selection for the logistic regression model with missing data.

Usage

```
model_selection(X.obs, y, seed = 200)
```

Arguments

`X.obs` Design matrix with missingness $N \times p$

`y` Response vector $N \times 1$

`seed` An integer as a seed set for the radom generator. The default value is 200.

Value

A list with components

`subset_choose` The index of variates included in the best model selected.

`beta` Estimated β for the best model.

`sig2` Estimated Σ for the best model.

`mu` Estimated μ for the best model.

Examples

```
# Generate dataset
N <- 40  # number of subjects
p <- 3     # number of explanatory variables
mu.star <- rep(0,p)  # mean of the explanatory variables
Sigma.star <- diag(rep(1,p)) # covariance
beta.star <- c(1, 1,  0) # coefficients
beta0.star <- 0 # intercept
beta.true = c(beta0.star,beta.star)
X.complete <- matrix(rnorm(N*p), nrow=N)%*%chol(Sigma.star) +
              matrix(rep(mu.star,N), nrow=N, byrow = TRUE)
p1 <- 1/(1+exp(-X.complete%*%beta.star-beta0.star))
y <- as.numeric(runif(N)<p1)
# Generate missingness
p.miss <- 0.10
patterns <- runif(N*p)<p.miss #missing completely at random
X.obs <- X.complete
X.obs[patterns] <- NA
# model selection for SAEM
list.saem.select = model_selection(X.obs,y)
print(list.saem.select$subset_choose)
print(list.saem.select$beta)
```

pred_saem *pred_saem*

Description

Prediction on test with missing values for the logistic regression model.

Usage

```
pred_saem(X.test, beta.saem, mu.saem, sig2.saem, seed = 200,
  method = "map")
```

Arguments

X.test	Design matrix in test set.
beta.saem	Estimated β by SAEM.
mu.saem	Estimated μ by SAEM.
sig2.saem	Estimated Σ by SAEM.
seed	An integer as a seed set for the radom generator. The default value is 200.
method	The name of method to deal with missing values in test set. It can be 'map'(maximum a posteriori) or 'impute' (imputation by conditional expectation). Default is 'map'.

Value

pr.saem	The prediction result for logistic regression: the probability of response y=1.

Examples

```
# Generate dataset
N <- 100  # number of subjects
p <- 3     # number of explanatory variables
mu.star <- rep(0,p)  # mean of the explanatory variables
Sigma.star <- diag(rep(1,p)) # covariance
beta.star <- c(1, 1,  0) # coefficients
beta0.star <- 0 # intercept
beta.true = c(beta0.star,beta.star)
X.complete <- matrix(rnorm(N*p), nrow=N)%*%chol(Sigma.star) +
              matrix(rep(mu.star,N), nrow=N, byrow = TRUE)
p1 <- 1/(1+exp(-X.complete%*%beta.star-beta0.star))
y <- as.numeric(runif(N)<p1)
# Generate missingness
p.miss <- 0.10
patterns <- runif(N*p)<p.miss #missing completely at random
X.obs <- X.complete
X.obs[patterns] <- NA
```

```
# SAEM
list.saem = miss.saem(X.obs,y)

# Generate test set with missingness
Nt = 50
X.test <- matrix(rnorm(Nt*p), nrow=Nt)%*%chol(Sigma.star)+
              matrix(rep(mu.star,Nt), nrow=Nt, byrow = TRUE)
p1 <- 1/(1+exp(-X.test%*%beta.star-beta0.star))
y.test <- as.numeric(runif(Nt)<p1)

# Prediction on test set
pr.saem <- pred_saem(X.test, list.saem$beta, list.saem$mu, list.saem$sig2)
pred.saem <- (pr.saem>0.5)*1
table(y.test, pred.saem)
```

Index