This form documents the artifacts associated with the article (i.e., the data and code supporting the computational findings) and describes how to reproduce the findings.

Part 1: Data

- ☐ This paper does not involve analysis of external data (i.e., no data are used or the only data are generated by the authors via simulation in their code).
- ☒ I certify that the author(s) of the manuscript have legitimate access to and permission to use the data used in this manuscript.

Abstract

We consider both simulated and real-world data sets. The simulation studies include continuous, count, and classification responses generated on 2-D U-shaped and 3-D torus domains, as well as Euclidean settings based on the Friedman function involving only unstructured features. The real data analyses involve spatial and network-structured data sets with diverse response types, including income, housing prices, voting behavior, cancer incidence counts, and binary indicators of air quality and flooding, using geographic coordinates or county adjacency graphs as structured features alongside numerical covariates.

Availability

- ☒ Data **are** publicly available.
- ☐ Data **cannot be made** publicly available.

If the data are publicly available, see the *Publicly available data* section. Otherwise, see the *Non-publicly available data* section, below.

Publicly available data

- ☐ Data are available online at:
- ☒ Data are available as part of the paper's supplementary material.
- ☐ Data are publicly available by request, following the process described here:
- ☐ Data are or will be made available through some other mechanism, described here:

Non-publicly available data

Description

File format(s)

- ☒ CSV or other plain text.
- ☒ Software-specific binary format (.Rda, Python pickle, etc.): .RData
- ☐ Standardized binary format (e.g., netCDF, HDF5, etc.):
- ☐ Other (please specify):

Data dictionary

- ☒ Provided by authors in the following file(s): `data/README.md`
- ☐ Data file(s) is(are) self-describing (e.g., netCDF files)
- ☐ Available at the following URL:

Additional Information (optional)

Part 2: Code

Abstract

The model implementation and data analysis in the paper were done in R and C++. The `R/` folder provides code for 1) demonstrating the generation of GS-BART input data under different data settings ,2) fitting GS-BART to simulated and real-world data using Informed Importance Tempering (IIT) method, 3) Bayesian predictive inference of GS-BART, and 4) reproducing key figures and tables in the paper.

Description

Code format(s)

- ☒ Script files
  - ☒ R
  - ☐ Python
  - ☐ Matlab
  - ☐ Other:
- ☒ Package
  - ☒ R
  - ☐ Python
  - ☐ MATLAB toolbox
  - ☐ Other:
- ☒ Reproducible report
  - ☒ R Markdown
  - ☐ Jupyter notebook
  - ☐ Other:
- ☐ Shell script
- ☐ Other (please specify):

Supporting software requirements

Version of primary software used R version 4.4.1.

Libraries and dependencies used by the code Required R packages and their version numbers are listed in `dependencies.txt`.

GS-BART and Generalized GS-BART are implemented in R and C++. The R package source code is provided in the `R packages/` directory.

Supporting system/hardware requirements (optional)

Parallelization used

- ☐ No parallel code used
- ☒ Multi-core parallelization on a single machine/node
  - Number of cores used: 5
- ☐ Multi-machine/multi-node parallelization
  - Number of nodes and cores used:

License

- ☒ MIT License (default)
- ☐ BSD
- ☐ GPL v3.0

☐ Creative Commons
☐ Other: (please specify)

Additional information (optional)

The code provided in this repository consists of a main R Markdown file `reproduce.Rmd`, which calls R scripts in `R/` to reproduce key results in the paper.

The R code in `R/` is organized as follows:

- Code for simulation studies:
  - `sim_continuous_fit_model.R`: Implements the GS-BART model for continuous-response simulation data.
  - `sim_count_fit_model.R`: Implements the GS-BART model for count-response simulation data.
  - `sim_classification_fit_model.R`: Implements the GS-BART model for classification simulation data.
  - `GSBART_sim_results.R`: Reproduces the key figures and tables from the simulation studies.
- Code for real data analysis:
  - `real_continuous_fit_model.R`: Implements the GS-BART model for continuous real-world data.
  - `real_count_fit_model.R`: Implements the GS-BART model for count real-world data.
  - `real_classification_fit_model.R`: Implements the GS-BART model for classification real-world data.
  - `GSBART_real_results.R`: Reproduces the key figures and tables from the real data analysis.
- Generation of GS-BART Input Data:
  - `graph_gen_fun.R`: Provides functions for generating GS-BART input data.
  - `graph_gen_demo.R`: Demonstrates how to generate GS-BART input data under different data settings.

Part 3: Reproducibility workflow

Scope

The provided workflow reproduces:

☐ Any numbers provided in text in the paper
☐ The computational method(s) presented in the paper (i.e., code is provided that implements the method(s))
☐ All tables and figures in the paper
☒ Selected tables and figures in the paper, as explained and justified below: This workflow reproduces all tables and figures related to GS-BART in the key sections of the paper, including 1) Section 5 (Simulation Studies and Real Data Analysis), and 2) Sections S4.2 of the Supplementary Material, which provide supplementary results to Sections 5, respectively.

Workflow

Location

The workflow is available:

☒ As part of the paper's supplementary material.
☐ In this Git repository:

☐ Other (please specify):

Format(s)

☐ Single master code file
☐ Wrapper (shell) script(s)
☒ Self-contained R Markdown file, Jupyter notebook, or other literate programming approach
☐ Text file (e.g., a readme-style file) that documents workflow
☐ Makefile
☐ Other (more detail in *Instructions* below)

Instructions

Instructions for use

All workflow information to reproduce key results of the paper is contained in `reproduce.Rmd`. The steps for running the workflow are:

Step 1: Install all required R packages in `dependencies.txt`.

Step 2 (Optional): Compile the source code for the GS-BART library by running the following command in your R Console:

```
install.packages('R packages/GSBart_0.0.1.tar.gz', type = "source", repo = NULL)
install.packages('R packages/G2SBart_0.0.1.tar.gz', type = "source", repo = NULL)
```

This step is optional and only required if you want to re-run GS-BART model fitting and prediction.

Step 3 (Optional): Modify the options in Line 19-25 of `reproduce.Rmd` accordingly. By default, key figures and tables are reproduced from pre-computed model outputs in `data/`. To re-generate model outputs, make the following change(s):

- To re-run GS-BART model in continuous simulation data, set `sim_continuous_fit = TRUE`.
- To re-run GS-BART model in count simulation data, set `sim_count_fit = TRUE`.
- To re-run GS-BART model in classification simulation data, set `sim_classification_fit = TRUE`.
- To re-run GS-BART model in continuous real data, set `real_continuous_fit = TRUE`.
- To re-run GS-BART model in count real data, set `real_count_fit = TRUE`.
- To re-run GS-BART model in classification real data, set `real_classification_fit = TRUE`.

Step 4: Run the workflow in `reproduce.Rmd` by compiling the R Markdown file. For example, this can be done by running the following command in your terminal:

```
Rscript -e "rmarkdown::render('reproduce.Rmd')"
```

Reproducible results can be found in the generated `reproduce.html` file.

Expected run-time

Approximate time needed to reproduce the analyses on a standard desktop machine:

☐ < 1 minute
☐ 1-10 minutes
☐ 10-60 minutes
☐ 1-8 hours
☒ > 8 hours
☐ Not feasible to run on a desktop machine, as described here:

Additional information (optional)

Notes (optional)