Package 'simml'

May 24, 2019

Type Package

Title Single-Index Models with Multiple-Links

Version 0.1.0

Author Park, H., Petkova, E., Tarpey, T., Ogden, R.T.

Maintainer Hyung Park <parkh15@nyu.edu>

Description A major challenge in estimating treatment decision rules from a randomized clinical trial dataset with covariates measured at baseline lies in detecting relatively small treatment effect modification-related variability (i.e., the treatment-by-covariates interaction effects on treatment outcomes) against a relatively large non-treatment-related variability (i.e., the main effects of covariates on treatment outcomes). The class of Single-Index Models with Multiple-Links is a novel single-index model specifically designed to estimate a single-index (a linear combination) of the covariates associated with the treatment effect modification-related variability, while allowing a nonlinear association with the treatment outcomes via flexible link functions. The models provide a flexible regression approach to developing treatment decision rules based on patients' data measured at baseline. We refer to Petkova, Tarpey, Su, and Ogden (2017) <doi: 10.1093/biostatistics/kxw035> and ``A constrained single-index model for estimating interactions between a treatment and covariates" (under review, 2019) for detail. The main function of this package is simml().

License GPL-3

Imports mgcv, plyr

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2019-05-24 12:40:03 UTC

R topics documented:

`der.link` *A subfunction used in estimation*

Description

This function computes the 1st derivative of the treatment-specific "smooth" w.r.t. the single index, using finite difference.

Usage

```
der.link(g.fit, arg.number = 2, eps = 10^(-6))
```

Arguments

`g.fit`	a `mgcv::gam` object
`arg.number`	the argument of `g.fit` that we take derivative w.r.t.; the default is `arg.number=2` (i.e., take deriviative w.r.t. the single-index.)
`eps`	a small finite difference used in numerical differentiation.

See Also

`fit.simml`, `simml`

`fit.simml` *Single-index models with multiple-links (the workhorse function)*

Description

`fit.simml` is the workhorse function for Single-index models with multiple-links (SIMML). The function estimates a linear combination (a single-index) of covariates X, and models the treatment-specific outcome y, via treatment-specific nonparametrically-defined link functions.

Usage

```
fit.simml(y, Tr, X, mu.hat = NULL, family = "gaussian",
  ortho.constr = TRUE, bs = "ps", k = 8, alpha.ini = NULL,
  ind.to.be.positive = 1, pen.order = 0, lambda = 0, max.iter = 30,
  eps.iter = 0.01, trace.iter = TRUE)
```

Arguments

y	a n-by-1 vector of treatment outcomes; y is assumed to follow an exponential family distribution; any distribution supported by mgcv::gam.
Tr	a n-by-1 vector of treatment indicators; each element represents one of the L(>1) treatment conditions; e.g., c(1,2,1,1,1...); can be a factor-valued.
X	a n-by-p matrix of pre-treatment covarates.
mu.hat	a n-by-1 vector for efficinecy augmentation provided by the user; the defult is NULL; the optimal choice for this vector is h(E(y|X)), where h is the canonical link function.
family	specifies the distribution of y; e.g., "gaussian", "binomial", "poisson"; the defult is "gaussian"; can be any family supported by mgcv::gam.
ortho.constr	separates the interaction effects from the main effect (without this, the interaction effect can be confounded by the main effect; the default is TRUE.
bs	type of basis for representing the treatment-specific smooths; the defult is "ps" (p-splines); any basis supported by mgcv::gam can be used, e.g., "cr" (cubic regression splines)
k	basis dimension; the same number (k) is used for all treatment groups, however, the smooths of different treatments have different roughness parameters.
alpha.ini	an initial solution of alpha.coef; a p-by-1 vector; the defult is NULL.
ind.to.be.positive	for identifiability of the solution alpha.coef, we restrict the jth component of alpha.coef to be positive; by default j=1.
pen.order	0 indicates the ridge penalty; 1 indicates the 1st difference penalty; 2 indicates the 2nd difference penalty, used in a penalized least squares (LS) estimation of alpha.coef.
lambda	a regularziation parameter associated with the penalized LS of alpha.coef.
max.iter	an integer specifying the maximum number of iterations for alpha.coef update.
eps.iter	a value specifying the convergence criterion of algorithm.
trace.iter	if TRUE, trace the estimation process and print the differences in alpha.coef.

Details

SIMML captures the effect of covariates via a single-index and their interaction with the treatment via nonparametric link functions. Interaction effects are determined by distinct shapes of the link functions. The estimated single-index is useful for comparing differential treatment efficacy. The resulting simml object can be used to estimate an optimal treatment decision rule for a new patient with pretreatment clinical information.

Value

a list of information of the fitted SIMML including

alpha.coef	the estimated single-index coefficients.
g.fit	a mgcv:gam object containing information about the estimated treatment-specific link functions.

alpha.ini	the initial value used in the estimation of alpha.coef
alpha.path	solution path of alpha.coef over the iterations
d.alpha	records the magnitude of change in alpha.coef over the solution path, alpha.path
scale.X	sd of pretreatment covariates X
center.X	mean of pretreatment covariates X
L	number of different treatment options
p	number of pretreatment covariates X
n	number of subjects

Author(s)

Park, Petkova, Tarpey, Ogden

See Also

pred.simml, fit.simml

---

generate.data *A dataset generation function*

Description

generate.data generates an example dataset from a mean model that has a "main" effect component and a treatment-by-covariates interaction effect component (and a random component for noise).

Usage

```
generate.data(n = 200, p = 10, family = "gaussian",
  correlationX = 0, sigmaX = 1, sigma = 0.4, w = 2, delta = 1,
  pi.1 = 0.5, true.alpha = NULL, true.eta = NULL)
```

Arguments

n	sample size.
p	dimension of covariates.
family	specifies the distribution of the outcome y; "gaussian", "binomial", "poisson"; the defult is "gaussian"
correlationX	correlation among the covariates.
sigmaX	standard deviation of the covariates.
sigma	standard deviation of the random noise term (for gaussian response).
w	controls the nonliarity of the treatment-specific link functions that define the interaction effect component.

	`w=1` linear
	`w=2` nonlinear
`delta`	controls the intensity of the main effect; can take any intermediate value, e.g., `delta= 1.4`.
	`delta=1` moderate main effect
	`delta=2` big main effect
`pi.1`	probability of being assigned to the treatment 1
`true.alpha`	a p-by-1 vector of the true single-index coefficients (associated with the interaction effect component); if `NULL`, `true.alpha` is set to be `(1, 0.5, 0.25, 0.125, 0,...0)'` (only the first 4 elements are nonzero).
`true.eta`	a p-by-1 vector of the true main effect coefficients; if `NULL`, `true.eta` is set to be `(0,..., 0.125, 0.25, 0.25, 1)'` (only the last 4 elements are nonzero).

Value

`y`	a n-by-1 vector of treatment outcomes.
`Tr`	a n-by-1 vector of treatment indicators.
`X`	a n-by-p matrix of pretreatment covariates.
`SNR`	the "signal" (interaction effect) to "nuisance" (main effect) variance ratio (SNR) in the canonical parameter function.
`true.alpha`	the true single-index coefficient vector.
`true.eta`	the true main effect coefficient vector.
`optTr`	a n-by-1 vector of treatments, indicating the optimal treatment selections.
`value.opt`	the "value" implied by the optimal treatment decision rule, `optTr`.

`pred.simml` *SIMML prediction function*

Description

This function makes predictions from an estimated SIMML, given a (new) set of pretreatment covariates. The function returns a set of predicted outcomes for each treatment condition and a set of recommended treatment assignments (assuming a larger value of the outcome is better).

Usage

```
pred.simml(simml.obj, newx, type = "response", maximize = TRUE)
```

Arguments

simml.obj	a simml object
newx	a (n-by-p) matrix of new values for the pretreatment covariates X at which predictions are to be made.
type	the type of prediction required; the default "response" is on the scale of the response variable; the alternative "link" is on the scale of the linear predictors.
maximize	the default is TRUE, assuming a larger value of the outcome is better; if FALSE, a smaller value is assumed to be prefered.

Value

pred.new	a (n-by-L) matrix of predicted values; each column represents a treatment option.
trt.rule	a (n-by-1) vector of suggested treatment assignments

Author(s)

Park, Petkova, Tarpey, Ogden

See Also

simml,fit.simml

---

simml *Single-index models with multiple-links (the main function)*

Description

simml is the wrapper function for Single-index models with multiple-links (SIMML). The function estimates a linear combination (a single-index) of covariates X, and models the treatment-specific outcome y, via treatment-specific nonparametrically-defined link functions.

Usage

```
simml(y, Tr, X, mu.hat = NULL, family = "gaussian",
  ortho.constr = TRUE, bs = "ps", k = 8, alpha.ini = NULL,
  ind.to.be.positive = 1, pen.order = 0, lambda = 0, max.iter = 30,
  eps.iter = 0.01, trace.iter = TRUE, bootstrap = FALSE,
  nboot = 200, boot.alpha = 0.05, seed = 1357)
```

Arguments

y	a n-by-1 vector of treatment outcomes; y is assumed to follow an exponential family distribution; any distribution supported by mgcv::gam.
Tr	a n-by-1 vector of treatment indicators; each element represents one of the L(>1) treatment conditions; e.g., c(1,2,1,1,1...); can be a factor-valued.
X	a n-by-p matrix of pre-treatment covarates.
mu.hat	a n-by-1 vector for efficinecy augmentation provided by the user; the defult is NULL; the optimal choice for this vector is h(E(y|X)), where h is the canonical link function.
family	specifies the distribution of y; e.g., "gaussian", "binomial", "poisson"; the defult is "gaussian"; can be any family supported by mgcv::gam.
ortho.constr	separates the interaction effects from the main effect (without this, the interaction effect can be confounded by the main effect; the default is TRUE.
bs	type of basis for representing the treatment-specific smooths; the defult is "ps" (p-splines); any basis supported by mgcv::gam can be used, e.g., "cr" (cubic regression splines)
k	basis dimension; the same number (k) is used for all treatment groups, however, the smooths of different treatments have different roughness parameters.
alpha.ini	an initial solution of alpha.coef; a p-by-1 vector; the defult is NULL.
ind.to.be.positive	for identifiability of the solution alpha.coef, we restrict the jth component of alpha.coef to be positive; by default j=1.
pen.order	0 indicates the ridge penalty; 1 indicates the 1st difference penalty; 2 indicates the 2nd difference penalty, used in a penalized least squares (LS) estimation of alpha.coef.
lambda	a regularziation parameter associated with the penalized LS of alpha.coef.
max.iter	an integer specifying the maximum number of iterations for alpha.coef update.
eps.iter	a value specifying the convergence criterion of algorithm.
trace.iter	if TRUE, trace the estimation process and print the differences in alpha.coef.
bootstrap	if TRUE, compute bootstrap confidence intervals for the single-index coefficients, alpha.coef; the default is FALSE.
nboot	when bootstrap=TRUE, a value specifying the number of bootstrap replications.
boot.alpha	specifies bootstrap CI percentiles; e.g., 0.05 gives 95% CIs; 0.1 gives 90% CIs.
seed	when bootstrap=TRUE, randomization seed used in bootstrap resampling.

Details

SIMML captures the effect of covariates via a single-index and their interaction with the treatment via nonparametric link functions. Interaction effects are determined by distinct shapes of the link functions. The estimated single-index is useful for comparing differential treatment efficacy. The resulting simml object can be used to estimate an optimal treatment decision rule for a new patient with pretreatment clinical information.

Value

a list of information of the fitted SIMML including

alpha.coef	the estimated single-index coefficients.
g.fit	a mgcv:gam object containing information about the estimated treatment-specific link functions.
alpha.ini	the initial value used in the estimation of alpha.coef
alpha.path	solution path of alpha.coef over the iterations
d.alpha	records the change in alpha.coef over the solution path, alpha.path
scale.X	sd of pretreatment covariates X
center.X	mean of pretreatment covariates X
L	number of different treatment options
p	number of pretreatment covariates X
n	number of subjects
boot.ci	(1-boot.alpha/2) percentile bootstrap CIs (LB, UB) associated with alpha.coef

Author(s)

Park, Petkova, Tarpey, Ogden

See Also

pred.simml, fit.simml

Examples

```
## application of SIMML (on a simulated dataset) (see help(generate.data) for data generation).

family <- "gaussian"   #"poisson"
delta = 1              # moderate main effect
w=2                    # if w=2 (w=1), a nonlinear (linear) contrast function
n=500                  # number of subjects
p=10                   # number of pretreatment covariates

# generate a training dataset
data <- generate.data(n= n, p=p, delta = delta, w= w, family = family)
data$SNR  # the ratio of interactions("signal") vs. main effects("noise") in the canonical param.
Tr <- data$Tr
y  <- data$y
X <- data$X

# generate a (large, 10^5) testing dataset
data.test <- generate.data(n=10^5, p=p, delta = delta,  w= w, family = family)
Tr.test <- data.test$Tr
y.test  <- data.test$y
X.test <- data.test$X
data.test$value.opt     # the optimal "value"
```

```
##  estimate SIMML
#1) SIMML without efficiency augmenation
simml.obj1 <- simml(y, Tr, X, family = family)

#2) SIMML with efficiency augmenation
# we can improove efficinecy by using the efficiency augmentation term, mu.hat.
# mu.hat is estimated by a main effect only model (y~ X).
glm.fit <- glm(y ~ X, family=family)  # could also use cv.glmnet to obtain a mu.hat
mu.hat <- as.vector(predict(glm.fit, newx =X, type="link"))
simml.obj2 <- simml(y, Tr, X, mu.hat = mu.hat, family = family)


## apply the estimated SIMMLs to the testing set and obtain treatment assignment rules.
simml.trt.rule1 <- pred.simml(simml.obj1, newx= X.test)$trt.rule
# "value" estimation (estimated by IPWE)
simml.value1 <-  mean(y.test[simml.trt.rule1 == Tr.test])
simml.value1

simml.trt.rule2 <- pred.simml(simml.obj2, newx= X.test)$trt.rule
# "value" estimation (estimated by IPWE)
simml.value2 <-  mean(y.test[simml.trt.rule2 == Tr.test])
simml.value2

# compare these to the optimal "value"
data.test$value.opt


## estimate the MC (modified covariates) model of Tien et al 2014

n.t <- summary(as.factor(Tr)); pi.t <- n.t/sum(n.t)
mc <-  (as.numeric(Tr) + pi.t[1] -2) *cbind(1, X)   # 0.5*(-1)^as.numeric(Tr) *cbind(1, X)
mc.coef  <-  coef(glm(y ~ mc, family =  family))
mc.trt.rule <- (cbind(1, X.test) %*% mc.coef[-1] > 0) +1
# "value" estimation (estimated by IPWE)
mc.value  <-  mean(y.test[mc.trt.rule == Tr.test])
mc.value


## visualization of the estimated treatment-specific link functions of SIMML
simml.obj1$alpha.coef      # estimated single-index coefficients
g.fit <- simml.obj1$g.fit  # estimated trt-specific link functions; "g.fit" is a mgcv::gam object.
#plot(g.fit)


## by using the package "mgcViz", we can improve the visualization.
# install.packages("mgcViz")
# mgcViz depends on "rgl". "rgl" depends on XQuartz, which you can download from xquartz.org
# library(mgcViz)

## transform the "mgcv::gam" object to a "mgcViz" object (to improve visualization)
#g.fit <- getViz(g.fit)
```

```
#plot1  <- plot( sm(g.fit,1) )  # for treatment group 1
#plot1 + l_fitLine(colour = "red") + l_rug(mapping = aes(x=x, y=y), alpha = 0.8) +
#  l_ciLine(mul = 5, colour = "blue", linetype = 2) + l_points(shape = 19, size = 1, alpha = 0.1) +
#  xlab(expression(paste("z = ", alpha*minute, "x")))  +  ylab("y") +
#  ggtitle("Treatment group 1 (Tr =1)") +  theme_classic()

#plot2 <- plot( sm(g.fit,2) )   # for treatment group 2
#plot2 + l_fitLine(colour = "red") + l_rug(mapping = aes(x=x, y=y), alpha = 0.8) +
#  l_ciLine(mul = 5, colour = "blue", linetype = 2) + l_points(shape = 19, size = 1, alpha = 0.1) +
#  xlab(expression(paste("z = ", alpha*minute, "x"))) +ylab("y") +
#  ggtitle("Treatment group 2 (Tr =2)") + theme_classic()


#trans = function(x) x + g.fit$coefficients[2]

#plotDiff(s1 = sm(g.fit, 2), s2 = sm(g.fit, 1), trans=trans) +  l_ciPoly() +
#  l_fitLine() + geom_hline(yintercept = 0, linetype = 2) +
#  xlab(expression(paste("z = ", alpha*minute, "x")) ) +
#  ylab("(Treatment 2 effect) - (Treatment 1 effect)") +
#  ggtitle("Contrast between two treatment effects") +
#  #geom_vline(xintercept=-0.45, linetype="dotted", color = "red", size=0.8) +
#  theme_classic()


# another way of visualization, using ggplot2
#library(ggplot2)
#dat  <- data.frame(y= simml.obj1$g.fit$model$y,
#                   x= simml.obj1$g.fit$model$single.index,
#                   Treatment= simml.obj1$g.fit$model$Tr)
#g.plot <- ggplot(dat, aes(x=x, y=y, color=Treatment, shape=Treatment, linetype=Treatment)) +
#   geom_point(aes(color=Treatment, shape=Treatment), size=1, fill="white") +
#   scale_colour_brewer(palette="Set1", direction=-1) + theme_classic() +
#   xlab(expression(paste(alpha*minute,"x"))) + ylab("y")
#g.plot + geom_smooth(method=gam, formula= y~ s(x, bs=simml.obj1$bs, k=simml.obj1$k),
#                     se=TRUE, fullrange=TRUE, alpha = 0.35)


#### can obtain bootstrap CIs associated with the single-index coefficients (alpha.coef).
glm.fit <- glm(y ~ X, family=family)  # could also use cv.glmnet.
mu.hat <- as.vector(predict(glm.fit, newx= X, type= "link"))    # efficiency augmentation vector
simml.obj <- simml(y,Tr,X, mu.hat=mu.hat, family=family, bootstrap =TRUE, nboot=15, max.iter=7)
# the default is to use 200 bootstrap replications.
simml.obj$alpha.coef
simml.obj$boot.ci    # displays a set of (1-boot.alpha/2) percentile bootstrap CIs (LB, UB).

# compare the estimates to the true alpha.coef.
data$true.alpha
```

Index