Package 'DNetCausalPATT'

May 22, 2024

Type Package

Title Estimation of Conditional Average Treatment Effects (CATE) and Population Average Treatment Effects on the Treated (PATT)

Version 0.0.1

Maintainer Nguyen Huynh <nguyenhuynh831@gmail.com>

Description DNetCausalPATT is an R package that provides functions to estimate Conditional Average Treatment Effects (CATE) and Population Average Treatment Effects on the Treated (PATT) from experimental or observational data using the Super Learner (SL) ensemble method and Deep neural networks. The package first provides functions to implement meta-learners such as the Single-learner (S-learner) and Two-learner (T-learner) described in Künzel et al. (2019) for estimating the CATE. The S- and T-learner are each estimated using the SL ensemble method and deep neural networks. It then provides functions to implement the Ottoboni and Poulos (2020) PATT-C estimator to obtain the PATT from experimental data with noncompliance by using the SL ensemble method and deep neural networks.

License GPL-3

Encoding UTF-8

LazyData true

Imports ROCR, xgboost, SuperLearner, class, randomForest, glmnet, gam, e1071, gbm

Suggests testthat, ggplot2, tidyr

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

NeedsCompilation no

Author Nguyen K. Huynh [aut, cre] (<https://orcid.org/0000-0002-6234-7232>), Bumba Mukherjee [aut]

R topics documented:

`complier_mod` *Train complier model using ensemble methods*

Description

Train model using group exposed to treatment with compliance as binary outcome variable and covariates.

Usage

```
complier_mod(data, ID = NULL, SL.library = NULL)
```

Arguments

`data`	list object of experimental data.
`ID`	string for name of indentifier variable.
`SL.library.`	Employs extreme gradient boosting, elastic net regression, random forest, and neural nets.

Value

model object of trained model.

`complier_predict` *Complier model prediction*

Description

Predict Compliance from control group in experimental data

Usage

```
complier_predict(complier.mod, exp.data)
```

Arguments

`complier.mod`	output from trained ensemble superlearner model.
`exp.data`	experimental dataset

Value

data.frame object with true compliers, predicted compliers in the control group, and all compliers (actual + predicted).

---

neuralnet_complier_mod

Train compliance model using neural networks

Description

Train model using group exposed to treatment with compliance as binary outcome variable and covariates.

Usage

```
neuralnet_complier_mod(
  complier.formula,
  exp.data,
  treat.var,
  algorithm = "rprop+",
  hidden.layer = c(4, 2),
  ID = NULL,
  stepmax = 1e+05
)
```

Arguments

complier.formula	formula for complier variable as outcome and covariates (c ~ x)
exp.data	data.frame for experimental data.
treat.var	string for treatment variable.
algorithm	string for algorithm for training neural networks. Default set to the Resilient back propagation with weight backtracking (rprop+). Other algorithms include backprop', rprop-', 'sag', or 'slr' (see neuralnet package).
hidden.layer	vector for specifying hidden layers and number of neurons.
ID	string for identifier variable
stepmax	maximum number of steps.

Value

trained complier model object

neuralnet_pattc_counterfactuals *Assess Population Data counterfactuals*

Description

Create counterfactual datasets in the population for compliers and noncompliers. Then predict potential outcomes using trained model from `neuralnet_response_model`.

Usage

```
neuralnet_pattc_counterfactuals(
  pop.data,
  neuralnet.response.mod,
  ID = NULL,
  cluster = NULL
)
```

Arguments

`pop.data`	population data.
`neuralnet.response.mod`	trained model from. `neuralnet_response_model`.
`ID`	string for identifier variable.
`cluster`	string for clustering variable (currently unused).

Value

`data.frame` of predicted outcomes of response variable from counterfactuals.

neuralnet_predict *Predicting Compliance from experimental data*

Description

Predicting Compliance from control group experimental data

Usage

```
neuralnet_predict(neuralnet.complier.mod, exp.data, treat.var, compl.var)
```

Arguments

`neuralnet.complier.mod`	results from `neuralnet_complier_mod`
`exp.data`	`data.frame` of experimental data
`treat.var`	string for treatment variable
`compl.var`	string for compliance variable

Value

data.frame object with true compliers, predicted compliers in the control group, and all compliers (actual + predicted).

`neuralnet_response_model` *Modeling Responses from experimental data Using Deep NN*

Description

Model Responses from all compliers (actual + predicted) in experimental data using neural network.

Usage

```
neuralnet_response_model(
  response.formula,
  exp.data,
  neuralnet.compliers,
  compl.var,
  algorithm = "rprop+",
  hidden.layer = c(4, 2),
  stepmax = 1e+05
)
```

Arguments

response.formula	formula for response variable and covariates (y ~ x)
exp.data	data.frame of experimental data.
neuralnet.compliers	data.frame of compliers (actual + predicted) from neuralnet_predict.
compl.var	string of compliance variable
algorithm	neural network algorithm, default set to "rprop+".
hidden.layer	vector specifying hidden layers and number of neurons.
stepmax	maximum number of steps for training model.

Value

trained response model object

pattc_counterfactuals *Assess Population Data counterfactuals*

Description

Create counterfactual datasets in the population for compliers and noncompliers. Then predict potential outcomes from counterfactuals.

Usage

```
pattc_counterfactuals(
  pop.data,
  response.mod,
  ID = NULL,
  cluster = NULL,
  cut.point = 0.5
)
```

Arguments

pop.data	population dataset
response.mod	trained model from response_model.
cut.point	

patt_deep_nn *Estimate PATT_C using Deep NN*

Description

estimates the Population Average Treatment Effect of the Treated from experimental data with noncompliers using Deep Neural Networks.

Usage

```
patt_deep_nn(
  response.formula,
  exp.data,
  pop.data,
  treat.var,
  compl.var,
  compl.algorithm = "rprop+",
  response.algorithm = "rprop+",
  compl.hidden.layer = c(4, 2),
  response.hidden.layer = c(4, 2),
  compl.stepmax = 1e+05,
  response.stepmax = 1e+05,
  ID = NULL,
  cluster = NULL,
  bootse = FALSE,
```

```
  bootp = FALSE,
  bootn = 999
)
```

Arguments

response.formula	formula of response variable as outcome and covariates (y ~ x)
exp.data	data.frame of experimental data. Must include binary treatment and compliance variables.
pop.data	data.frame of population data. Must include binary compliance variable
treat.var	string for treatment variable.
compl.var	string for compliance variable
compl.algorithm	string for algorithim to train neural network for compliance model. Default set to "rprop+". See (neuralnet package for available algorithms).
response.algorithm	string for algorithim to train neural network for response model. Default set to "rprop+". See (neuralnet package for available algorithms).
compl.hidden.layer	vector for specifying hidden layers and number of neurons in complier model.
response.hidden.layer	vector for specifying hidden layers and number of neurons in response model.
compl.stepmax	maximum number of steps for complier model
response.stepmax	maximum number of steps for response model
ID	string for identifier variable
cluster	string for cluster variable.
bootse	logical for bootstrapped standard erros.
bootp	logical for bootstrapped p values.
bootn	logical for number of bootstraps.

Value

results of weighted t test as PATTC estimate.

---

patt_ensemble *PATT_C SL Ensemble*

Description

PATT_C_SL_Ensemble estimates the Population Average Treatment Effect of the Treated from experimental data with noncompliers using the super learner ensemble that includes extreme gradient boosting, glmnet (elastic net regression), random forest and neural nets.

Usage

```
patt_ensemble(
  response.formula,
  exp.data,
  pop.data,
  treat.var,
  compl.var,
  createSL = TRUE,
  ID = NULL,
  cluster = NULL,
  bootse = FALSE,
  bootp = FALSE,
  bootn = 999
)
```

Arguments

response.formula	formula for the effects of covariates on outcome variable (y ~ x).
exp.data	data.frame object for experimental data. Must include binary treatment and compliance variable.
pop.data	data.frame object for population data. Must include binary compliance variable.
treat.var	string for binary treatment variable.
compl.var	string for binary compliance variable.
createSL	logical. If TRUE will call on create.SL to create SL wrappers.
ID	string for name of identifier.
cluster	string for name of cluster variable.
bootse	logical for bootstrapped standard errors.
bootp	logical for bootstrapped p values.
bootn	number of bootstrap sample.

Value

results of weighted t test as PATTC estimate.

---

response_model *Response model from experimental data using SL ensemble*

Description

Train response model (response variable as outcome and covariates) from all compliers (actual + predicted) in experimental data using SL ensemble.

Usage

```
response_model(
  exp.data,
  exp.compliers,
  family = "binomial",
  ID = NULL,
  SL.library = NULL
)
```

Arguments

exp.data	experimental dataset.
exp.compliers	data.frame object of compliers from complier_predict.
family	string for "gaussian" or "binomial".
ID	string for identifier variable.
SL.library	vector of names of ML algorithms used for ensemble model.

Value

trained response model.

ST_learner_DeepNN *S_T-learner DeepNN*

Description

ST_learner_DeepNN implements the S-learner and T-learner for estimating CATE using Deep Neural Networks. The Resilient back propagation (Rprop) algorithm is used for training neural networks.

Usage

```
ST_learner_DeepNN(
  data,
  cov.formula,
  treat.var,
  meta.learner.type,
  stepmax = 1e+05,
  nfolds = 5,
  algorithm = "rprop+",
  hidden.layer = c(4, 2),
  linear.output = FALSE
)
```

Arguments

data	data.frame object of data.
cov.formula	formula description of the model y ~ x(list of covariates).
treat.var	string for the name of treatment variable.
meta.learner.type	string specifying is the S-learner and "T.Learner" for the T-learner model.
stepmax	maximum number of steps for training model.
nfolds	number of folds for cross-validation. Currently supports up to 5 folds.
algorithm	a string for the algorithm for the neural network. Default set to rprop+, the Resilient back propagation (Rprop) with weight backtracking algorithm for training neural networks.
hidden.layer	vector of integers specifying layers and number of neurons.
linear.output	logical specifying regression (TRUE) or classification (FALSE) model.

Value

vector of CATEs estimated by the meta learners for each observation.

ST_learner_ensemble *S_T-learner Ensemble*

Description

ST_learner_ensemble implements the S-learner and T-learner for estimating CATE using the super learner ensemble method. The super learner in this case includes the following machine learning algorithms: extreme gradient boosting, glmnet (elastic net regression), random forest and neural nets.

Usage

```
ST_learner_ensemble(
  data,
  cov.formula,
  treat.var,
  meta.learner.type,
  learners = c("SL.glmnet", "SL.xgboost", "SL.ranger", "SL.nnet"),
  nfolds = 5
)
```

Arguments

data	data.frame object of data
cov.formula	formula description of the model y ~ x(list of covariates)
treat.var	string for the name of treatment variable.
meta.learner.type	string specifying is the S-learner and "T.Learner" for the T-learner model.
learners	vector for super learner ensemble that includes extreme gradient boosting, glmnet, random forest, and neural nets.
nfolds	number of folds for cross-validation. Currently supports up to 5 folds.

Value

vector of CATEs estimated by the meta learners for each observation.

`WtC` *WtC*

Description

Weighted t-test to estimate PATT-C

Usage

```
WtC(
  x,
  y,
  c = NULL,
  weight = NULL,
  weighty = NULL,
  weightc = NULL,
  cluster = NULL,
  clustery = NULL,
  clusterc = NULL,
  samedata = TRUE,
  alternative = "two.tailed",
  mean1 = TRUE,
  bootse = TRUE,
  bootp = FALSE,
  bootn = 1000,
  drops = "pairwise",
  equivalence = FALSE
)
```

Arguments

`equivalence`

Index