Package ‘bartcs’

August 6, 2022

Title Bayesian Additive Regression Trees for Confounder Selection

Version 0.1.2

Description Fit Bayesian Regression Additive Trees (BART) models to select relevant confounders among a large set of potential confounders and to estimate average treatment effect. For more information, see Kim et al. (2022) <doi:10.48550/arXiv.2203.11798>.

License GPL (>= 3)

URL `https://github.com/yooyh/bartcs`

BugReports `https://github.com/yooyh/bartcs/issues`

Depends R (>= 2.10)

Imports ggcharts, ggplot2, invgamma, MCMCpack, Rcpp, rlang, rootSolve, stats

Suggests knitr, microbenchmark, rmarkdown

LinkingTo Rcpp

VignetteBuilder knitr

Encoding UTF-8

LazyData true

RoxygenNote 7.2.1

NeedsCompilation yes

Author Yeonghoon Yoo [aut, cre]

Maintainer Yeonghoon Yoo <`yooyh.stat@gmail.com`>

Repository CRAN

Date/Publication 2022-08-06 07:30:13 UTC

R topics documented:

`bartcs-package` *bartcs: Bayesian Additive Regression Trees for Confounder Selection*

Description

Fit Bayesian Regression Additive Trees (BART) models to select relevant confounders among a large set of potential confounders and to estimate average treatment effect. For more information, see Kim et al. (2022).

Details

Functions in `bartcs` serve one of three purposes.

1. Functions for fitting: `sbart()` and `mbart()`.
2. Functions for summary: `summary()`, `plot()` and `gelman_rubin()`.
3. Utility function for OpenMP: `count_omp_thread()`.

References

Kim, C., Tec, M., & Zigler, C. M. (2022). Bayesian Nonparametric Adjustment of Confounding. *arXiv preprint arXiv:2203.11798*. doi:10.48550/arXiv.2203.11798

`bart` *Fit BART models to select confounders and estimate treatment effect*

Description

Fit Bayesian Regression Additive Trees (BART) models to select relevant confounders among a large set of potential confounders and to estimate average treatment effect $(Y(1) - Y(0))$.

Usage

```
sbart(
  Y, trt, X,
  trt_treated     = 1,
  trt_control     = 0,
  num_tree        = 50,
  num_chain       = 4,
  num_burn_in     = 100,
  num_thin        = 0,
  num_post_sample = 100,
  step_prob       = c(0.28, 0.28, 0.44),
  alpha           = 0.95,
  beta            = 2,
  nu              = 3,
  q               = 0.95,
  dir_alpha       = 5,
  boot_size       = NULL,
  parallel        = NULL,
  verbose         = TRUE
)

mbart(
  Y, trt, X,
  trt_treated     = 1,
  trt_control     = 0,
  num_tree        = 50,
  num_chain       = 4,
  num_burn_in     = 100,
  num_thin        = 0,
  num_post_sample = 100,
  step_prob       = c(0.28, 0.28, 0.44),
  alpha           = 0.95,
  beta            = 2,
  nu              = 3,
  q               = 0.95,
  dir_alpha       = 5,
  boot_size       = NULL,
  parallel        = NULL,
  verbose         = TRUE
)
```

Arguments

`Y` Outcome variable.

`trt` Treatment variable.

`X` Potential confounders.

`trt_treated` Value of `trt` for treated group.

`trt_control` Value of `trt` for control group.

`num_tree` Number of trees in BART model.

`num_chain` Number of MCMC chains. Need to set `num_chain` > 1 for Gelman-Rubin diagnostic.

`num_burn_in` Number of MCMC samples to be discarded per chain as initial burn-in periods.

`num_thin` Number of thinning per chain. One in every `num_thin` samples are selected.

`num_post_sample`
: Final number of posterior samples per chain. Number of MCMC iterations per chain is `burn_in + num_thin * num_post_sample`.

`step_prob`
: A vector of tree alteration probabilities (GROW, PRUNE, CHANGE). Each alteration is proposed to change the tree structure. Default setting is `(0.28, 0.28, 0.44)`.

`alpha, beta`
: Hyperparameters for tree regularization prior. A terminal node of depth `d` will split with probability of `alpha * (1 + d)^(-beta)`.

`nu, q`
: Values to calibrate hyperparameter of sigma prior. Default setting is `(nu, q) = (3, 0.95)` from Chipman et al. (2010).

`dir_alpha`
: Hyperparameter of Dirichlet prior for selection probabilities.

`boot_size`
: Number of bootstrap sample size. Bootstrap samples will be used to compute potential outcomes $Y(1)$ and $Y(0)$.

`parallel`
: If `TRUE`, model fitting will be parallelized with respect to `n = nrow(X)`. Parallelization is recommended for very high `n` only.

`verbose`
: If `TRUE`, message will be printed during training. If `FALSE`, message will be suppressed.

Details

`sbart()` and `mbart()` fit an exposure model and outcome model(s) for estimating treatment effect with adjustment of confounders in the presence of a large set of potential confounders (Kim et al. 2022).

The exposure model $E[A|X]$ and the outcome model(s) $E[Y|A, X]$ are linked together with a common Dirichlet prior that accrues posterior selection probability to confounders (X) on the basis of association with both the exposure (A) and the outcome (Y).

There is a distinction between fitting each outcome model for the treated and control groups and fitting a single outcome model for the entire sample.

- `sbart()` specifies two **"separate"** outcome models for two binary treatment levels. Thus, it fits three models: one exposure model and two separate outcome models for $A = 0, 1$.
- `mbart()` specifies a single **"marginal"** outcome models. Thus, it fits two models: one exposure model and one outcome model for the entire sample.

All inferences are made with outcome model(s).

Value

A `bartcs` object. A `bartcs` object is a list with following components.

`ATE`
: Aggregated posterior samples of average treatment effect $(Y(1) - Y(0))$.

`Y1`
: Aggregated posterior samples of potential outcome $Y(1)$.

`Y0`
: Aggregated posterior samples of potential outcome $Y(0)$.

`var_prob`
: Aggregated posterior inclusion probability of each variable.

`chains`
: A list of results from each MCMC chain. Each list element consists of followings.

- ATE Posterior sample of average treatment effect $(Y(1) - Y(0))$.
- Y1 Posterior sample of potential outcome $Y(1)$.
- Y0 Posterior sample of potential outcome $Y(0)$.
- var_prob Posterior inclusion probability of each variable.
- var_count Number of selection of each variable in each MCMC iteration. Its dimension is num_post_sample * ncol(X).
- sigma2_out Posterior sample of sigma2 in the outcome model.
- dir_alpha Posterior sample of dir_alpha.

model	sbart or mbart.
label	Column names of X.
params	Parameters used in the model.

References

Chipman, H. A., George, E. I., & McCulloch, R. E. (2010). BART: Bayesian additive regression trees. *The Annals of Applied Statistics, 4*(1), 266-298. doi:10.1214/09AOAS285

Kim, C., Tec, M., & Zigler, C. M. (2022). Bayesian Nonparametric Adjustment of Confounding. *arXiv preprint arXiv:2203.11798*. doi:10.48550/arXiv.2203.11798

Examples

```
data(ihdp, package = "bartcs")
mbart(
  Y               = ihdp$y_factual,
  trt             = ihdp$treatment,
  X               = ihdp[, 6:30],
  num_tree        = 10,
  num_chain       = 2,
  num_post_sample = 20,
  num_burn_in     = 10,
  verbose         = FALSE
)
sbart(
  Y               = ihdp$y_factual,
  trt             = ihdp$treatment,
  X               = ihdp[, 6:30],
  num_tree        = 10,
  num_chain       = 2,
  num_post_sample = 20,
  num_burn_in     = 10,
  verbose         = FALSE
)
```

count_omp_thread *Count the number of OpenMP threads for parallel computation*

Description

`count_omp_thread()` counts the number of OpenMP threads for parallel computation. If it returns 1, OpenMP is not viable.

Usage

```
count_omp_thread()
```

Value

Number of OpenMP thread(s).

Examples

```
count_omp_thread()
```

gelman_rubin *Gelman-Rubin diagnostic for* `bartcs` *objects.*

Description

`gelman_rubin()` computes Gelman-Rubin diagnostic for `bartcs` objects.

Usage

```
gelman_rubin(x)
```

Arguments

`x`	A `bartcs` object.

Value

Gelman-Rubin diagnostic value.

Examples

```
data(ihdp, package = "bartcs")
x <- mbart(
  Y               = ihdp$y_factual,
  trt             = ihdp$treatment,
  X               = ihdp[, 6:30],
  num_tree        = 10,
  num_chain       = 2,
  num_post_sample = 20,
  num_burn_in     = 10,
  verbose         = FALSE
)

gelman_rubin(x)
```

ihdp *Infant Health and Development Program Data*

Description

Infant Health and Development Program (IHDP) is a randomized experiment from 1985 to 1988 which studied the effect of home visits on cognitive test scores for infants.

Usage

```
ihdp
```

Format

treatment Given treatment.

y_factual Observed outcome.

y_cfactual Potential outcome given the opposite treatment.

mu0 Control conditional means.

mu1 Treated conditional means.

X1 ~ X6 Confounders with continuous values.

X7 ~ X25 Confounders with binary values.

Details

This dataset was first used by Hill (2011), then used by other researchers (Shalit et al. 2017, Louizos et al. 2017).

Source

Our version of dataset is the dataset used by Louizos et al. (2017). This is the first realization of 10 generated datasets and you can find other realizations from https://github.com/AMLab-Amsterdam/CEVAE.

References

Hill, J. L. (2011). Bayesian nonparametric modeling for causal inference. *Journal of Computational and Graphical Statistics, 20*(1), 217-240. doi:10.1198/jcgs.2010.08162

Louizos, C., Shalit, U., Mooij, J. M., Sontag, D., Zemel, R., & Welling, M. (2017). Causal effect inference with deep latent-variable models. *Advances in neural information processing systems, 30*. doi:10.48550/arXiv.1705.08821 `https://github.com/AMLab-Amsterdam/CEVAE`

Shalit, U., Johansson, F. D., & Sontag, D. (2017, July). Estimating individual treatment effect: generalization bounds and algorithms. In *International Conference on Machine Learning* (pp. 3076-3085). PMLR. doi:10.48550/arXiv.1606.03976

`plot.bartcs` *Draw plot for* `bartcs` *object*

Description

Two options are available: posterior inclusion probability (pip) plot and trace plot.

Usage

```
## S3 method for class 'bartcs'
plot(x, method = NULL, parameter = NULL, ...)
```

Arguments

`x`	A `bartcs` object.
`method`	`"pip"` for posterior inclusion probability plot or `"trace"` for trace plot.
`parameter`	Target of parameter for traceplot.
`...`	Additional arguments for pip plot. Check `?ggcharts::bar_chart` for possible arguments.

Details

PIP plot:
When a posterior sample is sampled during training, `sbart()` or `mbart()` also counts which variables are included in the model and compute pip for each variable. For `bartcs` object `x`, this is stored in `x$var_count` and `x$var_prob` respectively. `plot(method = "pip")` uses this information and draws plot using `ggcharts::bar_chart()`.

Traceplot:
Parameters are recorded for each MCMC iterations. Parameters include `"ATE"`, `"Y1"`, `"Y0"`, `"dir_alpha"`, and either `"sigma2_out"` from `mbart()` or `"sigma2_out1"` and `"sigma2_out0"` from `sbart()`. Vertical line indicates burn-in.

Value

A `ggplot` object of either pip plot or trace plot.

Examples

```
data(ihdp, package = "bartcs")
x <- mbart(
  Y               = ihdp$y_factual,
  trt             = ihdp$treatment,
  X               = ihdp[, 6:30],
  num_tree        = 10,
  num_chain       = 2,
  num_post_sample = 20,
  num_burn_in     = 10,
  verbose         = FALSE
)

# pip plot
plot(x, method = "pip")
plot(x, method = "pip", top_n = 10)
plot(x, method = "pip", threshold = 0.5)
# Check `?ggcharts::bar_chart` for other possible arguments.

# trace plot
plot(x, method = "trace")
plot(x, method = "trace", "Y1")
plot(x, method = "trace", "dir_alpha")
```

---

summary.bartcs *Summary for* bartcs *object*

Description

Provide summary for bartcs object.

Usage

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

Arguments

object A bartcs object.

... Additional arguments. Not yet supported.

Details

summary() computes Gelman-Rubin diagnostic and 95% posterior credible interval for both aggregated outcome and individual outcomes from each chain.

Value

Provide list with following components.

`model` `sbart` or `mbart`.

`trt_value` Treatment values for each treatment group: `trt_treated` for treatment group and `trt_control` for control group.

`tree_params` Parameters used for tree structure.

`chain_params` Parameters used for MCMC chains.

`gelman_rubin` Gelman-Rubin diagnostic value.

`outcome` Summary of outcomes from the model. This includes both aggregated outcome and individual outcomes from each chain.

Examples

```
data(ihdp, package = "bartcs")
x <- mbart(
  Y               = ihdp$y_factual,
  trt             = ihdp$treatment,
  X               = ihdp[, 6:30],
  num_tree        = 10,
  num_chain       = 2,
  num_post_sample = 20,
  num_burn_in     = 10,
  verbose         = FALSE
)
summary(x)
```

Index