Package ‘DeepLearningCausal’

July 30, 2024

Type Package
Title Causal Inference with Super Learner and Deep Neural Networks
Version 0.0.104
Maintainer Nguyen K. Huynh <khoinguyen.huynh@r.hit-u.ac.jp>
Description 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) <doi:10.1073/pnas.1804597116> 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) <doi:10.1515/jci-2018-0035> 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, caret, neuralnet, SuperLearner, class, xgboost, randomForest, glmnet, gam, e1071, gbm, Hmisc, weights
Suggests testthat, ggplot2, tidyr, dplyr
RoxygenNote 7.2.3
Depends R (>= 4.1.0)
NeedsCompilation no
Author Nguyen K. Huynh [aut, cre] (<https://orcid.org/0000-0002-6234-7232>), Bumba Mukherjee [aut] (<https://orcid.org/0000-0002-3453-601X>), Irvin (Chen-Yu) Lee [aut] (<https://orcid.org/0000-0004-5913-8925>)
Repository CRAN
Date/Publication 2024-07-30 00:10:02 UTC
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(  
  exp.data,  
  complier.formula,  
  treat.var,  
  ID = NULL,  
  SL.learners = c("SL.lmnet", "SL.xgboost", "SL.ranger", "SL.nnet", "SL.glm")
)
**complier_predict**

**Complier model prediction**

**Description**

Predict Compliance from control group in experimental data.

**Usage**

```r
complier_predict(complier.mod, exp.data, treat.var, compl.var)
```

**Arguments**

- `complier.mod`: output from trained ensemble superlearner model.
- `exp.data`: data.frame object of experimental dataset.
- `treat.var`: string specifying the binary treatment variable.
- `compl.var`: string specifying binary complier variable.

**Value**

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

**Arguments**

- `exp.data`: list object of experimental data.
- `complier.formula`: formula to fit compliance model (`c ~ x`) using complier variable and covariates.
- `treat.var`: string specifying the binary treatment variable.
- `ID`: string for name of identifier variable.
- `SL.learners`: vector of strings for ML classifier algorithms. If left NULL employs extreme gradient boosting, elastic net regression, random forest, and neural nets.

**Value**

model object of trained model.
exp_data

Survey Experiment of Support for Populist Policy

Description

Shortened version of survey response data that incorporates a vignette survey experiment. The vignette describes an international crisis between country A and B. After reading this vignette, respondents are randomly assigned to the control group or to one of two treatments: policy prescription to said crisis by strong (populist) leader and centrist (non-populist) leader. The respondents are then asked whether they are willing to support the policy decision to fight a war against country A, which is the dependent variable.

Usage

data(exp_data)

Format

exp_data:
A data frame with 257 rows and 12 columns:

female  Gender.
age     Age of participant.
income  Monthly household income.
religion Religious denomination
practicing_religion Importance of religion in life.
education Educational level of participant.
political_ideology Political ideology of participant.
employment Employment status of participant.
marital_status Marital status of participant.
job_loss  Concern about job loss.
strong_leader Binary treatment measure of leader type.
support_war  Binary outcome measure for willingness to fight war.

Source

Yadav and Mukherjee (2024)
Description

Extended experiment data with 514 observations

Usage

data(exp_data_full)

Format

exp_data_full:  
A data frame with 514 rows and 12 columns:  
female Gender.  
age Age of participant.  
income Monthly household income.  
religion Religious denomination  
practicing_religion Importance of religion in life.  
education Educational level of participant.  
political_ideology Political ideology of participant.  
employment Employment status of participant.  
marital_status Marital status of participant.  
job_loss Concern about job loss.  
strong_leader Binary treatment measure of leader type.  
support_war Binary outcome measure for willingness to fight war. #’ ...

Source

Yadav and Mukherjee (2024)

Description

metalearned_neural 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

```
metalearner_deepneural(
  data,
  cov.formula,
  treat.var,
  meta.learner.type,
  stepmax = 1e+05,
  nfolds = 5,
  algorithm = "rprop+",
  hidden.layer = c(4, 2),
  linear.output = FALSE,
  binary.outcome = 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.
binary.outcome logical specifying predicted outcome variable will take binary values or proportions.

Value

`metalearner_deepneural` of predicted outcome values and CATEs estimated by the meta learners for each observation.

Examples

```
# load dataset
data(exp_data)
# estimate CATEs with S Learner
set.seed(123456)
slearner.nn <- metalearner_deepneural(cov.formula = support_war ~ age + income + employed + job_loss,
  data = exp_data,
  treat.var = "strong_leader",
  meta.learner.type = "S.Learner",
  cov.formula = support_war ~ age + income + employed + job_loss,
  data = exp_data,
  treat.var = "strong_leader",
  meta.learner.type = "S.Learner",
  cov.formula = support_war ~ age + income + employed + job_loss,
  data = exp_data,
  treat.var = "strong_leader",
  meta.learner.type = "S.Learner",
  cov.formula = support_war ~ age + income + employed + job_loss,
  data = exp_data,
  treat.var = "strong_leader",
  meta.learner.type = "S.Learner",
  cov.formula = support_war ~ age + income + employed + job_loss,
  data = exp_data,
  treat.var = "strong_leader",
  meta.learner.type = "S.Learner",
  cov.formula = support_war ~ age + income + employed + job_loss,
  data = exp_data,
  treat.var = "strong_leader",
  meta.learner.type = "S.Learner",
  cov.formula = support_war ~ age + income + employed + job_loss,
  data = exp_data,
  treat.var = "strong_leader",
  meta.learner.type = "S.Learner",
  cov.formula = support_war ~ age + income + employed + job_loss,
  data = exp_data,
  treat.var = "strong_leader",
  meta.learner.type = "S.Learner",
  cov.formula = support_war ~ age + income + employed + job_loss,
  data = exp_data,
  treat.var = "strong_leader",
  meta.learner.type = "S.Learner",
  cov.formula = support_war ~ age + income + employed + job_loss,
  data = exp_data,
  treat.var = "strong_leader",
  meta.learner.type = "S.Learner",
  cov.formula = support_war ~ age + income + employed + job_loss,
  data = exp_data,
  treat.var = "strong_leader",
  meta.learner.type = "S.Learner",
  cov.formula = support_war ~ age + income + employed + job_loss,
  data = exp_data,
  treat.var = "strong_leader",
  meta.learner.type = "S.Learner",
  cov.formula = support_war ~ age + income + employed + job_loss,
  data = exp_data,
  treat.var = "strong_leader",
  meta.learner.type = "S.Learner",
  cov.formula = support_war ~ age + income + employed + job_loss,
  data = exp_data,
  treat.var = "strong_leader",
  meta.learner.type = "S.Learner",
  cov.formula = support_war ~ age + income + employed + job_loss,
  data = exp_data,
  treat.var = "strong_leader",
  meta.learner.type = "S.Learner",
  cov.formula = support_war ~ age + income + employed + job_loss,
  data = exp_data,
  treat.var = "strong_leader",
  meta.learner.type = "S.Learner",
  cov.formula = support_war ~ age + income + employed + job_loss,
  data = exp_data,
  treat.var = "strong_leader",
  meta.learner.type = "S.Learner",
  cov.formula = support_war ~ age + income + employed + job_loss,
  data = exp_data,
  treat.var = "strong_leader",
 meta.learner.type = "S.Learner",
```

metalearner_ensemble

Description

metalearner_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

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

SL.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.

binary.outcome: logical specifying predicted outcome variable will take binary values or proportions.

Value

metalearner_ensemble of predicted outcome values and CATEs estimated by the meta learners for each observation.

Examples

# load dataset
data(exp_data)

# load SuperLearner package
library(SuperLearner)

# estimate CATEs with S Learner
set.seed(123456)
slearner <- metalearner_ensemble(cov.formula = support_war ~ age + income + employed + job_loss,
data = exp_data,
treat.var = "strong_leader",
meta.learner.type = "S.Learner",
SL.learners = c("SL.glm"),
nfolds = 5,
binary.outcome = FALSE)

print(slearner)

# estimate CATEs with T Learner
set.seed(123456)
tlearner <- metalearner_ensemble(cov.formula = support_war ~ age + income + employed + job_loss,
data = exp_data,
treat.var = "strong_leader",
meta.learner.type = "T.Learner",
SL.learners = c("SL.xgboost","SL.ranger","SL.nnet"),
nfolds = 5,
binary.outcome = FALSE)

print(tlearner)
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+08
)

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,
  binary.outcome = FALSE
)

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).
- `binary.outcome`: logical specifying predicted outcome variable will take binary values or proportions.

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)
neuralnet_response_model

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).

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+08
)

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**

```r
pattc_counterfactuals(
  pop.data, 
  response.mod, 
  ID = NULL, 
  cluster = NULL, 
  binary.outcome = FALSE
)
```

**Arguments**

- `pop.data`: population dataset
- `response.mod`: trained model from `response_model`
- `ID`: string for identifier variable
- `cluster`: string for clustering variable
- `binary.outcome`: logical specifying whether predicted outcomes are proportions or binary (0-1).

**Value**

`data.frame` object of predicted outcomes of counterfactual groups.

---

**pattc_deepneural**  
**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

pattc_deepneural(
    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+08,
    response.stepmax = 1e+08,
    ID = NULL,
    cluster = NULL,
    binary.outcome = FALSE,
    bootstrap = FALSE,
    nboot = 1000
)

Arguments

response.formula
    formula of response variable as outcome and covariates \( y \sim x \)
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.
treat.var
    string for compliance variable
compl.algorithm
    string for algorithm to train neural network for compliance model. Default set to "rprop+". See (neuralnet package for available algorithms).
response.algorithm
    string for algorithm 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.
binary.outcome logical specifying predicted outcome variable will take binary values or proportions.

bootstrap logical for bootstrapped PATT-C.
nboot number of bootstrapped samples

Value

pattc_deepneural class object of results of t test as PATTC estimate.

Examples

```r
# load datasets
data(exp_data) # experimental data
data(pop_data) # population data
# specify models and estimate PATTC
set.seed(123456)
pattc_neural <- pattc_deepneural(response.formula = support_war ~ age + female +
income + education + employed + married +
hindu + job_loss,
exp.data = exp_data,
pop.data = pop_data,
treat.var = "strong_leader",
compl.var = "compliance",
compl.algorithm = "rprop+",
response.algorithm = "rprop+",
compl.hidden.layer = c(4,2),
response.hidden.layer = c(4,2),
compl.stepmax = 1e+09,
response.stepmax = 1e+09,
ID = NULL,
cluster = NULL,
binary.outcome = FALSE)

print(pattc_neural)

pattc_neural_boot <- pattc_deepneural(response.formula = support_war ~ age + female +
income + education + employed + married +
hindu + job_loss,
exp.data = exp_data,
pop.data = pop_data,
treat.var = "strong_leader",
compl.var = "compliance",
compl.algorithm = "rprop+",
response.algorithm = "rprop+",
compl.hidden.layer = c(4,2),
response.hidden.layer = c(4,2),
compl.stepmax = 1e+09,
response.stepmax = 1e+09,
ID = NULL,
cluster = NULL,
binary.outcome = FALSE,
bootstrap = TRUE,
```
pattc_ensemble

\[ \text{nboot} = 2000 \]

\[ \text{print(pattc_neural_boot)} \]

---

**pattc_ensemble**  
*PATT_C SL Ensemble*

**Description**

`pattc_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**

```r
pattc_ensemble(
  response.formula,
  exp.data,
  pop.data,
  treat.var,
  compl.var,
  compl.SL.learners = c("SL.glmnet", "SL.xgboost", "SL.ranger", "SL.nnet", "SL.glm"),
  response.SL.learners = c("SL.glmnet", "SL.xgboost", "SL.ranger", "SL.nnet", "SL.glm"),
  ID = NULL,
  cluster = NULL,
  binary.outcome = FALSE,
  bootstrap = FALSE,
  nboot = 1000
)
```

**Arguments**

- **response.formula**: formula for the effects of covariates on outcome variable \( y \sim x \).
- **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.
- **compl.SL.learners**: vector of names of ML algorithms used for compliance model.
- **response.SL.learners**: vector of names of ML algorithms used for response model.
- **ID**: ID variable.
- **cluster**: cluster variable.
- **binary.outcome**: binary outcome indicator.
- **bootstrap**: bootstrap indicator.
- **nboot**: number of bootstrap replicates.
response.SL.learners
   vector of names of ML algorithms used for response model.
ID
   string for name of identifier. (currently not used)
cluster
   string for name of cluster variable. (currently not used)
binary.outcome
   logical specifying predicted outcome variable will take binary values or proportions.
bootstrap
   logical for bootstrapped PATT-C.
nboot
   number of bootstrapped samples. Only used with bootstrap = FALSE

Value
   pattc_ensemble object of results of t test as PATTC estimate.

Examples
   # load datasets
data(exp_data_full) # full experimental data
data(exp_data) # experimental data
data(pop_data) # population data
# attach SuperLearner (model will not recognize learner if package is not loaded)
library(SuperLearner)
set.seed(123456)
# specify models and estimate PATTC
pattc <- pattc_ensemble(response.formula = support_war ~ age + income +
                       education + employed + job_loss,
                       exp.data = exp_data_full,
                       pop.data = pop_data,
                       treat.var = "strong_leader",
                       compl.var = "compliance",
                       compl.SL.learners = c("SL.glm", "SL.nnet"),
                       response.SL.learners = c("SL.glm", "SL.nnet"),
                       ID = NULL,
                       cluster = NULL,
                       binary.outcome = FALSE)

print(pattc)

pattc_boot <- pattc_ensemble(response.formula = support_war ~ age + income +
                           education + employed + job_loss,
                           exp.data = exp_data_full,
                           pop.data = pop_data,
                           treat.var = "strong_leader",
                           compl.var = "compliance",
                           compl.SL.learners = c("SL.glm", "SL.nnet"),
                           response.SL.learners = c("SL.glm", "SL.nnet"),
                           ID = NULL,
                           cluster = NULL,
                           binary.outcome = FALSE,
                           bootstrap = TRUE,
                           nboot = 1000)

print(pattc_boot)
**pop_data**

*World Value Survey India Sample*

**Description**

World Value Survey (WVS) Data for India in 2022. The variables drawn from the said WVS India data match the covariates from the India survey experiment sample.

**Usage**

data(pop_data)

**Format**

```r
pop_data:
A data frame with 846 rows and 13 columns:

female  Respondent’s Sex.
age Age of respondent.
income income group of Household.
religion Religious denomination
practicing_religion Importance of religion in respondent’s life.
education Educational level of respondent.
political_ideology Political ideology of respondent.
employment Employment status and full-time employee.
marital_status Marital status of respondent.
job_loss Concern about job loss.
support_war Binary (Yes/No) outcome measure for willingness to fight war.
strong_leader Binary measure of preference for strong leader. ... 
```

**Source**

Description


Usage

data(pop_data_full)

Format

pop_data_full:
A data frame with 11,813 rows and 13 columns:
  female  Respondent's Sex.
  age  Age of respondent.
  income  income group of Household.
  religion  Religious denomination
  practicing_religion  Importance of religion in respondent's life.
  education  Educational level of respondent.
  political_ideology  Political ideology of respondent.
  employment  Employment status and full-time employee.
  marital_status  Marital status of respondent.
  job_loss  Concern about job loss.
  support_war  Binary (Yes/No) outcome measure for willingness to fight war.
  strong_leader  Binary measure of preference for strong leader. ...

Source

print.metalearner_deepneural

Description
Print method for metalearner_deepneural

Usage
```r
## S3 method for class 'metalearner_deepneural'
print(x, ...)
```

Arguments
- `x` : metalearner_deepneural class object from metalearner_deepneural
- `...` : additional parameter

Value
list of model results

print.metalearner_ensemble

Description
Print method for metalearner_ensemble

Usage
```r
## S3 method for class 'metalearner_ensemble'
print(x, ...)
```

Arguments
- `x` : metalearner_ensemble class object from metalearner_ensemble
- `...` : additional parameter

Value
list of model results
### print.pattc_deepneural

#### Description
Print method for pattc_deepneural

#### Usage
```r
## S3 method for class 'pattc_deepneural'
print(x, ...)
```

#### Arguments
- **x**: pattc_deepneural class object from pattc_deepneural
- **...**: additional parameter

#### Value
list of model results

### print.pattc_ensemble

#### Description
Print method for pattc_ensemble

#### Usage
```r
## S3 method for class 'pattc_ensemble'
print(x, ...)
```

#### Arguments
- **x**: pattc_ensemble class object from pattc_ensemble
- **...**: additional parameter

#### Value
list of model results
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(
  response.formula,
  exp.data,
  compl.var,
  exp.compliers,
  family = "binomial",
  ID = NULL,
  SL.learners = c("SL.glmnet", "SL.xgboost", "SL.ranger", "SL.nnet", "SL.glm")
)

Arguments

response.formula
  formula to fit the response model (y ~ x) using binary outcome variable and covariates

exp.data
  experimental dataset.

cmpl.var
  string specifying binary complier variable

exp.compliers
  data.frame object of compliers from complier_predict.

family
  string for "gaussian" or "binomial".

ID
  string for identifier variable.

SL.learners
  vector of names of ML algorithms used for ensemble model.

Value

trained response model.
Index

* dataset
  exp_data, 4
  exp_data_full, 5
  pop_data, 17
  pop_data_full, 18

complier_mod, 2
complier_predict, 3

exp_data, 4
exp_data_full, 5

metalearnet_deepneural, 5
metalearnet_ensemble, 7

neuralnet_complier_mod, 9
neuralnet_pattc_counterfactuals, 10
neuralnet_predict, 10
neuralnet_response_model, 11

pattc_counterfactuals, 12
pattc_deepneural, 12
pattc_ensemble, 15
pop_data, 17
pop_data_full, 18
print.metalearnet_deepneural, 19
print.metalearnet_ensemble, 19
print.pattc_deepneural, 20
print.pattc_ensemble, 20

response_model, 21