Testing GPBART

Mateus Maia 20250756

2022-11-16

Setting a example

This a vignette to explain how to run a simple example of the model, setting its own prior and its hyperparameters. To start we going to use the `friedman` example as the dataset to be used.

```
library(testgpbart)

# Setting simualation parameters
n <- 50
seed <- 42
sd <- 0.1

# Loading the data
fried_data <- sim_friedman(n = n,seed = seed,sd = sd)

# Setting a cross-validation object
cv_obj <- k_fold(data = fried_data,dependent_variable = "y",
                 k_partitions = 5,seed = seed)

# Selecting one fold
fold <- 1
x_train <- cv_obj[[fold]]$x_train
y_train <- cv_obj[[fold]]$y_train
x_test <- cv_obj[[fold]]$x_test
y_test <- cv_obj[[fold]]$y_test
```

Once specified all the data setting the next necessary specification is with respect the model settings. Here the most important thing that we want to evaluate is the prior for ϕ_{tp}. To specify that there are two main arguments in the function: `proposal_phi_` and `prior_phi_` that are going to specify the proposal and the prior for a MH sampling in that algorithm. For the `proposal_phi_` we need to enter a list with the `proposal_mode` and a possible `grid` to be used depending on the defined proposal. Examples for each case are explained below

- **Sliding-window proposal**: for this proposal the proposal for a new ϕ^*_{tp} came from a sample that follows

$$\phi^*_{tp} \sim U\left(3/4\phi_{tp}, 4/3\phi_{tp}\right)$$

  . Therefore there is no grid to be defined and the function argument should be given by

```
proposal_phi_ = list(proposal_mode = "sliding_window", grid = NULL)
```

- **Discrete grid**: for this proposal be given by a discrete uniform distribution with a range of ϕ^*_{tp} values. Therefore, is necessary to enter a vector with all values that ϕ_{tp} in that grid. The example below demonstrate one option of possible grid to be used

```
proposal_phi_ = list(proposal_mode = "discrete_grid",
                     grid = c(seq(0.1,20,length.out = 20),100))
```

Lastly, the default argument of the function takes uses

```
proposal_phi_ = list(proposal_mode = "discrete_grid",
                     grid = NULL)
```

where a default grid (showed below) is used as the standard one.

```
phi_proposal_ <- sample(c(0.1,seq(0,10,by=0.5)[-1],seq(10,20,by = 1),
                          25,30,50,75,100,125),
                        size = 1)
```

For the prior definition we have that it will gonna follow a mixture of gammas

$$\pi_1 G(\alpha = \alpha_1, \beta = \beta_1) + \pi_2 \left(\alpha = \alpha_1, \beta = \beta_1\right)$$

, where π_i corresponds to the probability of each gamma, and α_i and β_i are the shape and the rate parameters of each one respectively. Therefore we need to specify all parameters of it through the list

```
phi_prior_ <- list(type = "gamma_mixture",
                   prob_1 = 0.5,shape_1 = 1,rate_1 = 1,
                   prob_2 = 0.5,shape_2 = 1000,rate_2 = 10)
```

The list above would refer to a prior given by

$$0.5 \times G(\alpha = 1, \beta = 1) + 0.5 \times (\alpha = 1000, \beta = 10)$$

If null arguments are chosen for the `phi_prior_` the default prior is used follows

$$0.3 \times G(\alpha = 1.5, \beta = 0.2) + 0.7 \times G\left(\alpha = 10000, \beta = 100\right)$$

The code for it would be

```
proposal_phi_ = list(type = NULL,
                     prob_1 = NULL, shape_1 = NULL,rate_1 = NULL,
                     prob_2 = NULL,  shape_2 = NULL, rate_2 = NULL)
```

Running the model

Finally to run the model we would have:

```
gp_bart_mod <- gp_bart(x_train = x_train,
                       y_train = c(y_train),
                       x_test = x_test,
                       n_tree = 2,bart_boolean = TRUE,
                       gp_variables_ = colnames(x_train), # Selecting all var.
                       rotation_variables_ = colnames(x_train))
```

Real data benchmarking

To verify over other real data benchmarking use the other data from the package

```
# All other datasets
auckland
baltimore
boston
columbus
```

```
ny_PCTOWNHOME
ny_PROPCAS
sponge
swmud
petrel
precip
```

Evaluating the results

In order to check the model's performance we can calculate the RMSE and the CRPS, the example below show how to calculate for each one of them.

```
# Calculating the RMSE
rmse(obs = y_test,pred = colMeans(gp_bart_mod$y_test_hat_post))

# Calcusting the CRPS
crps(y = y_test,means = colMeans(gp_bart_mod$y_test_hat_post),
     sds = rep(mean(gp_bart_mod$tau_post[(gp_bart_mod$prior$n_burn+1):
                                           gp_bart_mod$prior$n_mcmc])^(-1/2),
               length(y_test)))$CRPS
```

Analyze other parameters can be also meaningful and relevant to get a proper diagnostic of the model. To get that we can do some trace plot of the model, or analyze the behavior of any length parameter of a tree.

```
# Doing a trace plot for \tau
plot(gp_bart_mod$tau_post, type = "l", ylab = "tau", xlab = "n_iter",
     main = "Posterior distributionf of tau")
```

To get the posterior distribution for ϕ_{tp} we gonna need to select which tree t and which p we are selecting.

```
# Defining parameters
tree <- 1
p <- 1
n_post <- length(gp_bart_mod$posterior$phi_post)

# Creating the vec.
phi_post <- numeric(n_post)
for(i in 1:n_post){
     phi_post[i] <- gp_bart_mod$posterior$phi_post[[i]][tree,p]
}

# Checking a traceplot
plot(phi_post, type = "l", ylab = "tau", xlab = "n_iter",
     main = paste0("Tree: ", tree, " - p: ", p))

# Checking the density
plot(density(phi_post),
     main = paste0("Tree: ", tree, " - p: ", p))
```