Package ‘FindIt’

November 20, 2019

Version 1.2.0
Date 2019-11-20
Title Finding Heterogeneous Treatment Effects
Author Naoki Egami, Marc Ratkovic, Kosuke Imai
Maintainer Naoki Egami <negami@princeton.edu>
Depends R (>= 3.1.0), arm
Imports glmnet, lars, Matrix, quadprog, glinternet, igraph, sandwich,
lmtest, stats, graphics, utils, limSolve
Description The heterogeneous treatment effect estimation procedure
proposed by Imai and Ratkovic (2013)<DOI: 10.1214/12-AOAS593>.
The proposed method is applicable, for
example, when selecting a small number of most (or least)
efficacious treatments from a large number of alternative
treatments as well as when identifying subsets of the
population who benefit (or are harmed by) a treatment of
interest. The method adapts the Support Vector Machine
classifier by placing separate LASSO constraints over the
pre-treatment parameters and causal heterogeneity parameters of
interest. This allows for the qualitative distinction between
causal and other parameters, thereby making the variable
selection suitable for the exploration of causal heterogeneity.
The package also contains a class of functions, CausalANOVA,
which estimates the average marginal interaction effects (AMIEs)
by a regularized ANOVA as pro-
It contains a variety of regularization techniques to facilitate
analysis of large factorial experiments.

LazyLoad yes
LazyData yes
License GPL (>= 2)
Encoding UTF-8
RoxygenNote 7.0.0
FindIt-package

**NeedsCompilation**: no  
**Repository**: CRAN  
**Date/Publication**: 2019-11-20 21:50:02 UTC

**R topics documented:**

- FindIt-package  
- Carlson  
- CausalANOVA  
- ConditionalEffect  
- cv.CausalANOVA  
- FindIt  
- GerberGreen  
- LaLonde  
- plot.CausalANOVA  
- plot.PredictFindIt  
- predict.FindIt  
- summary.CausalANOVA  
- summary.FindIt  
- test.CausalANOVA

Index 26

---

**FindIt-package**  

**FindIt: Finding Heterogeneous Treatment Effects**

**Description**

<table>
<thead>
<tr>
<th>Package:</th>
<th>factorEx</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type:</td>
<td>Package</td>
</tr>
<tr>
<td>Version:</td>
<td>1.1.5</td>
</tr>
<tr>
<td>Date:</td>
<td>2019-11-19</td>
</tr>
</tbody>
</table>

**Author(s)**

Naoki Egami, Marc Ratkovic and Kosuke Imai.  
Maintainer: Naoki Egami <negami@princeton.edu>

**References**


Carlson Data from conjoint analysis in Carlson (2015).

Description

This data set gives the outcomes as well as treatment assignments of the conjoint analysis in Carlson (2015). Please Carlson (2015) and Egami and Imai (2019) for more details.

Format

A data frame consisting of 7 columns (including a treatment assignment vector) and 3232 observations.

<table>
<thead>
<tr>
<th>outcome</th>
<th>newRecordF</th>
<th>promise</th>
<th>coeth_voting</th>
<th>Degree</th>
</tr>
</thead>
<tbody>
<tr>
<td>integer</td>
<td>factor</td>
<td>factor</td>
<td>factor</td>
<td>factor</td>
</tr>
<tr>
<td>whether a profile is chosen</td>
<td>record as a politician</td>
<td>platform</td>
<td>whether a profile is coethnic to a respondent</td>
<td>job whether a profile has relevant degrees</td>
</tr>
<tr>
<td>0,1</td>
<td>7 levels</td>
<td>3 levels (job, clinic, education)</td>
<td>Yes, No</td>
<td>4 Yes, No</td>
</tr>
</tbody>
</table>

Source

Data from Carlson (2015).

References


CausalANOVA Estimating the AMEs and AMIEs with the CausalANOVA.

Description

CausalANOVA estimates coefficients of the specified ANOVA with regularization. By taking differences in coefficients, the function recovers the AMEs and AMIEs.

Usage

CausalANOVA(
int2.formula = NULL,
int3.formula = NULL,
data,
nway = 1,
pair.id = NULL,
diff = FALSE,
screen = FALSE,
screen.type = "fixed",
screen.num.int = 3,
collapse = FALSE,
collapse.type = "fixed",
collapse.cost = 0.3,
family = "binomial",
cluster = NULL,
maxIter = 50,
eps = 1e-05,
fac.level = NULL,
ord.fac = NULL,
select.prob = FALSE,
boot = 100,
seed = 1234,
verbose = TRUE
)

Arguments

formula A formula that specifies outcome and treatment variables.
int2.formula (optional). A formula that specifies two-way interactions.
int3.formula (optional). A formula that specifies three-way interactions.
data An optional data frame, list or environment (or object coercible by 'as.data.frame' to a data frame) containing the variables in the model. If not found in 'data', the variables are taken from 'environment(formula)', typically the environment from which 'CausalANOVA' is called.
nway With nway=1, the function estimates the Average Marginal Effects (AMEs) only. With nway=2, the function estimates the AMEs and the two-way Average Marginal Interaction Effects (AMIEs). With nway=3, the function estimates the AMEs, the two-way and three-way AMIEs. Default is 1.
pair.id (optional). Unique identifiers for each pair of comparison. This option is used when diff=TRUE.
diff A logical indicating whether the outcome is the choice between a pair. If diff=TRUE, pair.id should specify a pair of comparison. Default is FALSE.
screen A logical indicating whether select significant factor interactions with glmnet. When users specify interactions using int2.formula or int3.formula, this option is ignored. screen should be used only when users want data-driven selection of factor-interactions. With screen.type, users can specify how to screen factor interactions. We recommend to use this option when the number of factors is large, e.g., more than 6. Default is FALSE.
screen.type  Type for screening factor interactions. (1) "fixed" select the fixed number (specified by screen.num.int) of factor interactions. (2) "cv.min" selects factor-interactions with the tuning parameter giving the minimum cross-validation error. (3) "cv.1Std" selects factor-interactions with the tuning parameter giving a cross-validation error that is within 1 standard deviation of the minimum cv error.

screen.num.int (optional). The number of factor interactions to select. This option is used when and screen=TRUE and screen.type="fixed". Default is 3.

collapse  A logical indicating whether to collapse insignificant levels within factors. With collapse.type, users can specify how to collapse levels within factors. We recommend to use this option when the number of levels is large, e.g., more than 6. Default is FALSE.

collapse.type  Type for collapsing levels within factors. (1) "fixed" collapses levels with the fixed cost parameter (specified by collapse.cost). (2) "cv.min" collapses levels with the cost parameter giving the minimum cross-validation error. This option might take time. (3) "cv.1Std" collapses with the cost parameter giving a cross-validation error that is within 1 standard deviation of the minimum cv error. This option might take time.

collapse.cost (optional). A cost parameter ranging from 0 to 1. 1 corresponds to no collapsing. The closer to 0, the stronger regularization. Default is 0.3.

family  A family of outcome variables. "gaussian" when continuous outcomes "binomial" when binary outcomes. Default is "binomial".

cluster  Unique identifies with which cluster standard errors are computed.

maxIter  The number of maximum iteration for glinternet.

eps  A tolerance parameter in the internal optimization algorithm.

fac.level (optional). A vector containing the number of levels in each factor. The order of fac.level should match to the order of columns in the data. For example, when the first and second columns of the design matrix is "Education" and "Race", the first and second element of fac.level should be the number of levels in "Education" and "Race", respectively.

ord.fac (optional). Logical vectors indicating whether each factor has ordered (TRUE) or unordered (FALSE) levels. When levels are ordered, the function uses the order given by function levels(). If levels are ordered, the function places penalties on the differences between adjacent levels. If levels are unordered, the function places penalties on the differences based on every pairwise comparison.

select.prob (optional). A logical indicating whether selection probabilities are computed. This option might take time.

boot  The number of bootstrap replicates for select.prob. Default is 50.

seed  Seed for bootstrap.

verbose  Whether it prints the value of a cost parameter used.

Details

Regularization: screen and collapse.
Users can implement regularization in order to reduce false discovery rate and facilitate interpretation. This is particularly useful when analyzing factorial experiments with a large number of factors, each having many levels.

- When `screen=TRUE`, the function selects significant factor interactions with `glinternet` (Lim and Hastie 2015) before estimating the AMEs and AMIEs. This option is recommended when there are many factors, e.g., more than 6 factors. Alternatively, users can pre-specify interactions of interest using `int2.formula` and `int3.formula`.

- When `collapse=TRUE`, the function collapses insignificant levels within each factor by GashANOVA (Post and Bondell 2013) before estimating the AMEs and AMIEs. This option is recommended when there are many levels within some factors, e.g., more than 6 levels.

**Inference after Regularization:**

- When `screen=TRUE` or `collapse=TRUE`, in order to make valid inference after regularization, we recommend to use `test.CausalANOVA` function. It takes the output from `CausalANOVA` function and estimate the AMEs and AMIEs with `newdata` and provide confidence intervals. Ideally, users should split samples into two; use a half for regularization with `CausalANOVA` function and use the other half for inference with `test.CausalANOVA`.

- If users do not need regularization, specify `screen=FALSE` and `collapse=FALSE`. The function estimates the AMEs and AMIEs and compute confidence intervals with the full sample.

**Suggested Workflow:** (See Examples below as well)

1. Specify the order of levels within each factor using `levels()`. When `collapse=TRUE`, the function places penalties on the differences between adjacent levels when levels are ordered, it is crucial to specify the order of levels within each factor carefully.

2. Run `CausalANOVA`.
   - (a) Specify `formula` to indicate outcomes and treatment variables and `nway` to indicate the order of interactions.
   - (b) Specify `diff=TRUE` and `pair.id` if the outcome is the choice between a pair.
   - (c) Specify `screen`. `screen=TRUE` to implement data-driven selection of factor interactions. `screen=FALSE` to specify interactions through `int2.formula` and `int3.formula` by hand.
   - (d) Specify `collapse`. `collapse=TRUE` to implement data-driven collapsing of insignificant levels. `collapse=FALSE` to use the original number of levels.

3. Run `test.CausalANOVA` when `select=TRUE` or `collapse=TRUE`.

4. Run `summary` and `plot` to explore the AMEs and AMIEs.

5. Estimate conditional effects using `ConditionalEffect` function and visualize them using `plot` function.

**Value**

- `intercept`: An intercept of the estimated ANOVA model. If `diff=TRUE`, this should be close to 0.5.
- `formula`: The formula used in the function.
- `coefs`: A named vector of coefficients of the estimated ANOVA model.
vcov  The variance-covariance matrix for coefs. Only when select=FALSE and collapse=FALSE.

CI.table  The summary of AMEs and AMIEs with confidence intervals. Only when select=FALSE and collapse=FALSE.

AME  The estimated AMEs with the grand-mean as baselines.

AMIE2  The estimated two-way AMIEs with the grand-mean as baselines.

AMIE3  The estimated three-way AMIEs with the grand-mean as baselines.

... arguments passed to the function or arguments only for the internal use.

Author(s)
Naoki Egami and Kosuke Imai.

References


See Also
cv.CausalANOVA

Examples

data(Carlson)
## Specify the order of each factor
Carlson$newRecordF<- factor(Carlson$newRecordF,ordered=TRUE,
    levels=c("YesLC", "YesDis","YesMP",
    "noLC","noDis","noMP","noBusi"))
Carlson$promise <- factor(Carlson$promise,ordered=TRUE,levels=c("jobs","clinic","education"))
Carlson$coeth_voting <- factor(Carlson$coeth_voting,ordered=FALSE,levels=c("0","1"))
Carlson$relevantdegree <- factor(Carlson$relevantdegree,ordered=FALSE,levels=c("0","1"))

## #######################################
## Without Screening and Collapsing
## #######################################

#################### only AMEs ####################
fit1 <- CausalANOVA(formula=won ~ newRecordF + promise + coeth_voting + relevantdegree,
    data=Carlson, pair.id=Carlson$contestresp, diff=TRUE,
    cluster=Carlson$respcodeS, nway=1)
summary(fit1)
plot(fit1)

#################### AMEs and two-way AMIEs ####################
fit2 <- CausalANOVA(formula=won ~ newRecordF + promise + coeth_voting + relevantdegree, 
                   int2.formula = ~ newRecordF:coeth_voting, 
                   data=Carlson, pair.id=Carlson$contestresp,diff=TRUE, 
                   cluster=Carlson$respcodeS, nway=2)
summary(fit2)
plot(fit2, type="ConditionalEffect", fac.name=c("newRecordF","coeth_voting"))
ConditionalEffect(fit2, treat.fac="newRecordF", cond.fac="coeth_voting")

## Not run:
############### AMEs and two-way and three-way AMIEs ###############
## Note: All pairs within thee-way interactions should show up in int2.formula (Strong Hierarchy).
fit3 <- CausalANOVA(formula=won ~ newRecordF + promise + coeth_voting + relevantdegree, 
                   int2.formula = ~ newRecordF:promise + newRecordF:coeth_voting + 
                              promise:coeth_voting, 
                   int3.formula = ~ newRecordF:promise:coeth_voting, 
                   data=Carlson, pair.id=Carlson$contestresp,diff=TRUE, 
                   cluster=Carlson$respcodeS, nway=3)
summary(fit3)
plot(fit3, type="AMIE", fac.name=c("newRecordF","promise", "coeth_voting"),space=25,adj.p=2.2)

## End(Not run)

############### AMEs and two-way AMIEs ###############
## With Screening and Collapsing
## Sample Splitting
train.ind <- sample(unique(Carlson$respcodeS), 272, replace=FALSE)
test.ind <- setdiff(unique(Carlson$respcodeS), train.ind)
Carlson.train <- Carlson[is.element(Carlson$respcodeS,train.ind), ]
Carlson.test <- Carlson[is.element(Carlson$respcodeS,test.ind), ]

fit.r2 <- CausalANOVA(formula=won ~ newRecordF + promise + coeth_voting + relevantdegree, 
                      data=Carlson.train, pair.id=Carlson.train$contestresp,diff=TRUE, 
                      screen=TRUE, collapse=TRUE, 
                      cluster=Carlson.train$respcodeS, nway=2)
summary(fit.r2)

## refit with test.CausalANOVA
fit.r2.new <- test.CausalANOVA(fit.r2, newdata=Carlson.test, diff=TRUE, 
                               pair.id=Carlson.test$contestresp, cluster=Carlson.test$respcodeS)
summary(fit.r2.new)
plot(fit.r2.new)
plot(fit.r2.new, type="ConditionalEffect", fac.name=c("newRecordF","coeth_voting"))
ConditionalEffect(fit.r2.new, treat.fac="newRecordF", cond.fac="coeth_voting")
**ConditionalEffect**

**Description**

ConditionalEffect estimates a variety of conditional effects using the output from CausalANOVA.

**Usage**

```r
ConditionalEffect(
  object, 
  treat.fac = NULL, 
  cond.fac = NULL, 
  base.ind = 1, 
  round = 3, 
  inference = NULL, 
  verbose = TRUE
)
```

**Arguments**

- `object`: The output from CausalANOVA function.
- `treat.fac`: The name of factor acting as the main treatment variable.
- `cond.fac`: The name of factor acting as the conditioning (moderating) variable.
- `base.ind`: An indicator for the baseline of the treatment factor. Default is 1.
- `round`: Digits to round estimates. Default is 3.
- `inference`: (optional). This argument is mainly for internal use. It indicates whether CausalANOVA has done inference or not.
- `verbose`: Whether it prints the progress.

**Details**

See Details in CausalANOVA.

**Value**

- `ConditionalEffects`: The summary of estimated conditional effects.
- ... Arguments for the internal use.

**Author(s)**

Naoki Egami and Kosuke Imai.

**References**


See Also

CausalANOVA.

Examples

data(Carlson)
## Specify the order of each factor
Carlson$newRecordF <- factor(Carlson$newRecordF, ordered=TRUE, levels=c("YesLC", "YesDis", "YesMP", "noLC", "noDis", "noMP", "noBusi"))
Carlson$promise <- factor(Carlson$promise, ordered=TRUE, levels=c("jobs", "clinic", "education"))
Carlson$coeth_voting <- factor(Carlson$coeth_voting, ordered=FALSE, levels=c("0", "1"))
Carlson$relevantdegree <- factor(Carlson$relevantdegree, ordered=FALSE, levels=c("0", "1"))

## Without Screening and Collapsing
## #############################################################################
## AMEs and two-way AMIEs #############################################################################
fit2 <- CausalANOVA(formula=won ~ newRecordF + promise + coeth_voting + relevantdegree, int2.formula = ~ newRecordF:coeth_voting, data=Carlson, pair.id=Carlson$contestresp, diff=TRUE, cluster=Carlson$respcodeS, nway=2)
summary(fit2)
plot(fit2, type="ConditionalEffect", fac.name=c("newRecordF","coeth_voting"))
ConditionalEffect(fit2, treat.fac="newRecordF", cond.fac="coeth_voting")

---

cv.CausalANOVA

Cross validation for the CausalANOVA.

Description

cv.CausalANOVA implements cross-validation for CausalANOVA to select the collapse.cost parameter. CausalANOVA runs this function internally when defaults when collapse.type=cv.min or collapse.type=cv.1Std.

Usage

cv.CausalANOVA(
  formula,
  int2.formula = NULL,
  int3.formula = NULL,
  data,
  nway = 1,
  pair.id = NULL,
diff = FALSE,
cv.collapse.cost = c(0.1, 0.3, 0.7),
nfolds = 5,
screen = FALSE,
screen.type = "fixed",
screen.num.int = 3,
family = "binomial",
cluster = NULL,
maxIter = 50,
eps = 1e-05,
seed = 1234,
fac.level = NULL,
ord.fac = NULL,
verbose = TRUE
)

Arguments

formula a formula that specifies outcome and treatment variables.
int2.formula (optional). A formula that specifies two-way interactions.
int3.formula (optional). A formula that specifies three-way interactions.
data an optional data frame, list or environment (or object coercible by 'as.data.frame' to a data frame) containing the variables in the model. If not found in 'data', the variables are taken from 'environment(formula)', typically the environment from which 'CausalANOV A' is called.
nway With nway=1, the function estimates the Average Marginal Effects (AMEs) only. With nway=2, the function estimates the AMEs and the two-way Average Marginal Interaction Effects (AMIEs). With nway=3, the function estimates the AMEs, the two-way and three-way AMIEs. Default is 1.
pair.id (optional). Unique identifiers for each pair of comparison. This option is used when diff=TRUE.
diff A logical indicating whether the outcome is the choice between a pair. If diff=TRUE, pair.id should specify a pair of comparison. Default is FALSE.
cv.collapse.cost A vector containing candidates for a cost parameter ranging from 0 to 1. 1 corresponds to no regularization and the smaller value corresponds to the stronger regularization. Default is c(0.1, 0.3, 0.7).
nfolds number of folds - default is 5. Although nfolds can be as large as the sample size (leave-one-out CV), it is not recommended for large datasets.
screen A logical indicating whether select significant factor interactions with glinternet. When users specify interactions using int2.formula or int3.formula, this option is ignored. screen should be used only when users want data-driven selection of factor-interactions. With screen.type, users can specify how to screen factor interactions. We recommend to use this option when the number of factors is large, e.g., more than 6. Default is FALSE.
screen.type: Type for screening factor interactions. (1) "fixed" select the fixed number (specified by screen.num.int) of factor interactions. (2) "cv.min" selects factor-interactions with the tuning parameter giving the minimum cross-validation error. (3) "cv.1Std" selects factor-interactions with the tuning parameter giving a cross-validation error that is within 1 standard deviation of the minimum cv error.

screen.num.int: (optional). The number of factor interactions to select. This option is used when and screen=TRUE and screen.type="fixed". Default is 3.

family: A family of outcome variables. "gaussian" when continuous outcomes "binomial" when binary outcomes. Default is "binomial".

cluster: Unique identifies with which cluster standard errors are computed.

maxIter: The number of maximum iteration for glinternet.

eps: A tolerance parameter in the internal optimization algorithm.

seed: an argument for set.seed().

fac.level: optional. A vector containing the number of levels in each factor. The order of fac.level should match to the order of columns in the data. For example, when the first and second columns of the design matrix is "Education" and "Race", the first and second element of fac.level should be the number of levels in "Education" and "Race", respectively.

ord.fac: optional. logical vectors indicating whether each factor has ordered (TRUE) or unordered (FALSE) levels. When levels are ordered, the function uses the order given by function levels(). If levels are ordered, the function places penalties on the differences between adjacent levels. If levels are unordered, the function places penalties on the differences based on every pairwise comparison.

verbose: whether it prints the value of a cost parameter used.

Details

See Details in CausalANOVA.

Value

cv.error: The mean cross-validated error - a vector of length length(cv.t).
cv.min: A value of t that gives minimum cv.missclass.
cv.1Std: The largest value of t such that error is within 1 standard error of the minimum.
cv.each.mat: A matrix containing cross-validation errors for each fold and cost parameter.
cv.cost: The cv.collapse.cost used in the function.

Author(s)

Naoki Egami and Kosuke Imai.
FindIt

References


See Also

CausalANOVA.

Examples

data(Carlson)
## Specify the order of each factor
Carlson$newRecordF<- factor(Carlson$newRecordF,ordered=TRUE,
  levels=c("YesLC", "YesDis","YesMP",
   "noLC","noDis","noMP","noBusi"))
Carlson$promise <- factor(Carlson$promise,ordered=TRUE,levels=c("jobs","clinic","education"))
Carlson$coeth_voting <- factor(Carlson$coeth_voting,ordered=FALSE,levels=c("0","1"))
Carlson$relevantdegree <- factor(Carlson$relevantdegree,ordered=FALSE,levels=c("0","1"))

## Collapsing Without Screening
## AMEs and two-way AMIs
# We show a very small example for illustration.
# Recommended to use cv.collapse.cost=c(0.1,0.3,0.5) and nfolds=10 in practice.
fit.cv <- cv.CausalANOVA(formula=won ~ newRecordF + promise + coeth_voting + relevantdegree,
  int2.formula = ~ newRecordF:coeth_voting,
  data=Carlson, pair.id=Carlson$contestresp,diff=TRUE,
  cv.collapse.cost=c(0.1,0.3), nfolds=2,
  cluster=Carlson$respcodeS, nway=2)
fit.cv

FindIt

FindIt for Estimating Heterogeneous Treatment Effects

Description

FindIt returns a model with the most predictive treatment-treatment interactions or treatment-covariate interactions.
Usage

FindIt(
  model.treat,  
  model.main,   
  model.int,    
  data = NULL,  
  type = "binary",  
  treat.type = "multiple",  
  nway,         
  search.lambdas = TRUE,  
  lambdas = NULL,  
  make.twoway = TRUE,  
  make.allway = TRUE,  
  wts = 1,        
  scale.c = 1,    
  scale.int = 1,  
  fit.glmnet = TRUE,  
  make.reference = TRUE,  
  reference.main = NULL,  
  threshold = 0.999999
)

Arguments

model.treat A formula that specifies outcome and treatment variables.
model.main An optional formula that specifies pre-treatment covariates to be adjusted.
model.int  A formula specifying pre-treatment covariates to be interacted with treatment assignments when treat.type="single".
data An optional data frame, list or environment (or object coercible by 'as.data.frame' to a data frame) containing the variables in the model. If not found in 'data', the variables are taken from 'environment(formula)', typically the environment from which 'FindIt' is called.
type "binary" for a binary outcome variable, which needs to be integer class; "continuous" for a continuous outcome variable.
treat.type "single" for interactions between a single treatment variable, which needs to be integer class; and multiple pre-treatment covariates specified with model.int; "multiple" is used when treatment-treatment interactions are of interest and treat is a matrix of multiple treatments.
nway An argument passed to makeallway when treat.type="multiple". FindIt generates treatment-treatment interactions up to the order specified with this argument. In general, it is recommended to use the number of factorial treatments. The current version covers up to four way interactions.
search.lambdas Whether to search for the tuning parameters for the LASSO constraints. If FALSE, lambdas must be supplied.
lambdas Tuning parameters to be given to FindIt; only used if search.lambdas=FALSE.
make.twoway If make.twoway=TRUE, all possible two-way interactions for the pre-treatment covariates specified in model.main and model.int are generated within FindIt. The default is set to be TRUE.

make.allway If make.allway=TRUE, all possible treatment-treatment interactions for multiple treatments are generated when treat.type="multiple". Interactions of the order up to the value of nway is computed.

wts An optional set of scaling weights. The default is 1.

scale.c A set of weights for recaling the pre-treatment covariates; only used if make.twoway=FALSE. maketwoway is useful for generating these.

scale.int A set of weights for recaling the covariates to be interacted with treatment variables; only used if make.twoway=FALSE. maketwoway is useful for generating these.

fit.glmnet Whether to fit using the coordinate descent method in glmnet (TRUE) or the regularization path method of LARS (FALSE).

make.reference Whether to make a reference matrix to check which columns are dropped when makeallway=TRUE.

reference.main If make.allway=FALSE and researchers generate a matrix of all possible interactions between factorial treatments, reference from makeallway function is better to be passed to FindIt through this argument.

threshold An argument passed to makeallway when treat.type="multiple". Threshold to drop correlated columns when makeallway is used.

Details

Implements the alternating line search algorithm for estimating the tuning parameters, as described in Imai and Ratkovic (2013).

Value

coeffs A named vector of scaled coefficients

coeffs.orig A vector of coefficients on the original scale, if scale.c and scale.t was used

fit Fitted values on an SVM scale

names.out Names of the coefficients

y A vector of observed outcomes

X.c A matrix of pre-treatment covariates to be adjusted

X.t A matrix of treatments and treatment-treatment interactions, or treatment-covariate interactions

GCV GCV statistic at the minimum

ATE When treat.type="single", the estimated ATE. When treat.type="multiple", the estimated treatment effect of each unique treatment combination

lambdas Tuning parameters used for the fit

reference When treat.type="multiple", after making all interaction terms, columns with no variation or columns perfectly correlated with one of other columns are automatically dropped. reference shows which columns are kept and dropped.
Author(s)
Naoki Egami, Marc Ratkovic and Kosuke Imai.

References


Examples

# Example 1: Treatment-Covariate Interaction

```r
# The model includes a treatment variable,  
# nine covariates to be interacted with the treatment variable,  
# and the same nine covariates to be adjusted.

# Not run:
# Run to find the LASSO parameters
F1 <- FindIt(model.treat = outcome ~ treat,  
             model.main = ~ age+educ+black+hisp+white+marr+nodegr+log.re75+u75,  
             model.int = ~ age+educ+black+hisp+white+marr+nodegr+log.re75+u75,  
             data = LaLonde,  
             type = "binary",  
             treat.type = "single")

# End(Not run)

# Fit with uncovered lambda parameters.
F1 <- FindIt(model.treat = outcome ~ treat,  
             model.main = ~ age+educ+black+hisp+white+marr+nodegr+log.re75+u75,  
             model.int = ~ age+educ+black+hisp+white+marr+nodegr+log.re75+u75,  
             data = LaLonde,  
             type = "binary",  
             treat.type = "single",  
             search.lambdas = FALSE,  
             lambdas = c(-3.8760,-4.0025) )
```
summary(F1)

## Returns all the estimated treatment effects.
pred1 <- predict(F1)
## Top 10
head(pred1$data, n=10)
## Bottom 10
tail(pred1$data, n=10)

## Visualize all the estimated treatment effects.
## Not run:
plot(pred1)

## End(Not run)

# Example 2: Treatment-Treatment Interaction
# Not run:
data(GerberGreen)

## The model includes four factorial treatments and
## all two, three, four-way interactions between them.
## Four pre-treatment covariates are adjusted.

## Run to search for lambdas.
F2 <- FindIt(model.treat= voted98 ~ persngrp+phnscrpt+mailings+appeal,
             nway=4,
             model.main= ~ age+majorpty+vote96.1+vote96.0,
             data = GerberGreen,
             type= "binary",
             treat.type= "multiple")

## Fit, given selected lambdas.
F2 <- FindIt(model.treat= voted98 ~ persngrp+phnscrpt+mailings+appeal,
             nway=4,
             model.main= ~ age+majorpty+vote96.1+vote96.0,
             data = GerberGreen,
             type= "binary",
             treat.type= "multiple",
             search.lambdas= FALSE,
             lambdas= c(-15.000,-6.237))

## Returns coefficient estimates.
summary(F2)

## Returns predicted values for unique treatment combinations.
pred2 <- predict(F2, unique= TRUE)
## Top 10
head(pred2$data, n=10)
## Bottom 10
tail(pred2$data, n=10)
## Visualize predicted values for each treatment combination.
plot(pred2)

## End(Not run)

---

### Data from the 1998 New Haven Get-Out-the-Vote Experiment

#### Description

This data set contains the most recent corrected data from the field experiment analyzed in Gerber and Green (2000).

#### Format

A data frame consisting of 9 columns and 29,380 observations.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>voted98</td>
<td>integer</td>
<td>voted in 1998</td>
</tr>
<tr>
<td>persngrp</td>
<td>factor</td>
<td>personal contact attempted</td>
</tr>
<tr>
<td>phnscrpt</td>
<td>factor</td>
<td>script read to phone respondents</td>
</tr>
<tr>
<td>mailings</td>
<td>factor</td>
<td>number of mailings sent</td>
</tr>
<tr>
<td>appeal</td>
<td>factor</td>
<td>content of message</td>
</tr>
<tr>
<td>age</td>
<td>integer</td>
<td>age of respondent</td>
</tr>
<tr>
<td>majorpty</td>
<td>factor</td>
<td>Democratic or Republican</td>
</tr>
<tr>
<td>voted96.1</td>
<td>factor</td>
<td>voted in 1996</td>
</tr>
<tr>
<td>voted96.0</td>
<td>factor</td>
<td>abstained in 1996</td>
</tr>
</tbody>
</table>

Note: The levels of phnscrpt and appeal are follows.

**phnscrpt**: Script read to phone respondents

0  No phone
1  Civic-Blood
2  Civic
3  Civic or Blood-Civic
4  Neighbor
5  Neighbor or Civic-Neighbor
6  Close

**appeal**: Content of message

1  Civic Duty
2  Neighborhood Solidarity
3  Close Election


References


LaLonde National Supported Work Study Experimental Data

Description

This data set gives the outcomes as well as treatment assignments and covariates for the National Supported Work Study, as analyzed in LaLonde (1986).

Format

A data frame consisting of 12 columns (including a treatment assignment vector) and 2787 observations.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>outcome</td>
<td>integer</td>
<td>whether earnings in 1978 are larger than in 1975</td>
<td>0,1</td>
</tr>
<tr>
<td>treat</td>
<td>integer</td>
<td>whether the individual received the treatment</td>
<td>0,1</td>
</tr>
<tr>
<td>age</td>
<td>numeric</td>
<td>age in years</td>
<td></td>
</tr>
<tr>
<td>educ</td>
<td>numeric</td>
<td>education in years</td>
<td></td>
</tr>
<tr>
<td>black</td>
<td>factor</td>
<td>black or not</td>
<td>0,1</td>
</tr>
<tr>
<td>hisp</td>
<td>factor</td>
<td>hispanic or not</td>
<td>0,1</td>
</tr>
<tr>
<td>white</td>
<td>factor</td>
<td>white or not</td>
<td>0,1</td>
</tr>
<tr>
<td>marr</td>
<td>factor</td>
<td>married or not</td>
<td>0,1</td>
</tr>
<tr>
<td>nodegr</td>
<td>factor</td>
<td>an indicator for no high school degree</td>
<td>0,1</td>
</tr>
<tr>
<td>log.re75</td>
<td>numeric</td>
<td>log of earnings in 1975</td>
<td></td>
</tr>
<tr>
<td>u75</td>
<td>factor</td>
<td>unemployed in 1975</td>
<td>0,1</td>
</tr>
<tr>
<td>wts.extrap</td>
<td>numeric</td>
<td>extrapolation weights to the 1978 Panel Study for Income Dynamics dataset</td>
<td></td>
</tr>
</tbody>
</table>

Source

Data from the National Supported Work Study. A benchmark matching dataset. 1975 earnings are pre-treatment.

References

plot.CausalANOVA  Plotting CausalANOVA

Description
Plotting CausalANOVA

Usage
## S3 method for class 'CausalANOVA'
plot(
x,  
fac.name,  
treat.ind = 1,  
type = "ConditionalEffect",  
space = 15,  
xlim,  
...)

Arguments

x An output from CausalANOVA

fac.name Factor names to plot. Length should be 2.

treat.ind Which factor serves as the main treatment. Should be 1 (the first element of fac.name) or 2 (the second element of fac.name).

type What types of effects to plot. Should be one of AME, AMIE and ConditionalEffect.

space Space on the left side of the plot.

xlim Range for the x-axis

... Other graphical parameters

plot.PredictFindIt  Plot estimated treatment effects or predicted outcomes for each treatment combination.

Description
Plot estimated treatment effects when treat.type="single" and predicted outcomes for each treatment combination when treat.type="multiple".

Usage
## S3 method for class 'PredictFindIt'
plot(x, main, xlab, ylab, interactive = FALSE, ...)

**Arguments**

- `x`: output from `predict.FindIt`.
- `main`: the argument specifying the main title of the plot.
- `xlab`: the argument specifying the name of x axis.
- `ylab`: the argument specifying the name of y axis.
- `interactive`: whether to make a plot interactive; default is FALSE.
- `...`: further arguments passed to or from other methods.

**Details**

Plot estimated treatment effects when `treat.type="single"` and predicted outcomes for each treatment combination when `treat.type="multiple"`.

**Value**

- `plot`: Plot estimated treatment effects when `treat.type="single"` and predicted outcomes for each treatment combination when `treat.type="multiple"`.

**Author(s)**

Naoki Egami, Marc Ratkovic and Kosuke Imai.

**Examples**

```r
## See the help page for FindIt() for an example.
```

---

**Description**

`predict.FindIt` takes an output from `FindIt` and returns estimated treatment effects when `treat.type="single"` and predicted outcomes for each treatment combination when `treat.type="multiple"`.

**Usage**

```r
## S3 method for class 'FindIt'
predict(
  object,
  newdata,
  sort = TRUE,
  decreasing = TRUE,
  wts = 1,
  unique = FALSE,
  ...
)
```
Arguments

- **object**: An output object from `FindIt`.
- **newdata**: An optional data frame in which to look for variables with which to predict. If omitted, the data used in `FindIt` is used.
- **sort**: Whether to sort samples according to estimated treatment effects.
- **decreasing**: When `sort=TRUE`, whether to sort the output in descending order or not.
- **wts**: Weights.
- **unique**: If `unique=TRUE`, `predict` returns estimated treatment effects or predicted outcomes for unique samples.
- **...**: Further arguments passed to or from other methods.

Details

Useful for computing estimated treatment effects or predicted outcomes for each treatment combination. By using `newdata`, researchers can compute them for any samples.

Value

- **data**: A matrix of estimated treatment effects when `treat.type="single"` and predicted outcomes for each treatment combination when `treat.type="multiple"`.

Author(s)

Naoki Egami, Marc Ratkovic and Kosuke Imai.

Examples

```r
## See the help page for `FindIt()` for an example.
```

```r
summary.CausalANOVA summary.CausalANOVA

Description

Summarizing CausalANOVA output

Usage

```r
## S3 method for class 'CausalANOVA'
summary(object, digit = 4, verbose = TRUE, verbose.full = TRUE, ...)
```
summary.FindIt

Arguments

object An object from CausalANOVA
digit the number of digits
verbose report additional summary results
verbose.full report full summary results
... Other parameters

Description

Summarizing FindIt output

Usage

## S3 method for class 'FindIt'
summary(object, ...)

Arguments

object An object from FindIt
... Other parameters

test.CausalANOVA Estimating the AMEs and AMIEs after Regularization with the CausalANOVA.

Description

test.CausalANOVA estimates the AMEs and AMIEs with confidence intervals after regularization with CausalANOVAfunction.

Usage

test.CausalANOVA(
  fit,
  newdata,
  collapse.level = TRUE,
  diff = FALSE,
  pair.id = NULL,
  cluster = NULL
)
Arguments

- **fit**: The output from `CausalANOVA` function.
- **newdata**: A data frame to use for re-estimating the AMEs and AMIEs with confidence intervals.
- **collapse.level**: A logical indicating whether to collapse insignificant levels within factors as suggested by the `CausalANOVA` output users provide.
- **diff**: A logical indicating whether the outcome is the choice between a pair. If `diff=TRUE`, `pair.id` should specify a pair of comparison. Default is `FALSE`.
- **pair.id**: (optional). Unique identifiers for each pair of comparison. This option is used when `diff=TRUE`.
- **cluster**: Unique identifies with which cluster standard errors are computed.

Details

See Details in `CausalANOVA`.

Value

- **fit**: The output of class `CausalANOVA`.

Author(s)

Naoki Egami and Kosuke Imai.

References


See Also

- `CausalANOVA`.

Examples

```r
## With Screening and Collapsing
## data(Carlson)
## Specify the order of each factor
Carlson$newRecordF<- factor(Carlson$newRecordF, ordered=TRUE, levels=c("YesLC", "YesDis","YesMP", "noLC","noDis","noMP","noBusi"))
```
Carlson$promise <- factor(Carlson$promise, ordered=TRUE, levels=c("jobs","clinic","education"))
Carlson$coeth_voting <- factor(Carlson$coeth_voting, ordered=FALSE, levels=c("0","1"))
Carlson$relevantdegree <- factor(Carlson$relevantdegree, ordered=FALSE, levels=c("0","1"))

## Sample Splitting
train.ind <- sample(unique(Carlson$respcodeS), 272, replace=FALSE)
test.ind <- setdiff(unique(Carlson$respcodeS), train.ind)
Carlson.train <- Carlson[is.element(Carlson$respcodeS, train.ind),]
Carlson.test <- Carlson[is.element(Carlson$respcodeS, test.ind),]

#################### AMEs and two-way AMIEs ####################
fit.r2 <- CausalANOVA(formula=won ~ newRecordF + promise + coeth_voting + relevantdegree,
data=Carlson.train, pair.id=Carlson.train$contestresp, diff=TRUE,
screen=TRUE, collapse=TRUE,
cluster=Carlson.train$respcodeS, nway=2)
summary(fit.r2)

## refit with test.CausalANOVA
fit.r2.new <- test.CausalANOVA(fit.r2, newdata=Carlson.test, diff=TRUE,
pair.id=Carlson.test$contestresp, cluster=Carlson.test$respcodeS)
summary(fit.r2.new)
plot(fit.r2.new)
plot(fit.r2.new, type="ConditionalEffect", fac.name=c("newRecordF","coeth_voting"))
ConditionalEffect(fit.r2.new, treat.fac="newRecordF", cond.fac="coeth_voting")
Index

*Topic datasets
Carlson, 3
GerberGreen, 18
LaLonde, 19

*Topic package
FindIt-package, 2

Carlson, 3
CausalANOVA, 3, 10, 13, 24
CausalANOVAfit(FindIt-package), 2
cluster_se_glm(FindIt-package), 2
CoefExtract(FindIt-package), 2
Collapsing(FindIt-package), 2
ConditionalEffect, 8
CreateANOVAconst(FindIt-package), 2
CreatelevelIndex(FindIt-package), 2
CreateWeights(FindIt-package), 2
cv.CausalANOVA, 7, 10

FindIt, 13
FindIt-package, 2
FullVCOV(FindIt-package), 2

GerberGreen, 18
glinternet.cv(FindIt-package), 2
Glsei(FindIt-package), 2

indTwo2Three(FindIt-package), 2

LaLonde, 19
Lcombinefunction(FindIt-package), 2
lengthSlack(FindIt-package), 2

makeallway(FindIt-package), 2
maketwowway(FindIt-package), 2

NoRegularization(FindIt-package), 2

plot.CausalANOVA, 20
plot.cv.CausalANOVA(cv.CausalANOVA), 10
plot.PredictFindIt, 20
predict.FindIt, 21
PsyConstraintCombine(FindIt-package), 2
rangeCausalANOVAfit(FindIt-package), 2
ScreenINT(FindIt-package), 2
stab.CausalANOVA(FindIt-package), 2
summary.CausalANOVA, 22
summary.FindIt, 23
test.CausalANOVA, 23
Zcombinefunction(FindIt-package), 2