Package: DRDRtest (via r-universe)

September 9, 2024

Title A Nonparametric Doubly Robust Test for Continuous Treatment Effect

Version 0.1

Description Implement the statistical test proposed in Weng et al. (2021) to test whether the average treatment effect curve is constant and whether a discrete covariate is a significant effect modifier.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.1.2

Imports KernSmooth, stats, SuperLearner

NeedsCompilation no

Author Guangwei Weng [aut, cre]

Maintainer Guangwei Weng <wengx076@umn.edu>

Date/Publication 2021-09-28 09:10:06 UTC

Repository https://weng-gw.r-universe.dev

RemoteUrl https://github.com/cran/DRDRtest

RemoteRef HEAD

RemoteSha 2c26c56b78c126fa95eec181f70c888dca500d46

Contents

`drdrtest` *The function for performing tests of average treatment effects with user specified nuisance functions*

Description

This is the function for testing average treatment effects with user specified nuisance functions.

Usage

```
drdrtest(
  y,
  a,
  l,
  arange,
  pifunc,
  mufunc,
  h = NULL,
  b = 1000,
  dist = "TwoPoint",
  pi.low = 0.01,
  a.grid.size = 401
)
```

Arguments

`y`	A vector containing the outcomes for each observation
`a`	A vector containing the treatment levels (dosage) for each observation
`l`	A data.frame containing the observations of covariates
`arange`	A vector of length 2 giving the lower bound and upper bound of treatment levels
`pifunc`	A user specifid function or wapper that takes treatment a as the first argument and covariates l as the second argument and return propensit scores
`mufunc`	A user specifid function or wapper that takes treatment a as the first argument and covariates l as the second argument and return outcome regression values
`h`	bandwidth to be used in kernel regression. If not specified, will by default use "rule of thumb" bandwidth selector
`b`	number of Bootstrap samples to be generated
`dist`	distibution used to generate residuals for Bootstrap samples. Currently only have two options, "TwoPoint" and "Rademachar"
`pi.low`	Lower bound to truncate propensity scores
`a.grid.size`	size of equally spaced grid points over `arange` to be generate for numerically evaluating the integral in test statistic

Value

A list containing

p.value: P value of the test result

test.stat: Value of the observed test statistic

Bootstrap.samples: A vector containing test statistic values from Bootstrap samples

loc.fit: A list containg evalution points of average treatment effect and the corresponding values

bandwidth: Bandwidth used in kernel regression

Examples

```
mu.mod<-function(a,l,delta,height){
   mu <- as.numeric(l%*%c(0.2,0.2,0.3,-0.1))+triangle(a-2.5,delta,height)+a*(-0.1*l[,1]+0.1*l[,3])
    return(mu)
}
triangle <- function(a,delta,height){
    y <- exp(-a^2/((delta/2)^2))*height
    return(y)
}
set.seed(2000)
n <- 500
d <- 4
sigma <- 0.05
delta <- 1
height <- 0
arange<-c(0.01,4.99)

l <- matrix(rnorm(n*d),ncol=d)
colnames(l) <- paste("l",1:4,sep="")
logit.lambda <- as.numeric(l%*%c(0.1,0.1,-0.1,0.2))
lambda <- exp(logit.lambda)/(1+exp(logit.lambda))
a <- rbeta(n, shape1 = lambda, shape2 =1-lambda)*5

mu <- mu.mod(a,l,delta,height)
residual.list <- rnorm(n,mean=0,sd=sigma)
y <- mu+residual.list

## We use the oracal propensity score and outcome regression for illustration
pifunc <- function(a,l){
   l <- as.matrix(l)
   logit.lambda <- as.numeric(l%*%c(0.1,0.1,-0.1,0.2))
   lambda <- exp(logit.lambda)/(1+exp(logit.lambda))
   return(dbeta(a/5,shape1=lambda,shape2 = 1-lambda)/5)
}

mufunc <- function(a,l){
   l <- as.matrix(l)
   return(mu.mod(a,l,delta,height))
}
out <- drdrtest(y,a,data.frame(l),arange,pifunc,mufunc)
```

`drdrtest.base` *The base function for performing tests of average treatment effects*

Description

This is the base function for testing average treatment effects. Users can use specify the nuisance function values by themselves.

Usage

```
drdrtest.base(
  y,
  a,
  pi,
  varpi,
  mu,
  ma,
  arange,
  h = NULL,
  b = 1000,
  dist = "TwoPoint",
  a.grid.size = 401
)
```

Arguments

`y`	A vector containing the outcomes for each observation
`a`	A vector containing the treatment levels (dosage) for each observation
`pi`	A vector containing the propensity scores for each observation
`varpi`	A vector containing the mean propensity scores for each observation
`mu`	A vector containing the outcome regression function values for each observation
`ma`	A vector containing the mean outcome regression fucntion values for each observation
`arange`	A vector of length 2 giving the lower bound and upper bound of treatment levels
`h`	bandwidth to be used in kernel regression. If not specified, will by default use "rule of thumb" bandwidth selector
`b`	number of Bootstrap samples to be generated
`dist`	distibution used to generate residuals for Bootstrap samples. Currently only have two options, "TwoPoint" and "Rademachar"
`a.grid.size`	size of equally spaced grid points over `arange` to be generate for numerically evaluating the integral in test statistic

Value

A list containing

p.value: P value of the test result

test.stat: Value of the observed test statistic

Bootstrap.samples: A vector containing test statistic values from Bootstrap samples

loc.fit: A list containg evalution points of average treatment effect and the corresponding values

bandwidth: Bandwidth used in kernel regression

Examples

```
mu.mod<-function(a,l,delta,height){
   mu <- as.numeric(l%*%c(0.2,0.2,0.3,-0.1))+triangle(a-2.5,delta,height)+a*(-0.1*l[,1]+0.1*l[,3])
    return(mu)
}
triangle <- function(a,delta,height){
    y <- exp(-a^2/((delta/2)^2))*height
    return(y)
}
set.seed(2000)
n <- 500
d <- 4
sigma <- 0.5
delta <- 1
height <- 0
arange<-c(0.01,4.99)

l <- matrix(rnorm(n*d),ncol=d)
colnames(l) <- paste("l",1:4,sep="")
logit.lambda <- as.numeric(l%*%c(0.1,0.1,-0.1,0.2))
lambda <- exp(logit.lambda)/(1+exp(logit.lambda))
a <- rbeta(n, shape1 = lambda, shape2 =1-lambda)*5

mu <- mu.mod(a,l,delta,height)
residual.list <- rnorm(n,mean=0,sd=sigma)
y <- mu+residual.list

## We use the oracal propensity score and outcome regression for illustration
pilist <- dbeta(a/5, shape1=lambda, shape2 = 1-lambda)/5
varpilist <- colMeans(matrix(dbeta(rep(a,each=n)/5,
                                   shape1=rep(lambda,n),
                                   shape2 = 1-rep(lambda,n))/5, nrow=n))
mulist <- mu
malist <-colMeans(matrix(mu.mod(rep(a,each=n),l[rep(1:n,n),],delta,height),nrow=n))

out <- drdrtest.base(y,a,pilist,varpilist,mulist,malist,arange)
```

`drdrtest.superlearner` *The function for performing tests of average treatment effects with SuperLearner*

Description

This is the function for testing average treatment effects with user specified nuisance functions.

Usage

```
drdrtest.superlearner(
  y,
  a,
  l,
  arange,
  pi.sl.lib = c("SL.earth", "SL.glm", "SL.gam", "SL.glmnet"),
  mu.sl.lib = c("SL.earth", "SL.glm", "SL.gam", "SL.glmnet"),
  mu.family = "gaussian",
  h = NULL,
  b = 1000,
  dist = "TwoPoint",
  a.grid.size = 401,
  pi.low = 0.01,
  pi.var.low = 0.01
)
```

Arguments

`y`	A vector containing the outcomes for each observation
`a`	A vector containing the treatment levels (dosage) for each observation
`l`	A data.frame containing the observations of covariates
`arange`	A vector of length 2 giving the lower bound and upper bound of treatment levels
`pi.sl.lib`	Models will be used by SuperLearner to estiamte propensity scores
`mu.sl.lib`	Models will be used by SuperLearner to estiamte outcome regression function
`mu.family`	Type of response. Currently only support "gaussian" and "binomial"
`h`	bandwidth to be used in kernel regression. If not specified, will by default use "rule of thumb" bandwidth selector
`b`	number of Bootstrap samples to be generated
`dist`	distibution used to generate residuals for Bootstrap samples. Currently only have two options, "TwoPoint" and "Rademachar"
`a.grid.size`	size of equally spaced grid points over `arange` to be generate for numerically evaluating the integral in test statistic
`pi.low`	Lower bound to truncate propensity scores
`pi.var.low`	Lower bound to truncate conditional variance of treament (used in propensity score estimation).

Value

A list containing

p.value: P value of the test result

test.stat: Value of the observed test statistic

Bootstrap.samples: A vector containing test statistic values from Bootstrap samples

loc.fit: A list containg evalution points of average treatment effect and the corresponding values

bandwidth: Bandwidth used in kernel regression

Examples

```
mu.mod<-function(a,l,delta,height){
   mu <- as.numeric(l%*%c(0.2,0.2,0.3,-0.1))+triangle(a-2.5,delta,height)+a*(-0.1*l[,1]+0.1*l[,3])
    return(mu)
}
triangle <- function(a,delta,height){
    y <- exp(-a^2/((delta/2)^2))*height
    return(y)
}
set.seed(2000)
n <- 500
d <- 4
sigma <- 0.05
delta <- 1
height <- 0
arange<-c(0.01,4.99)

l <- matrix(rnorm(n*d),ncol=d)
colnames(l) <- paste("l",1:4,sep="")
logit.lambda <- as.numeric(l%*%c(0.1,0.1,-0.1,0.2))
lambda <- exp(logit.lambda)/(1+exp(logit.lambda))
a <- rbeta(n, shape1 = lambda, shape2 =1-lambda)*5

mu <- mu.mod(a,l,delta,height)
residual.list <- rnorm(n,mean=0,sd=sigma)
y <- mu+residual.list

out <- drdrtest.superlearner(y,a,l,arange,pi.sl.lib=c("SL.glm"),mu.sl.lib=c("SL.glm"))
```

`drdrtest_em` *The base function for testing a effect modifier with user specified nuisance functions*

Description

This is the function for testing whether a discrete covariate is an effect modifier with user specified nuisance functions

Usage

```
drdrtest_em(
  y,
  a,
  l,
  class_label,
  arange,
  pifunc,
  mufunc,
  h = NULL,
  b = 1000,
  dist = "TwoPoint",
  pi.low = 0.01,
  a.grid.size = 401
)
```

Arguments

y	A vector containing the outcomes for each observation
a	A vector containing the treatment levels (dosage) for each observation
l	A data.frame containing the observations of covariates
class_label	A vector containing the class label (label for the effect modifier) for each observation.
arange	A vector of length 2 giving the lower bound and upper bound of treatment levels
pifunc	A user specifid function or wapper that takes treatment a as the first argument and covariates l as the second argument and return propensit scores
mufunc	A user specifid function or wapper that takes treatment a as the first argument and covariates l as the second argument and return outcome regression values
h	bandwidth to be used in kernel regression. If not specified, will by default use "rule of thumb" bandwidth selector
b	number of Bootstrap samples to be generated
dist	distibution used to generate residuals for Bootstrap samples. Currently only have two options, "TwoPoint" and "Rademachar"
pi.low	Lower bound to truncate propensity scores
a.grid.size	size of equally spaced grid points over arange to be generate for numerically evaluating the integral in test statistic

Value

A list containing

p.value: P value of the test result

test.stat: Value of the observed test statistic

Bootstrap.samples: A vector containing test statistic values from Bootstrap samples

bandwidth: Bandwidth used in kernel regression

Examples

```
d <- 4
n <- 200
sigma <- 0.5
delta <- 1
height <-1
arange <- c(0,5)
triangle <- function(a,height){
   y <- exp(-a^2/((1/2)^2))*height
   return(y)
}
mu.mod<-function(a,l,delta,height){
   mu <- as.numeric(l%*%c(0.2,0.2,0.3,-0.1*delta))+
         triangle(a-2.5,height)+a*(-0.1*l[,1]+0.1*delta*l[,4])
   return(mu)
}
l <- matrix(rnorm(n*d),ncol=d)
l[,4] <- ifelse(l[,4]>0,1,0)
colnames(l) <- paste("l",1:4,sep="")

logit.lambda <- as.numeric(l%*%c(0.1,0.1,-0.1,0))
lambda <- exp(logit.lambda)/(1+exp(logit.lambda))
a <- rbeta(n, shape1 = lambda, shape2 =1-lambda)*5

mu <- mu.mod(a,l,delta,height)
residual.list <- rnorm(n,mean=0,sd =sigma)
y <- mu+residual.list

class_label <- l[,4]

pifunc <- function(a,l){
   l <- as.matrix(l)
   logit.lambda <- as.numeric(l%*%c(0.1,0.1,-0.1,0))
   lambda <- exp(logit.lambda)/(1+exp(logit.lambda))
   return(pmin(dbeta(a/5,shape=lambda,shape2=1-lambda)/5,100))
}

mufunc <- function(a,l){
   return(mu.mod(a,as.matrix(l),delta,height))
}

out <- drdrtest_em(y,a,l,class_label,arange,pifunc,mufunc)
```

`drdrtest_em.base` *The base function for testing effect modifiers*

Description

This is the base function for testing whether a discrete covariate is an effect modifier.

Usage

```
drdrtest_em.base(
  ylist,
  alist,
  pilist,
  varpilist,
  mulist,
  malist,
  arange,
  h = NULL,
  b = 1000,
  dist = "TwoPoint",
  a.grid.size = 401
)
```

Arguments

ylist	A list containing vectors of outcomes for each class
alist	A list containing vectors of treatment levels (dosage) for each class
pilist	A list containing vectors of propensity scores for each class
varpilist	A list containing vectors of mean propensity scores for each class
mulist	A list containing vectors of outcome regression function values for each class
malist	A list containing vectors of mean outcome regression values for each class
arange	A vector of length 2 giving the lower bound and upper bound of treatment levels
h	bandwidth to be used in kernel regression. If not specified, will by default use "rule of thumb" bandwidth selector
b	number of Bootstrap samples to be generated
dist	distibution used to generate residuals for Bootstrap samples. Currently only have two options, "TwoPoint" and "Rademachar"
a.grid.size	size of equally spaced grid points over arange to be generate for numerically evaluating the integral in test statistic

Value

A list containing

p.value: P value of the test result

test.stat: Value of the observed test statistic

Bootstrap.samples: A vector containing test statistic values from Bootstrap samples

bandwidth: Bandwidth used in kernel regression

Examples

```
d <- 4
n <- 200
sigma <- 0.5
delta <- 1
height <-1
arange <- c(0,5)
triangle <- function(a,height){
   y <- exp(-a^2/((1/2)^2))*height
   return(y)
}
mu.mod<-function(a,l,delta,height){
   mu <- as.numeric(l%*%c(0.2,0.2,0.3,-0.1*delta))+
         triangle(a-2.5,height)+a*(-0.1*l[,1]+0.1*delta*l[,4])
   return(mu)
}
l <- matrix(rnorm(n*d),ncol=d)
l[,4] <- ifelse(l[,4]>0,1,0)
colnames(l) <- paste("l",1:4,sep="")

logit.lambda <- as.numeric(l%*%c(0.1,0.1,-0.1,0))
lambda <- exp(logit.lambda)/(1+exp(logit.lambda))
a <- rbeta(n, shape1 = lambda, shape2 =1-lambda)*5

mu <- mu.mod(a,l,delta,height)
residual.list <- rnorm(n,mean=0,sd =sigma)
y <- mu+residual.list

class_label <- l[,4]
ylist <- split(y,class_label)
alist <- split(a,class_label)
pilist <- split(pmin(dbeta(a/5,shape1=lambda,shape2=1-lambda)/5,100),class_label)
mulist <- split(mu,class_label)

varpilist <- list()
malist <- list()
for(c in c(0,1)){
   ac <- a[class_label==c]
   lc <- l[class_label==c,]

   logit.lambdac <- as.numeric(lc[rep(1:nrow(lc),nrow(lc)),]%*%c(0.1,0.1,-0.1,0))
   lambdac <- exp(logit.lambdac)/(1+exp(logit.lambdac))
   varpic <- colMeans(matrix(pmin(dbeta(rep(ac,each=length(ac))/5,
                                  shape1=lambdac,
                                  shape2 = 1-lambdac)/5,100),nrow=length(ac)))

   mac <- colMeans(matrix(mu.mod(rep(ac,each=length(ac)),
                                    lc[rep(1:nrow(lc),nrow(lc)),],
                                    delta,height),
                          nrow=length(ac)))

   varpilist[[as.character(c)]]<-varpic
```

```
    malist[[as.character(c)]] <- mac
    }

out <- drdrtest_em.base(ylist,alist,pilist,varpilist,mulist,malist,arange)
```

`drdrtest_em.superlearner`

The function for testing a effect modifier with SuperLearner

Description

This is the function for testing whether a discrete covariate is an effect modifier with SuperLearner

Usage

```
drdrtest_em.superlearner(
  y,
  a,
  l,
  class_label,
  arange,
  pi.sl.lib = c("SL.earth", "SL.glm", "SL.gam", "SL.glmnet"),
  mu.sl.lib = c("SL.earth", "SL.glm", "SL.gam", "SL.glmnet"),
  mu.family = "gaussian",
  h = NULL,
  b = 1000,
  dist = "TwoPoint",
  pi.low = 0.01,
  pi.var.low = 0.01,
  a.grid.size = 401
)
```

Arguments

`y`	A vector containing the outcomes for each observation
`a`	A vector containing the treatment levels (dosage) for each observation
`l`	A data.frame containing the observations of covariates
`class_label`	A vector containing the class label (label for the effect modifier) for each observation.
`arange`	A vector of length 2 giving the lower bound and upper bound of treatment levels
`pi.sl.lib`	Models will be used by SuperLearner to estiamte propensity scores
`mu.sl.lib`	Models will be used by SuperLearner to estiamte outcome regression function
`mu.family`	Type of response. Currently only support "gaussian" and "binomial"
`h`	bandwidth to be used in kernel regression. If not specified, will by default use "rule of thumb" bandwidth selector

b	number of Bootstrap samples to be generated
dist	distibution used to generate residuals for Bootstrap samples. Currently only have two options, "TwoPoint" and "Rademachar"
pi.low	Lower bound to truncate propensity scores
pi.var.low	Lower bound to truncate conditional variance of treament (used in propensity score estimation).
a.grid.size	size of equally spaced grid points over arange to be generate for numerically evaluating the integral in test statistic

Value

A list containing

p.value: P value of the test result

test.stat: Value of the observed test statistic

Bootstrap.samples: A vector containing test statistic values from Bootstrap samples

bandwidth: Bandwidth used in kernel regression

Examples

```
d <- 4
n <- 200
sigma <- 0.5
delta <- 1
height <-1
arange <- c(0,5)
triangle <- function(a,height){
   y <- exp(-a^2/((1/2)^2))*height
   return(y)
}
mu.mod<-function(a,l,delta,height){
   mu <- as.numeric(l%*%c(0.2,0.2,0.3,-0.1*delta))+
         triangle(a-2.5,height)+a*(-0.1*l[,1]+0.1*delta*l[,4])
   return(mu)
}
l <- matrix(rnorm(n*d),ncol=d)
l[,4] <- ifelse(l[,4]>0,1,0)
colnames(l) <- paste("l",1:4,sep="")

logit.lambda <- as.numeric(l%*%c(0.1,0.1,-0.1,0))
lambda <- exp(logit.lambda)/(1+exp(logit.lambda))
a <- rbeta(n, shape1 = lambda, shape2 =1-lambda)*5

mu <- mu.mod(a,l,delta,height)
residual.list <- rnorm(n,mean=0,sd =sigma)
y <- mu+residual.list

class_label <- l[,4]
out <- drdrtest_em.superlearner(y,a,l,l[,4],arange,pi.sl.lib=c("SL.glm"),mu.sl.lib=c("SL.glm"))
```

Index