Using the `mybcart` package

Bruna Wundervald

March, 2019

This section will discuss the use of the BCART model implemented by hand in `R` (R Core Team (2018)), available at `https://github.com/brunaw/my_BART`.

The data was simulated using the model equation proposed in (Friedman (1991)). A dataset with 1000 rows was simulated, containing a response variable Y and its relationship to a matrix of predictors $\mathbf{X}$ as

$$y_i = 10sin(\pi x_{i1} x_{i2}) + 20(x_{i3} - 0.5)^2 + 10x_{i4} + 5x_{i5} + \epsilon_i,\ \epsilon_i \overset{iid}{\sim} N(0, \sigma^2), \tag{1}$$

where $\mathbf{X} \in [0, 1]$, meaning that the predictors were randomly drawn from a standard Uniform distribution. Figure 1 shows the relationship between each covariable and the response in this dataset. The nonlinearities and strenght of the interactcion between X_1 and X_2 are clear from the plot and the model equation, what ususally indicates that a tree based method would do better than a linear model.

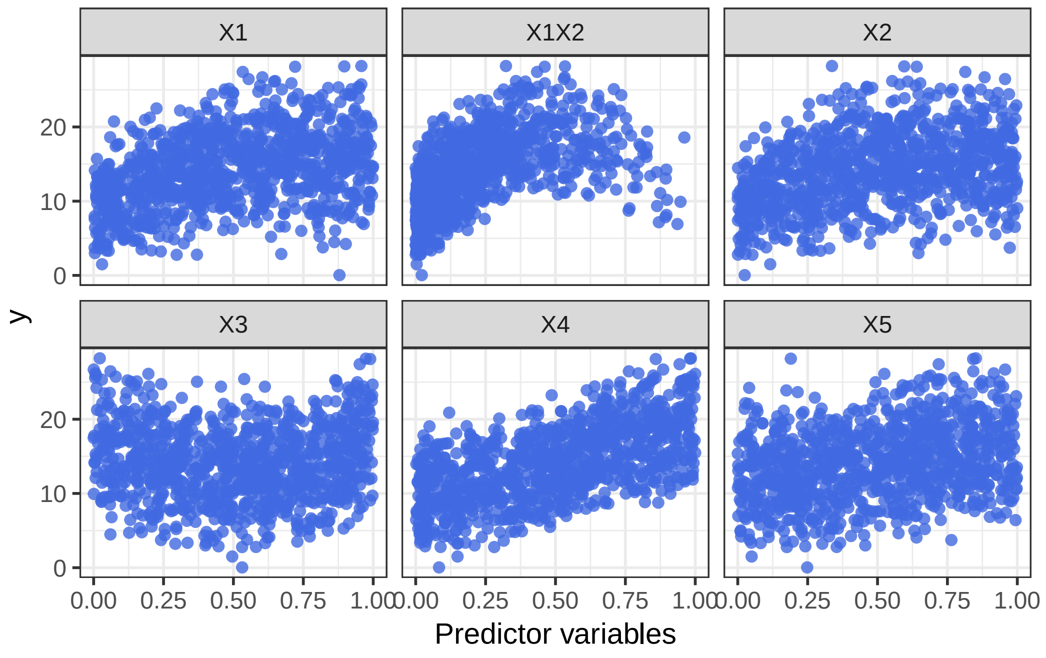

Figure 1: Relationship between the response variable and the predictors, for the data simulated with the Friedman method

For standard matters, the response variable is always scaled in our function, so the true $\sigma^2_y = 1$ and $\mu_y = 0$. The data was separated in train (app. 80% of the observations) and test (app. 20% of the observations). The train set was used to build the model, while the test was used to assess the quality of the prediction made by the final model. Table 1 shows the exact number of observation and proportion in each set.

We run our BCART code written by hand in R with 10000 iterations. The model considers the following prior for the means in each final node

$$\mu_\mu \sim \mathcal{N}(\mu_\mu, \sigma_\mu)$$

with $\mu_\mu = 0$ and $\sigma_\mu = \frac{y_{max}}{2}$, in a way the $\mu_\mu + 2\sigma_\mu = y_{max}$. As for the dispersion parameter of y, the prior is set as

$$\sigma_y^2 \sim InvGamma(\nu/2, \lambda\nu/2),$$

and the parameters λ and ν are found in order to give high probabilities of the BCART improving an ordinary least squares model (Kapelner and Bleich (2016)).

The chain for the posteriors values for σ_y^2 in our BCART model is presented in Figure 2. Apparently, the chain is converging well to a value close to 0.4. There are some small jumps, but nothing that really affected the convergence. The sum of the squared errors for the trees in each iteration are represented in Figures 3 and 4. The density has some small peaks in various points, which are better identified in the chain plot. The 'convergence' for the sum of squared error is not as fast as for σ_y^2, since the chain has more disturbance jumps, but it also eventually stabilizes in a value around 300.

Table 1: Number of observations and proportions in the training and test set

set	n	percent
test	188	18.8%
train	812	81.2%

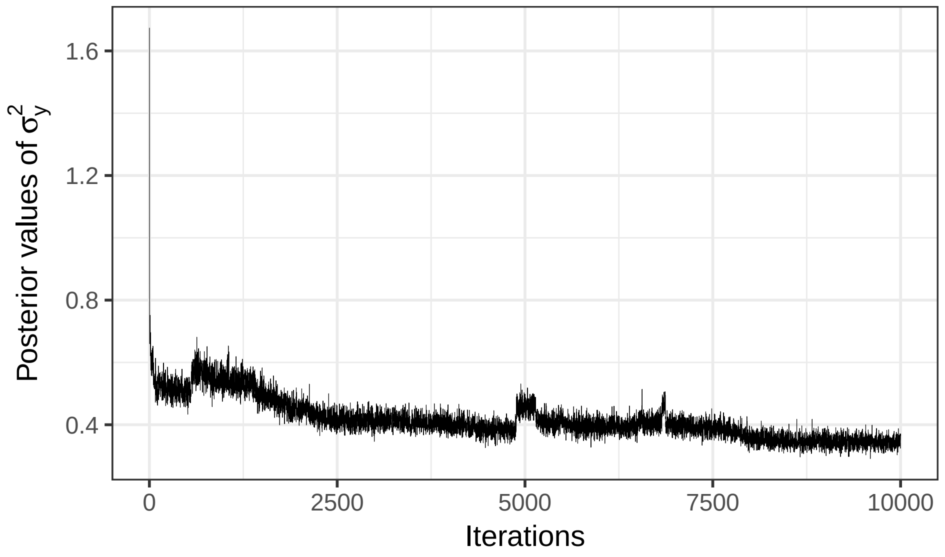

Figure 2: Chain of the posteriors values of σ_y^2 for a BCART model run with 10000 iterations

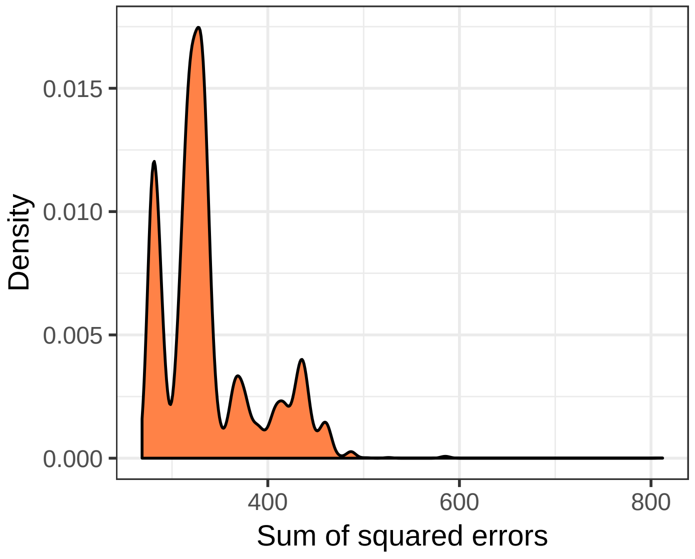

Figure 3: Density of sum of squared errors for all iterations of a BCART model run with 10000 iterations

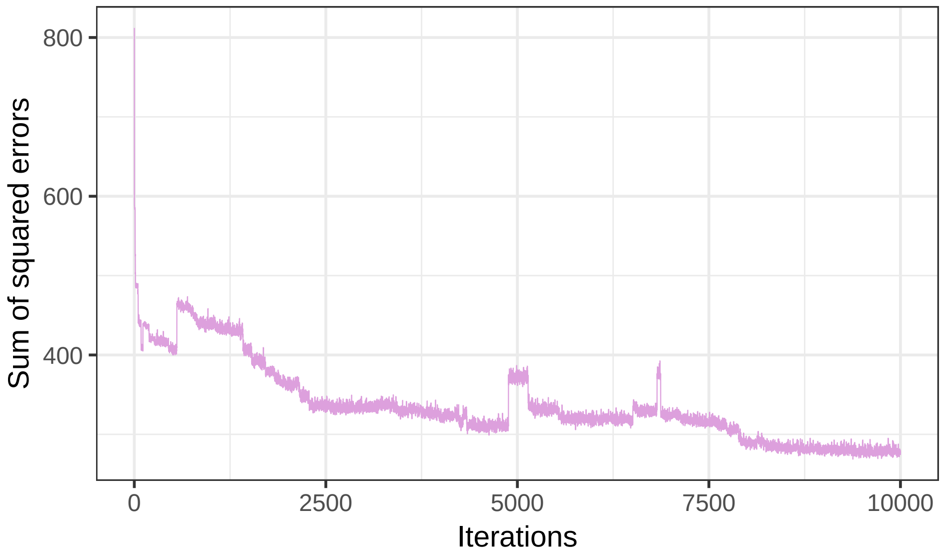

Figure 4: Chain of the sum of squared errors for a B-CART model run with 10000 iterations

In Table 2, we have the sum of squares of each node, the actual mean of y and the sampled values for μ using the test set. This table is shown for us to evaluate hot close the sampled vector μ is to the means of y, and what are the magnitudes of the SSR. The highest value for the SSR is 19, followed by 8.36, which are not so problematic, since those are the nodes with the most observations.

Comparing the sampled vector μ to the means in each node, we notice, in some cases, μ is close enough, as in node 2, 18 and 17, which have the smallest differences between the two values. On the other hand, we also have values for μ that are very dissimilar to the mean of y in the node, for which the worst case is node 22, followed by node 6. We conclude that, in general, our model is doing well for most of the nodes, as the SSR are mostly small and there are not many terminal nodes with a predicted μ highly different from the mean of y.

Table 2: Number of observation in each node, sum of squared errors, mean of reponse variable in each node, predictions for each node and difference between the predictions and the mean of y calculated in the test set for the the BCART model by hand.

Terminal node	**N**	**SSR of node**	**Mean of y**	**Prediction μ**	$\lvert\Delta\rvert$
1	20	8.3655759	1.0879026	1.1811396	0.0932370
2	5	0.3861218	1.7038866	1.6750519	0.0288346
3	3	2.2110082	1.6293809	2.0022985	0.3729176
4	4	1.4964365	1.1742133	1.0735486	0.1006647
5	8	7.7315778	0.8608978	1.7283068	0.8674090
6	3	3.3911446	0.1746179	1.1474496	0.9728316
7	5	0.5580049	1.0522706	0.8911906	0.1610800
8	3	0.4827284	0.7793404	0.6946262	0.0847142
9	15	4.3302991	0.2216585	0.4232476	0.2015891
10	18	7.3708203	-0.2887007	-0.3569353	0.0682347
11	4	1.1591034	0.3897473	0.1391794	0.2505678
12	43	19.0214892	-0.2300266	-0.0236719	0.2063547
13	20	6.7929217	-1.4195527	-1.3178136	0.1017391
14	9	3.7786860	0.0615870	0.5073813	0.4457943
15	7	1.5683653	-0.1825138	-0.0361611	0.1463527
16	6	3.2413661	-0.5780349	-0.6203806	0.0423457
17	4	0.7279391	-0.0128891	0.0528749	0.0399859
18	2	0.0035799	-1.3143844	-1.3356467	0.0212624
19	4	0.6612566	-0.9495645	-0.7398708	0.2096937
20	2	0.0212951	-0.3550087	-0.4580935	0.1030848
21	1	0.4669295	-1.1781138	-0.4947913	0.6833224
22	2	6.4033321	-1.8598555	-0.0890422	1.7708134

Comparing models

To finish the evaluation of the model, we compared its performance in the test set with the implementation of the BCART model made in the `bartMachine` package (Kapelner and Bleich (2016)) and a Random Forest model (Breiman (2001)). The `bartMachine` model was run with the `num_trees` set to 1, so it is a BCART model instead of BART, the `num_burn_in = 1000` as the number of burn-in iterations and the `num_iterations_after_burn_in = 9000`, representing the number os iterations after burn-in. This way, we have a comparable number to our model, since we used 10000 iterations. The Random Forest model was run with the implementation proposed in the `randomForest` package (Liaw and Wiener (2002)). The two additional models were also run with a scaled version of the response .

The results shown in Table 3 tells us that our model had a smaller mean squared error in the test set, when compared to the model produced by the `bartMachine` package. The model did not beat the Random Forest, but it also did not ended up too distant when considering the MSE. The average posterior for σ_y^2 was obtained for the two BCARTs, and our implementation had the largest one, even with the better results in the MSE.

Table 3: Mean squared error in the test set and average posterior σ_y^2 (for the Bayesian models only) for the BCART model by hand, the BCART usign the ‘bartMachine‘ package and a Random Forest.

	BCART by hand	bartMachine	randomForest
MSE	0.4264361	0.5751628	0.3234014
Average posterior variance	0.4021115	0.2621000	NA

Bibliography

Breiman, Leo. 2001. "Random Forests." *Machine Learning*. https://doi.org/10.1017/CBO9781107415324.004.

Friedman, Jerome H. 1991. "Rejoinder: Multivariate Adaptive Regression Splines." *The Annals of Statistics*. https://doi.org/10.1214/aos/1176347973.

Kapelner, Adam, and Justin Bleich. 2016. "bartMachine: Machine Learning with Bayesian Additive Regression Trees." *Journal of Statistical Software* 70 (4): 1–40. https://doi.org/10.18637/jss.v070.i04.

Liaw, Andy, and Matthew Wiener. 2002. "Classification and Regression by randomForest." *R News* 2 (3): 18–22. https://CRAN.R-project.org/doc/Rnews/.

R Core Team. 2018. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.