Package: VARSELECTEXPOSURE (via r-universe)

August 22, 2024

Type Package

Title Variable Selection Methods Including an Exposure Variable

Version 1.0.3

Author Alex Martinez [aut, cre], Andrew Chapple [aut]

Maintainer Alex Martinez <ahmartinez283@gmail.com>

Description Utilizes multiple variable selection methods to estimate Average Treatment Effect.

License GPL-2

LinkingTo Rcpp, RcppArmadillo

Encoding UTF-8

RoxygenNote 7.2.1

NeedsCompilation yes

Repository CRAN

Date/Publication 2023-04-27 13:50:08 UTC

Contents

BACKWARD_EXPOSURE *Performs deviance-based backwards variable selection in logistic regression with an exposure.*

Description

Returns the estimated Average Treatment Effect and estimated Relative Treatment Effect calculated by the optimal model chosen via backward selection including an exposure variable.

Usage

```
BACKWARD_EXPOSURE(Data)
```

Arguments

Data	Data frame containing outcome variable (Y), exposure variable (E), and candidate covariates.

Value

List containing (1) the estimated Average Treatment Effect, (2) estimated Relative Treatment Effect, (3) summary of the selected model, and (4) the first 6 rows of the data frame containing backward-selected covariates.

References

[1] **will contain our paper later**

Examples

```
###Generate data with n rows and p covariates, can be any number but we'll choose 750 rows
###and 7 covariates for this example
set.seed(3)

p = 7
n = 750
beta0 = rnorm(1, mean = 0, sd = 1)
betaE = rnorm(1, mean = 0, sd = 1)
beta0_E = rnorm(1, mean = 0, sd = 1)
betaX_E = c()
betaX_Y = c()
Y = rep(NA, n)
E = rep(NA, n)
pi0 = rep(NA, n)
pi1 = rep(NA, n)
data = data.frame(cbind(Y, E, pi0, pi1))
j = round(runif(1, 0, p))
for(i in 1:p){
  betaX_Y[i] = rnorm(1, mean = 0, sd = 0.5)
```

```
  betaX_E[i] = rnorm(1, mean = 0, sd = 0.5)
}
zeros = sample(1:p, j, replace =  FALSE)
betaX_Y[zeros] = 0
betaX_E[zeros] = 0
mu = 0
sigma = 1
for(i in 1:p){
  covar = rnorm(n, 0, 1)
  data[,i+4] = covar
  names(data)[i+4] = paste("X", i, sep = "")
}
for(i in 1:n){
  p.event_E = beta0_E + sum(betaX_E*data[i,5:(p+4)])
  pi1_E = exp(p.event_E)/(1+exp(p.event_E))
  data[i,2] = rbinom(1, 1, prob = pi1_E)
}
for(i in 1:n){
  p.event = beta0 + betaE + sum(betaX_Y*data[i,5:(p+4)])
  p.noevent = beta0 + sum(betaX_Y*data[i,5:(p+4)])
  pi0 = exp(p.noevent)/(1+exp(p.noevent))
  pi1 = exp(p.event)/(1+exp(p.event))
  data[i,3] = pi0
  data[i,4] = pi1
  if(data[i,2] == 1){
    data[i,1] = rbinom(1, 1, prob = pi1)
  }else{
    data[i,1] = rbinom(1, 1, prob = pi0)
  }
}
for(i in 1:n){
  p.event_E = beta0_E + sum(betaX_E*data[i,5:(p+4)])
  pi1_E = exp(p.event_E)/(1+exp(p.event_E))
  data[i,2] = rbinom(1, 1, prob = pi1_E)
}

###Raw data includes pi0 and pi1 columns used to fill Y and E, so to test
###the function we'll remove these

testdata = data[,-c(3,4)]

BACKWARD_EXPOSURE(testdata)
```

FORWARD_EXPOSURE *Performs deviance-based forwards variable selection in logistic regression with an exposure*

Description

Returns the estimated Average Treatment Effect and estimated Relative Treatment Effect calculated by the optimal model chosen via forward selection including an exposure variable.

Usage

```
FORWARD_EXPOSURE(Data)
```

Arguments

Data	Data frame containing outcome variable (Y), exposure variable (E), and candidate covariates.

Value

List containing (1) the estimated Average Treatment Effect, (2) summary of the selected model, and (3) the first 6 rows of the data frame containing forward-selected covariates.

References

[1] **will contain our paper later**

Examples

```
###Generate data with n rows and p covariates, can be any number but we'll choose 750 rows
###and 7 covariates for this example
set.seed(3)

p = 7
n = 750
beta0 = rnorm(1, mean = 0, sd = 1)
betaE = rnorm(1, mean = 0, sd = 1)
beta0_E = rnorm(1, mean = 0, sd = 1)
betaX_E = c()
betaX_Y = c()
Y = rep(NA, n)
E = rep(NA, n)
pi0 = rep(NA, n)
pi1 = rep(NA, n)
data = data.frame(cbind(Y, E, pi0, pi1))
j = round(runif(1, 0, p))
for(i in 1:p){
  betaX_Y[i] = rnorm(1, mean = 0, sd = 0.5)
  betaX_E[i] = rnorm(1, mean = 0, sd = 0.5)
}
zeros = sample(1:p, j, replace =  FALSE)
betaX_Y[zeros] = 0
betaX_E[zeros] = 0
mu = 0
sigma = 1
for(i in 1:p){
  covar = rnorm(n, 0, 1)
  data[,i+4] = covar
  names(data)[i+4] = paste("X", i, sep = "")
}
for(i in 1:n){
  p.event_E = beta0_E + sum(betaX_E*data[i,5:(p+4)])
```

```
  pi1_E = exp(p.event_E)/(1+exp(p.event_E))
  data[i,2] = rbinom(1, 1, prob = pi1_E)
}
for(i in 1:n){
  p.event = beta0 + betaE + sum(betaX_Y*data[i,5:(p+4)])
  p.noevent = beta0 + sum(betaX_Y*data[i,5:(p+4)])
  pi0 = exp(p.noevent)/(1+exp(p.noevent))
  pi1 = exp(p.event)/(1+exp(p.event))
  data[i,3] = pi0
  data[i,4] = pi1
  if(data[i,2] == 1){
    data[i,1] = rbinom(1, 1, prob = pi1)
  }else{
    data[i,1] = rbinom(1, 1, prob = pi0)
  }
}
for(i in 1:n){
  p.event_E = beta0_E + sum(betaX_E*data[i,5:(p+4)])
  pi1_E = exp(p.event_E)/(1+exp(p.event_E))
  data[i,2] = rbinom(1, 1, prob = pi1_E)
}

###Raw data includes pi0 and pi1 columns used to fill Y and E, so to test
###the function we'll remove these

testdata = data[,-c(3,4)]

FORWARD_EXPOSURE(testdata)
```

LIKE *Obtains likelihood*

Description

Calculates likelihood from observed outcome data and given covariate data/parameters.

Usage

```
LIKE(Y, X, beta0, beta)
```

Arguments

Y	Binary outcome vector.
X	Matrix of covariates.
beta0	Intercept parameter.
beta	Vector of covariate parameters of length p.

Value

Likelihood

MCMC_LOGIT_KEEP *Obtains posterior samples from an MCMC algorithm to perform variable selection.*

Description

Performs posterior sampling from an MCMC algorithm to estimate average treatment effect and posterior probability of inclusion of candidate variables.

Usage

```
MCMC_LOGIT_KEEP(Y, Z, PIN, MAX_COV, SdBeta, NUM_REPS)
```

Arguments

Y	Binary outcome vector.
Z	Matrix of covariates including binary exposure variable.
PIN	Prior probability of inclusion of candidate variables.
MAX_COV	Maximum number of covariates in desired model.
SdBeta	Prior standard deviation for generating distrubtion of proposal coefficients.
NUM_REPS	Number of MCMC iterations to perform.

Value

List containing (1) the posterior distribution of the estimated Average Treatment Effect, (2) the posterior distributions of the intercept parameter, (3) the posterior distributions of the rest of the coefficients including the exposure coefficient, and (4) the posterior distribution for the indication of whether or not the variable was included in a given iteration's model.

STEPWISE_EXPOSURE *Performs deviance-based stepwise variable selection in logistic regression with an exposure*

Description

Returns the estimated Average Treatment Effect and estimated Relative Treatment Effect calculated by the optimal model chosen via stepwise selection including an exposure variable.

Usage

```
STEPWISE_EXPOSURE(Data)
```

Arguments

Data	Data frame containing outcome variable (Y), exposure variable (E), and candidate covariates.

Value

List containing (1) the estimated Average Treatment Effect, (2) summary of the selected model, and (3) the first 6 rows of the data frame containing stepwise-selected covariates.

References

[1] **will contain our paper later**

Examples

```
###Generate data with n rows and p covariates, can be any number but we'll choose 750 rows
###and 7 covariates for this example
set.seed(3)

p = 7
n = 750
beta0 = rnorm(1, mean = 0, sd = 1)
betaE = rnorm(1, mean = 0, sd = 1)
beta0_E = rnorm(1, mean = 0, sd = 1)
betaX_E = c()
betaX_Y = c()
Y = rep(NA, n)
E = rep(NA, n)
pi0 = rep(NA, n)
pi1 = rep(NA, n)
data = data.frame(cbind(Y, E, pi0, pi1))
j = round(runif(1, 0, p))
for(i in 1:p){
  betaX_Y[i] = rnorm(1, mean = 0, sd = 0.5)
  betaX_E[i] = rnorm(1, mean = 0, sd = 0.5)
}
zeros = sample(1:p, j, replace =  FALSE)
betaX_Y[zeros] = 0
betaX_E[zeros] = 0
mu = 0
sigma = 1
for(i in 1:p){
  covar = rnorm(n, 0, 1)
  data[,i+4] = covar
  names(data)[i+4] = paste("X", i, sep = "")
}
for(i in 1:n){
  p.event_E = beta0_E + sum(betaX_E*data[i,5:(p+4)])
  pi1_E = exp(p.event_E)/(1+exp(p.event_E))
  data[i,2] = rbinom(1, 1, prob = pi1_E)
}
for(i in 1:n){
```

```
  p.event = beta0 + betaE + sum(betaX_Y*data[i,5:(p+4)])
  p.noevent = beta0 + sum(betaX_Y*data[i,5:(p+4)])
  pi0 = exp(p.noevent)/(1+exp(p.noevent))
  pi1 = exp(p.event)/(1+exp(p.event))
  data[i,3] = pi0
  data[i,4] = pi1
  if(data[i,2] == 1){
    data[i,1] = rbinom(1, 1, prob = pi1)
  }else{
    data[i,1] = rbinom(1, 1, prob = pi0)
  }
}
for(i in 1:n){
  p.event_E = beta0_E + sum(betaX_E*data[i,5:(p+4)])
  pi1_E = exp(p.event_E)/(1+exp(p.event_E))
  data[i,2] = rbinom(1, 1, prob = pi1_E)
}

###Raw data includes pi0 and pi1 columns used to fill Y and E, so to test
###the function we'll remove these

testdata = data[,-c(3,4)]

STEPWISE_EXPOSURE(testdata)
```

Index