Package: CERFIT (via r-universe)

October 30, 2026

Version 0.2.1

Title Causal Effect Random Forest of Interaction Trees

Description Fits a Causal Effect Random Forest of Interaction Trees (CERFIT) which is a modification of the Random Forest algorithm where each split is chosen to maximize subgroup treatment heterogeneity. Doing this allows it to estimate the individualized treatment effect for each observation in either randomized controlled trial (RCT) or observational data. For more information see L. Li, R. A. Levine, and J. Fan (2022) <doi:10.1002/sta4.457>.

License GPL (>= 2)

URL https://github.com/JustinThorp/CERFIT

BugReports https://github.com/JustinThorp/CERFIT/issues

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

LinkingTo Rcpp, RcppArmadillo

Imports partykit, CBPS, randomForest, twang, Rcpp, stats, glmnet, survival, grid

Depends R (>= 2.10)

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation yes

Author Justin Thorp [aut, cre], Joshua Moffat [aut], Luo Li [aut], Juanjuan Fan [aut]

Maintainer Justin Thorp <jjthorp@gmail.com>

Repository https://cran.r-universe.dev

Date/Publication 2026-10-30 14:20:30 UTC

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

RemoteRef HEAD

RemoteSha c371e6bcc5300f4ded0fd4c0c12216b2b15c2143

Contents

CERFIT 2
educational 5
MinDepth 6
predict.CERFIT 7
warts 8

Index **10**

CERFIT *Fits a Random Forest of Interaction Trees*

Description

Estimates individualized treatment effects (ITEs) using Random Forest of Interaction Trees. Works with randomized controlled trials (RCTs) and observational data. Treatment variables may be binary, categorical, ordered, or continuous. For binary and survival outcomes, useRes = TRUE must be specified.

Usage

```
CERFIT(
  formula,
  data,
  ntrees,
  subset = NULL,
  search = c("exhaustive", "sss"),
  method = c("RCT", "observational"),
  PropForm = c("randomForest", "CBPS", "GBM", "HI"),
  split = c("t.test"),
  mtry = NULL,
  nsplit = NULL,
  nsplit.random = FALSE,
  minsplit = 20,
  minbucket = round(minsplit/3),
  maxdepth = 30,
  oob = FALSE,
  a = 50,
  sampleMethod = c("bootstrap", "subsample", "subsampleByID", "allData"),
  useRes = TRUE,
  scale.y = FALSE,
  response = c("auto", "continuous", "binary", "survival"),
  surv_resid = c("deviance", "martingale"),
  surv_id = NULL
)
```

Arguments

formula	Formula to build CERFIT. Categorical predictors must be listed as a factor. e.g., Y ~ x1 + x2 | treatment. For survival outcomes, the response may be specified as Surv(time, event) or Surv(start, stop, event).
data	Data to grow a tree.
ntrees	Number of trees to grow. This value should not be too small, as observations may not be well represented and averaging is insufficient, leading to unstable results. A value of 1000 is often recommended, but it can be reduced for smaller datasets or to speed up computation.
subset	A logical vector that controls what observations are used to grow the forest. The default value will use the entire data frame.
search	Split search strategy. Options: "exhaustive" (evaluate all cut points) or "sss" (sigmoid approximation, experimental).
method	Study type. "RCT" for randomized data, "observational" for observational data.
PropForm	Method for estimating propensity scores (if method = "observational"). Options: "randomForest", "CBPS", "GBM", "HI". Not all options are compatible with all treatment types. See details.
split	Impurity measure splitting statistic. Currently supports "t.test".
mtry	Number of variables to consider at each split
nsplit	Number of candidate cut points. If NULL, all possible cut points are considered. If an integer is provided, that many cut points are randomly selected from the set of possible cut points.
nsplit.random	Default is FALSE. If TRUE, candidate cut points are chosen randomly.
minsplit	Number of observations required to continue growing tree.
minbucket	Number of observations required in each child node.
maxdepth	Maximum depth of tree.
oob	Logical, whether or not to use out-of-bag sample for predictions. Default is FALSE.
a	Sigmoid approximation variable (for "sss" which is still under development).
sampleMethod	Method to sample learning sample. Default is bootstrap. Subsample takes a subsample of the original data. SubsamplebyID samples by an ID column and uses all observations that have that ID. allData uses the entire data set for every tree.
useRes	Default is TRUE. If TRUE, fits the model using residuals from a regression of the response on covariates (excluding treatment). Improves stability and accuracy. Linear regression is used for continuous responses, logistic regression for binary responses, Cox proportional hazards model are used for survival responses. If response is binary or survival, useRes must be set equal to TRUE.
scale.y	Logical, standardize y when creating splits (For "sss" to increase stability).
response	Response type. Options are "auto", "continuous", "binary", and "survival". When response = auto, the function automatically detects a survival response if the formula uses Surv(...), detects a binary response if the outcome is stored as a

factor with exactly two levels, and otherwise treats the response as continuous. Thus, numeric continuous outcomes are recognized automatically, but binary outcomes coded numerically (for example, 0/1) are not automatically identified as binary and should be specified with response = binary if that behavior is desired.

`surv_resid` Residual type used for survival responses. Options are "deviance" and "martingale".

`surv_id` Optional subject ID column name for survival models with time-dependent covariates in counting-process format, e.g. Surv(start, stop, event). Required for that format.

Details

This function implements Random Forest of Interaction Trees proposed in Su (2018), which is a modification of the Random Forest algorithm where instead of a split being chosen to maximize prediction accuracy; each split is chosen to maximize subgroup treatment heterogeneity. It chooses the best split by maximizing the test statistic for $H_0 : \beta_3 = 0$ in the following linear model:

$Y_i = \beta_0 + \beta_1 I(X_{ij} < c) + \beta_2 I(Z = 1) + \beta_3 I(X_{ij} < c) I(Z = 1) + \varepsilon_i$

Where X_{ij} represents the splitting variable and Z = 1 represents treatment. So, by maximizing the test statistic for β_3 we are maximizing the treatment difference between the nodes.

The above equation only works when the data comes from a randomized controlled trial, but we can modify it to give us unbiased estimates of treatment effect in observational studies, as shown by Li et al. (2022). To do that we add propensity score into the linear model.

$Y_i = \beta_0 + \beta_1 I(X_{ij} < c) + \beta_2 I(Z = 1) + \beta_3 I(X_{ij} < c) I(Z = 1) + \beta_4 e_i + \varepsilon_i$

Where e_i represents the propensity score. The CERFIT function will estimate propensity score automatically when the method argument is set to observational.

To control how this function estimates propensity score, you can use the PropForm argument. Which can take four possible values randomForest, CBPS, GBM and HI. randomForest uses the randomForest package to use a random forest to estimate propensity score, CBPS uses Covariate Balancing Propensity Score to estimate propensity score, GBM uses generalized boosted regression models to estimate propensity score, and HI is the Hirano–Imbens generalized propensity score weighting for continuous treatments. Some of these options only work for certain treatment types. See the full list below.

- binary: GBM, CBPS, randomForest
- categorical: GBM, CBPS
- ordered: GBM, CBPS
- continuous: CBPS, HI

Note: CBPS option supports up to four categories/levels of treatment, if there are more than four use GBM.

Value

Returns a fitted CERFIT object which is a list containing the following elements:

- randFor: The Random Forest of Interaction Trees.

- trt.type: A string containing the treatment type of the data used to fit the model. Can be binary, multiple, ordered, or continuous.
- response.type: A string representing the response type of the data. Can be binary or continuous.
- useRes: A logical indicator that is TRUE if the model was fit on the residuals of a linear model.
- data: The data used to fit the model and includes the estimated propensity score if method was set to observational.

References

- Li, Luo, et al. Causal Effect Random Forest of Interaction Trees for Learning Individualized Treatment Regimes with Multiple Treatments in Observational Studies. Stat, 2022, https://doi.org/10.1002/sta4.457.
- Su, X., Peña, A., Liu, L., & Levine, R. (2018). Random forests of interaction trees for estimating individualized treatment effects in randomized trials. Statistics in Medicine, 37(17), 2547- 2560.
- G. W. Imbens, The role of the propensity score in estimating dose-response functions., Biometrika, 87 (2000), pp. 706–710.
- G. Ridgeway, D. McCarey, and A. Morral, The twang package: Toolkit for weighting and analysis of nonequivalent groups, (2006).
- A. Liaw and M. Wiener, Classification and regression by randomforest, R News, 2 (2002), pp. 18–22

Examples

```
fit <- CERFIT(Result_of_Treatment ~ sex + age + Number_of_Warts + Area + Time + Type | treatment,
data = warts,
ntrees = 30,
method = "RCT",
mtry = 2)
```

`educational` *Observational Educational Dataset*

Description

A simulated dataset containing the grades and other attributes of 1000 simulated students.

Usage

```
educational
```

Format

A data frame with 1000 rows and 7 variables:

SAT_MATH SAT Math Score

HSGPA High School GPA

AGE Age of Student

GENDER Gender of Student

URM Under Represented Minority

A Treatment Variable

Y Students Final Grade

Source

Wilke, Morten C., et al. "Estimating the Optimal Treatment Regime for Student Success Programs." Behaviormetrika, vol. 48, no. 2, 2021, pp. 309–343., https://doi.org/10.1007/s41237-021-00140-0.

`MinDepth` *Calculate Variable Importance*

Description

Calculates the average minimal depth of each predictor used to fit a CERFIT object. It calculates a variable's importance by using the variable's average minimal depth. Variables with a lower average minimal depth are more important.

Usage

```
MinDepth(cerfit)
```

Arguments

`cerfit`	A fitted CERFIT object

Details

The depth of the root node is zero and if a variable does not appear at any split in a tree it is assigned maxdepth + 1 for that tree.

Value

Returns a named vector with the name of each predictor used to fit the CERFIT object and its corresponding average minimal depth across all trees.

Examples

```
fit <- CERFIT(Result_of_Treatment ~ sex + age + Number_of_Warts + Area + Time + Type | treatment,
data = warts,
ntrees = 30,
method = "RCT",
mtry = 2)
importance <- MinDepth(fit)
```

`predict.CERFIT` *Get predictions from a CERFIT object*

Description

Get predictions from a CERFIT object

Usage

```
## S3 method for class 'CERFIT'
predict(
  object,
  newdata = NULL,
  gridval = NULL,
  prediction = c("overall", "by iter"),
  type = c("response", "ITE", "node", "opT"),
  alpha = 0.5,
  ...
)
```

Arguments

`object`	A fitted CERFIT object
`newdata`	New data to make predictions from. If not provided will make predictions on training data.
`gridval`	For continuous treatments with more than 15 unique values, the treatment values at which to predict. Must contain more than 10 values; shorter grids return an error. If `NULL`, the 0th, 5th, ..., 100th percentiles of the unique observed treatment values (21 values) are used. When the treatment has 15 or fewer unique values, `gridval` is ignored and predictions are made at each observed treatment value.
`prediction`	Return prediction using all trees ("overall") or using first i trees ("by iter").
`type`	Choose which value you wish to predict: 'response' will predict the potential outcome. 'ITE' will predict the individualized treatment effect. And 'opT' will predict the optimal treatment for each observation.
`alpha`	For continuous treatment. It is the mixing parameter for the elastic net regularization in each node. When equal to 0 it is ridge regression and when equal to 1 it is lasso regression.
`...`	Additional Arguments

Value

The return value depends on the prediction and type arguments.

With prediction = "overall": if type = "response", a matrix with n rows and one column per treatment level. If type = "ITE", a matrix with n rows and one column per contrast against the reference (lowest) treatment level, i.e. the number of levels minus one. If type = "opT", a matrix with n rows and two columns: the optimal treatment and the optimal response.

With prediction = "by iter": a named list of matrices, each with n rows and one column per tree. Column j is the estimate averaged over the first j trees, skipping trees that give no prediction for that observation, so the last column equals the "overall" result.

- type = "response": one matrix per treatment level (per gridval value for a continuous treatment), named "ycum<level>".
- type = "ITE": one matrix per contrast against the reference level, named "ycum<level>-ycum<reference>". A list is returned even for a binary treatment, where it has length one.

type = "opT" is not supported with prediction = "by iter".

Reproducibility with continuous treatments

For a continuous treatment (a numeric treatment with more than 10 distinct values), each terminal node fits a cubic dose-response curve with glmnet::cv.glmnet(), which assigns cross-validation folds at random. As a result, repeated calls to predict() on the same fitted object can return slightly different values, and separate calls with type = "response", "ITE", or "opT" will not exactly reconcile with one another. To make predictions reproducible, call set.seed() immediately before each predict() call, using the same seed for calls whose results will be compared. Alternatively, predict the responses once and compute the ITEs directly as resp[, -1] - resp[, 1], which is identical to type = "ITE".

Binary, multi-arm categorical, and ordered treatments, including numeric treatments with 10 or fewer distinct values, do not use this path; their predictions are deterministic, so a seed is needed only when fitting the model with CERFIT().

Examples

```
fit <- CERFIT(Result_of_Treatment ~ sex + age + Number_of_Warts + Area + Time + Type | treatment,
data = warts,
ntrees = 30,
method = "RCT",
mtry = 2)
ite <- predict(fit,type = "ITE")
```

---

warts *Randomized Controlled Trial Warts Dataset*

Description

A dataset comparing immunotherapy to cryotherapy treatments and their effectiveness of removing warts.

Usage

```
warts
```

Format

A data frame with 180 rows and 8 variables:

sex Patients Sex

age Patients Age

Time Time Elapsed Before Treatment

Number_of_Warts Number of Warts

Type Type of Wart

Area Wart Surface Area

Result_of_Treatment Treatment Outcome

treatment 0 for immunotherapy and 1 for cryotherapy

Source

Khozeimeh, Fahime, et al. "An Expert System for Selecting Wart Treatment Method." Computers in Biology and Medicine, vol. 81, 2017, pp. 167–175., https://doi.org/10.1016/j.compbiomed.2017.01.001.

Index

∗ **datasets**
    educational, 5
    warts, 8

CERFIT, 2

educational, 5

MinDepth, 6

predict.CERFIT, 7

warts, 8