CREATING BAYESIAN ADDITIVE REGRESSION TREE MODELS AND COMPARING ERROR RATES OF MODELS ON DIAGNOSING HEART DISEASE

Executive Summary

In this paper we explore Bayesian Additive Regression Trees (BART), which produce a dynamic model used for classification and regression. BART is a flexible model because it can account for interactions between variables easily and be implemented with missing data values. We explore the theory behind BART, and evaluate it as a classification model. The evaluation consisted of determining the error rate, or misclassification rate, of detecting heart disease in patients from the heart disease dataset provided by Kaggle. The dataset consists of 13 variables and 1 target variable which is the presence of heart disease. The analyses conducted in this work is an extension to the 2016 ACIC competition in which BART regression models were compared to other regression models. BART performed better than most models in the competition in terms of bias and root mean squared error. In this paper we compare BART as a classification model against a probit model, quadratic discriminant analysis model (QDA), and support vector machine model (SVM). The error rates of the 4 models were calculated with all of the 13 covariates, followed by another comparison using 10 of the covariates which were selected based on their significant p-values from the probit model. A third comparison was made with the 13 covariates, but with randomly placed NA values. The purpose of this was to compare the models' error rate in an application setting where we might have missing data. In each analysis, BART performed best by having the smallest error rate and a significantly lower rate than the probit model when the full heart disease dataset was used.

At the 2016 Atlantic Causal Inference Conference (ACIC), a competition was held to evaluate the performance of different methods for estimating causal effects from observed data. The competition focused on estimating the causal effect of a treatment on an outcome, given a set of covariates. Several methods were submitted in the competition and evaluated on their ability to estimate the true causal effect and their computational efficiency. Douglas Galagate of the University of Maryland and Nicole Bohme Carnegie of the University of Wisconsin-Milwaukee submitted the Bayesian Additive Regression Trees model (BART) which performed very well. Compared to other models in one part of the competition, BART had the least bias, least root mean squared error (RMSE), the shortest confidence intervals, and was computationally efficient. The results of the competition showed that Bayesian Additive Regression Trees (BART) performed the best overall. BART was able to estimate the causal effect of the treatment on the outcome with high accuracy and low bias. The method was also computationally efficient, making it suitable for large datasets. (Hill, 2011) With this model performing so well against others it is necessary to understand it and know how to implement our own BART model. The competition described above was based around regression models so we are going to explore BART as a classification model and determine how it measures up against other models in terms of their error rate, which is the rate of misclassification.

Before we dive deeper into BART it is worth noting some of its features that helped it standout from other methods. BART can handle missing data well and can capture interactions between variables. BART is flexible because it combines Bayesian inference and regression trees, allowing it to handle interactions between variables and capture nonlinearity between variables. (Hill, 2011) These features make BART well suited for applications across industries and useful for researchers who may not be the most proficient at modeling relationships between a response variable and covariates.

Growing and Pruning Bayesian Additive Regression Trees

Unlike traditional regression trees, which give one fit in which the data is split into binary partitions, BART fits multiple regression trees to the data and combines them to make the prediction model. BART models are built by combining a large number of these trees, and the final prediction is obtained by taking the average of all the predictions from the individual trees. BART trees are flexible in that they fit the data from continuous, categorical, and mixed data types. (Hill, 2011) As we will see shortly BART is based on breaking down the problem into smaller parts and looking at each part separately. It uses regression trees to do this. Imagine a tree with branches - each branch represents a different possibility for a feature or characteristic you're looking at. The tree splits into smaller branches until each branch only represents a small group of things that are similar in some way. (Tan & Roy, 2019)

BART models are so powerful due to its dynamic construction. Before we formally define BART let's explore how the model is created. We first determine how many trees we want to initialize and how many iterations of growing and pruning these trees we want. In our example let's say we have three covariates $X = (X_1, X_2, X_3)$ and our response variable Y is continuous. In Figure 1a we see the initial 4 trees as nodes, each of which have a value of $\frac{\bar{Y}}{m}$

where *m* = 4 referring to the 4 trees. Within each tree we see each $\widehat{\mu}^{B}_{jk}$ which is the mean parameter of kth node for the jth regression tree at iteration B. BART uses Markov Chain Monte Carlo (MCMC) to randomly draw the tree structure for each tree. We begin calculating the residual value, that is, the difference in the actual value less the sum of the $\widehat{\mu}^{B}_{jk}$ values for all but Tree 1. (Tan & Roy, 2019)

The next step is where the "Bayesian" comes in. We will refer to this later as the *Bayes step*. After r_1 is calculated, a new tree structure for T_1 will be suggested, let's call it T_1^*, using a Markov Chain Monte Carlo (MCMC) procedure. The structure will be accepted based on the likelihood of the observed residual value given the new structure ($r_1|T_1^*$), the likelihood of the residual given the previous structure ($r_1|T_1$), the probability of observing T_1^*, the probability of observing T_1, and the probability of moving from T_1^* to T_1 and the probability of moving from T_1 to T_1^*. This determines (T_1, M_1) and the process continues through each tree within the iteration. (Tan & Roy, 2019) We can see in Figure 1a below that as the process continues in Iteration 1, only Tree 3 gained a split with two branches adding a node.

Figure 1b gives the result after 5 iterations where the trees only have a maximum depth of 3. This is due to trees being penalized for branching very deep. This is explained later in the paper. We can think about this as the probability of any particular tree growing very deep decreasing.

More formally, a BART model is specified as

$$f(X) = \sum_{j=1}^{m} g(X, T_j, M_j).$$

In the above equation, *X* is our set of covariates, T_j is the jth tree, and $M_j = \{\mu_{1j}, \ldots, \mu_{bj}\}$ is the vector of terminal node parameters associated with T_j. What we are summing is actually many of residuals calculated from actual values and the predicted values being created at each tree with each iteration. Once we have the initialization in place, BART creates the tree structures. It does so with a "leave-one out" principle by calculating

$$r_1 = Y - \sum_{j\neq 1} g(X, T_j, M_j) = Y - [g(X, T_2, M_2) + g(X, T_3, M_3) + g(X, T_4, M_4)] = Y - 3\frac{\bar{Y}}{m}$$

Notice there was no g function for $j = 1$. This is where an algorithm is applied to determine the tree structure based on the likelihood described above. After doing so the the parameter values M_1 are updated based on the new tree structure that is accepted, T_1 or T^*_1, and then we move on to T_2 and M_2. To determine these values BART does the following

$$r_2 = Y - \sum_{j\neq 2} g(X, T_j, M_j) = Y - [g(X, T_1, M_1) + g(X, T_3, M_3) + g(X, T_4, M_4)] = Y - [\hat{\mu}^{(1)}_{11} + 2\frac{\bar{Y}}{m}]$$

Note that $\hat{\mu}^{(1)}_{11}$ is the updated parameter for Tree 1. The likelihood of the new struction T^*_2 is assessed and then is accepted or rejected. This process continues through the iteration steps until we reach our predetermined number of iterations on our predetermined number of trees. (Tan & Roy, 2019)

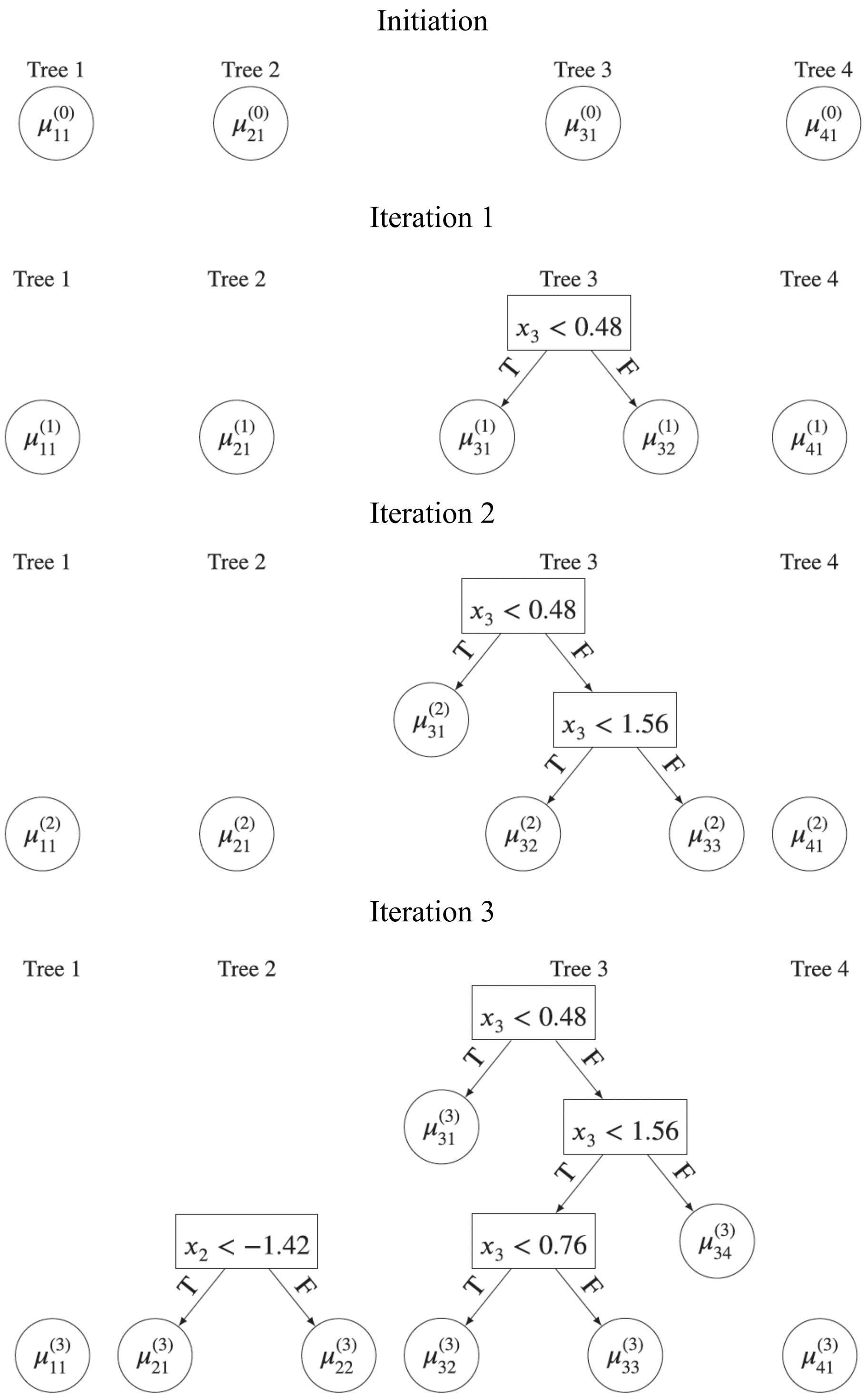
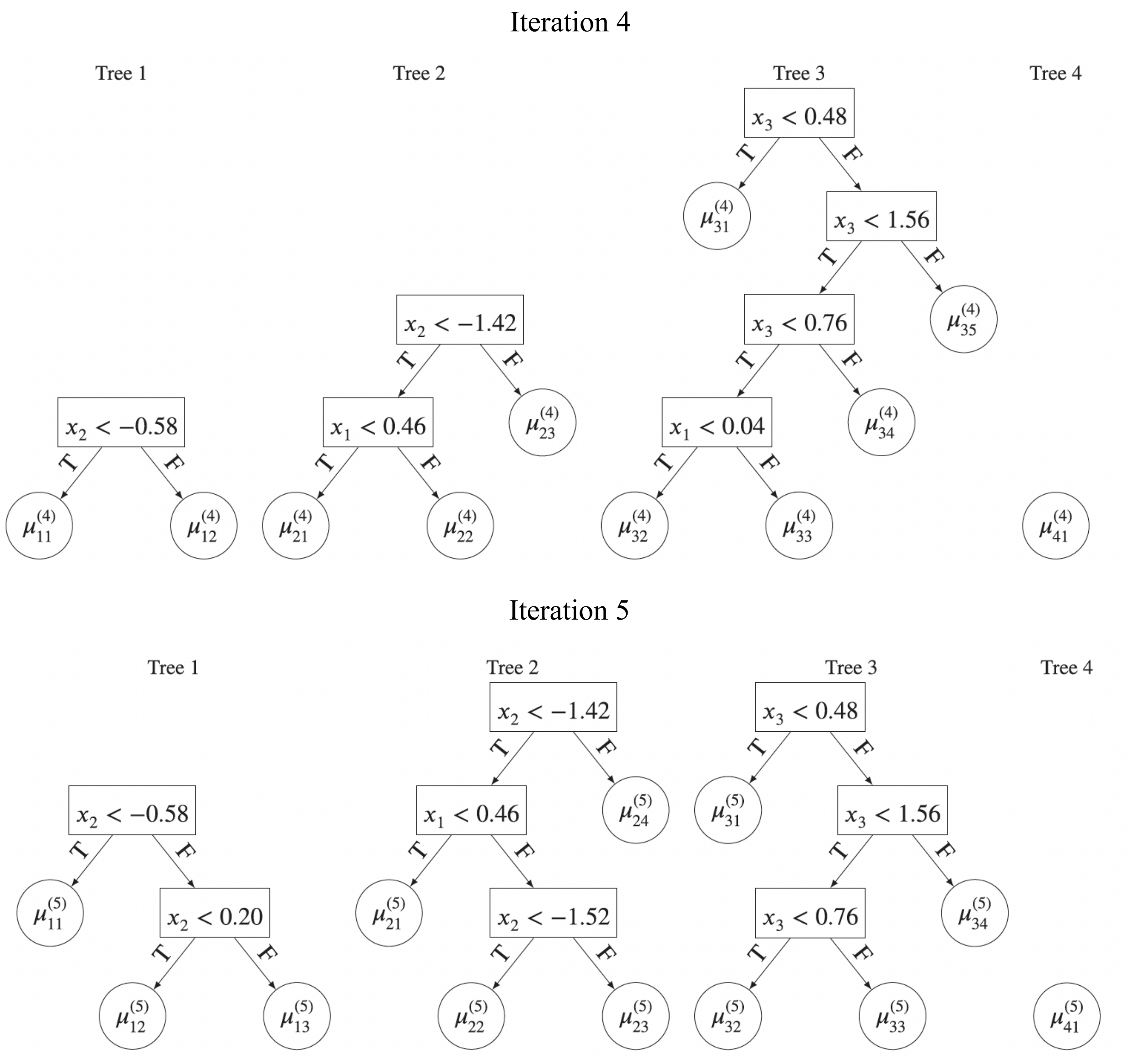


Figure 1a
The first 3 iterations of a BART model with 4 trees (Tan & Roy, 2019)


Iteration 4

Iteration 5

Figure 1b
The 4th and 5th iterations of a BART model with 4 trees (Tan & Roy, 2019)

A concern of any model is overfitting. Overfitting occurs when a machine learning model becomes too complex and memorizes the training data rather than learning the underlying patterns that generalize well to new, unseen data. This leads to a model that performs well on the training data but poorly when new data is fed into it. BART avoids overfitting in the Bayes step which was mentioned above as the penalizing feature of BART growing trees too deep which would cause overfitting. Each node at depth *d* has a probability of splitting given by $\frac{\alpha}{(1+d)^{\beta}}$. Here $\alpha \in \{0, 1\}$ where values closer to 1 increase the likelihood of a split. The parameter $\beta > 0$

reduces the number of terminal nodes as β increases. This creates weaker trees which are then summed together in our final model.

Above, we described the BART model used in regression and we also want to describe how BART is used as a classifier. The process is similar to the one above however it is transformed into a probit model as

$$P[Y = 1|x] = \Phi[\sum_{j=1}^{m} g(X, T_j, M_j)]$$

where Φ[.] is the standard normal cdf. We define variables Z = {Z_1, ... , Z_n} for our cdf as

$$Z_i \sim N\ [\sum_{j=1}^{m} g(X, T_j, M_j), 1] \quad \text{if } Y_i = 0$$

$$Z_i \sim N\ [\sum_{j=1}^{m} g(X, T_j, M_j), 1] \quad \text{if } Y_i = 1$$

We have the default interval (-3, 3) for Z which helps keep our probabilities in a range that prevents the trees from growing too deep. These values can be changed as desired, but they were left alone for this work. With Z as a continuous variable, our BART model will compute

$$Z = \sum_{j=1}^{m} g(X, T_j, M_j) + \varepsilon$$

where ε ~ N(0,1). The algorithm to compute Z is the same as above. Now that we understand how a BART model is created, we will now create a model based on the heart disease from Kaggle and compare the model to other types of models.

Our Data

The data used in this project comes from individuals from Cleveland, Hungary, Switzerland, and the VA Long Beach healthcare system. The original dataset contains 76 attributes, but the dataset as provided on Kaggle contains a subset of 14 covariates. The target variable takes a value of 0 for no disease and a value of 1 for the presence of heart disease. Below is a table of the possible covariates and whether they are discrete or continuous.

1. sex (factor)
2. chest pain type (factor)
3. Age (numeric)
4. resting blood pressure (numeric)
5. serum cholesterol in mg/dl (numeric)
6. fasting blood sugar > 120 mg/dl (factor)
7. resting electrocardiographic results (factor)
8. maximum heart rate achieved (numeric)
9. exercise induced angina (factor)
10. oldpeak = ST depression induced by exercise relative to rest (numeric)
11. the slope of the peak exercise ST segment (factor)
12. number of major vessels colored by fluoroscopy (factor)

13. thalassemia (factor)

Creating Our BART Model

To create our BART model we will be using the bartMachine function from the *bartMachine* R package. The code to generate our model is provided below. We split our heart disease data into training and testing datasets and use the former to create our model. The first two inputs are the covariates dataframe and the target vector, in that order. The next three parameters define how we want BART to run. The parameter `num_trees` defines how many trees we want initialized, `num_burn_in` defines how many of the initial trees we want to throw away, and `num_iterations_after_burn_in` defines how many of our iteration steps we want. Referring to our example above where we have 4 initial trees that start as nodes, our model will have 150. Where we have 5 total iterations above, here we will have 1000 after we throw away the first 100. The parameter `use_missing_data` tells the machine to continue running through the algorithm above even if there are missing values within the covariate data table. This highlights one of the features of BART: the bartMachine can create a model with missing data while other models would require removing rows with NA values. Depending upon the application, it still might be better to clean up the data set, but we could make the model regardless.

```
bart_machine = bartMachine(within(train, rm(target)), train$target,
                      num_trees=150, num_burn_in = 100,
                      num_iterations_after_burn_in = 1000,
                      use_missing_data = TRUE,
                      seed = 42)
```

Figure 2 contains some interesting plots that show the BART creation process. At left we see the percent of trees that were accepted to create new branches. The olive green circles are the % MCMC steps accepted for each of the trees in the iteration and the gray circles are those accepted in the 100 initial trees that were thrown out. To generate these plots we use the `plot_convergence_diagnostics(bart_machine)` function with the `bart_machine` object as the input.

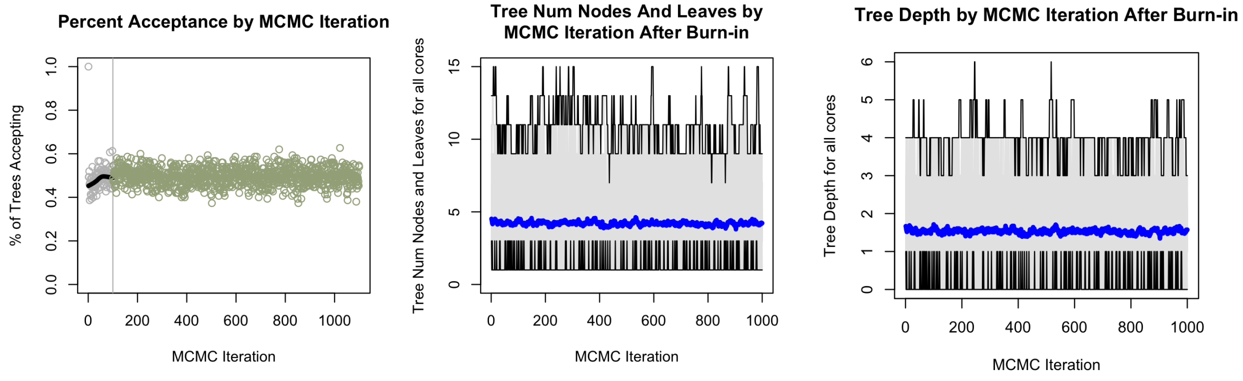


Figure 2
BART Structure over the MCMC Iterations

The middle plot displays the average number of nodes across each tree in the model at each of the 1000 MCMC iterations. The blue line just under 5 is the average number of nodes over all trees. The right graph shows the average tree depth across each tree in the model. The blue line is the average number of nodes over all trees. These plots demonstrate how BART favors many small trees, which on their own would be dubbed "weak learners", but when we take them together we get an effective and accurate predictor.

Analyzing and Comparing our BART Model

We will now analyze our BART model in terms of its error rate at incorrectly predicting heart disease and compare it to that of classifier models created using a logistic model, quadratic discriminant analysis model (QDA), and a support vector machine model (SVM). In our course this term we created all three of these models and compared error rates between models. The error rates are calculated as taking the sum of the false positives and false negatives divided by the total tested. We will follow a similar procedure here. We began by creating the 4 models using all 13 covariates with the same training set, creating the confusion matrices and then determining their error rates with 95% confidence intervals. Table 1 contains error rate with 95% confidence intervals and the interval coverage. Figure 3 contains the confusion matrices for each model accompanied by the error rates with 95% confidence intervals.

Model (Full dataset)	**Rate (95% CI)**	**CI Coverage**
BART	0.075 (0.045, 0.104)	0.059
Probit	0.156 (0.115, 0.196)	0.081
QDA	0.123 (0.087, 0.160)	0.073
SVM	0.127 (0.089, 0.164)	0.075

Table 1

Classification model using the full dataset, 95% confidence interval, and interval coverage

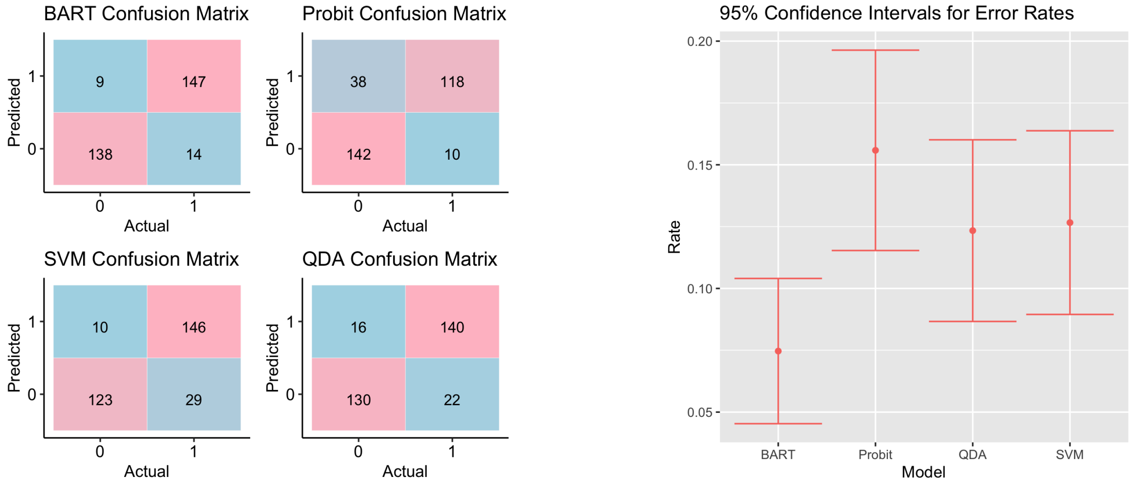


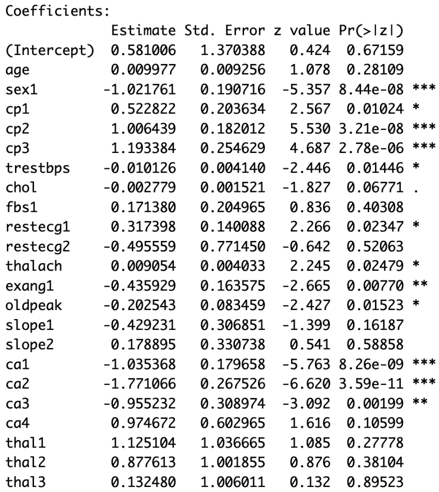
Figure 3

Confusion Matrices and 95% Confidence Intervals for Full Dataset

From this analysis we see that BART performed better than the other 3 models, and significantly better than the probit model. We also have narrower CI length in the BART model meaning that our confidence in capturing the true error is contained in a smaller range than the other variables.

Considering that too many variables can cause model complexity, let's explore a way to simplify our model. One method we discussed in our class to select which variables to remove was to remove those with the highest p-values. Since BART is a nonparametric model, we need to look elsewhere. Since the probit model is parameterized we will use this model. Figure 4a is an R print out of the coefficients, standard errors, Z values, and p-values. The `slope` and `thal` (thalassemia) factors had high p-values as did `fbs` (fasting blood sugar). Therefore, we eliminated these three variables and made new training and testing datasets to run the same analysis. Applying a similar approach to BART, we can look at the variable inclusion proportions which tell us about the relative influence of the different variables. We used the function `investigate_var_importance(bart_machine, num_replicates_for_avg = 20)` where we have BART model input and the number of replicates of BART used to generate variable inclusion proportions (VIP). This refers to the number of times the BART algorithm runs on the same data set to produce an estimate of variable inclusion proportions. The VIP measures the importance of each predictor variable in a BART model by computing the proportion of times the variable is selected for splitting in the trees across all the BART replicates. The higher the VIP value, the more important the variable is in predicting the response variable. Figure 4b is the R print out of the VIP which gives the variables on the horizontal axis and the inclusion proportion on the vertical axis. From this we see that `slope`, `thal`, and `fbs` were not the least included variables, but they were on the far right portion of the tail which highlight variables contained in less than 3% of the trees.

Although the third factor of the `thal` variable is on the left of the plot where it was contained in 4% of the trees where it is ranked 7 of 30 in terms of the most included variables in the trees. We will discuss this `thal_3` below, but in the meantime we will remove `slope`, `thal`, and `fbs`.



```
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.581006   1.370388   0.424  0.67159
age          0.009977   0.009256   1.078  0.28109
sex1        -1.021761   0.190716  -5.357 8.44e-08 ***
cp1          0.522822   0.203634   2.567  0.01024 *
cp2          1.006439   0.182012   5.530 3.21e-08 ***
cp3          1.193384   0.254629   4.687 2.78e-06 ***
trestbps    -0.010126   0.004140  -2.446  0.01446 *
chol        -0.002779   0.001521  -1.827  0.06771 .
fbs1         0.171380   0.204965   0.836  0.40308
restecg1     0.317398   0.140088   2.266  0.02347 *
restecg2    -0.495559   0.771450  -0.642  0.52063
thalach      0.009054   0.004033   2.245  0.02479 *
exang1      -0.435929   0.163575  -2.665  0.00770 **
oldpeak     -0.202543   0.083459  -2.427  0.01523 *
slope1      -0.429231   0.306851  -1.399  0.16187
slope2       0.178895   0.330738   0.541  0.58858
ca1         -1.035368   0.179658  -5.763 8.26e-09 ***
ca2         -1.771066   0.267526  -6.620 3.59e-11 ***
ca3         -0.955232   0.308974  -3.092  0.00199 **
ca4          0.974672   0.602965   1.616  0.10599
thal1        1.125104   1.036665   1.085  0.27778
thal2        0.877613   1.001855   0.876  0.38104
thal3        0.132480   1.006011   0.132  0.89523
```

Figure 4a
Probit Coefficients

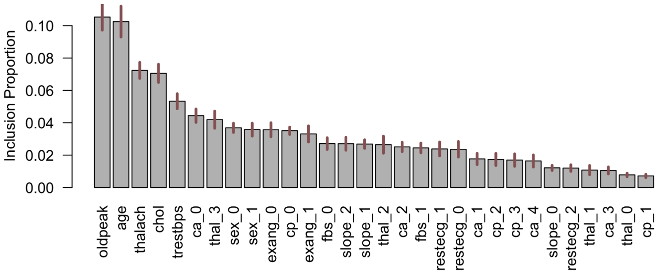


Figure 4b
BART Variable Importance

After creating new models with these variables removed from the training and testing sets we find the error rates all increased meaning each model incorrectly classified individuals more frequently. (Table 2 and Figure 5) We also see from this modified analysis that BART still performed best out of the 3 models in terms of error rate, the QDA model had the narrowest CI coverage.

Model (10 variables)	**Rate (95% CI)**	**CI Coverage**
BART	0.114 (0.078, 0.149)	0.071
Probit	0.159 (0.118, 0.200)	0.082
QDA	0.143 (0.104, 0.182)	0.039
SVM	0.133 (0.095, 0.171)	0.076

Table 2
Classification model using the 10 covariates, 95% confidence interval, and interval coverage

BART coming out ahead of the other models is a reasonable observation for a few reasons. First, humans are complex beings and it is difficult to isolate causal variables when there are so many at play in any being. To remove 3 variables when making a classification model to predict heart disease even when they are seemingly insignificant could remove interacting variables that are conditioned on other variables that on their own might be insignificant. In this experiment we removed `thal` by merely looking at p-values. But our variable inclusion report tells us that on its factors, `thal_3`, plays a more significant role compared to other variables in our BART model. Second, BART is a nonparametric model and can handle variable interactions easily due to its tree growing algorithm. (Hill, 2011) BART can piece apart the complexity of variable interactions that other models might miss.

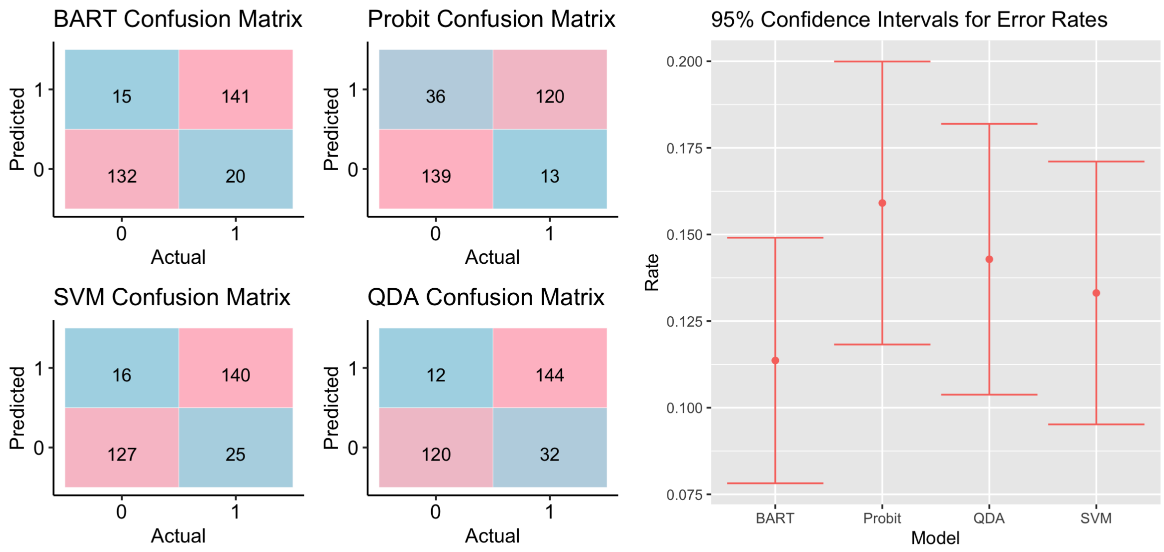


Figure 5
Confusion Matrices and 95% Confidence Intervals with 3 Variables Removed

Another feature of BART that makes it so easy to use is that it can handle missing data values in the dataset while still producing a decent model. To test this feature we randomly assigned NA values to our covariate dataset while retaining each value in `target`. For this test we randomly replaced just under 1% of the covariate values and only looked at the error rates. This test was conducted to simulate an application setting where data was either not collected about a patient or is just not possible to collect. The error rates are as follows: BART - 0.084, Probit - 0.156, QDA - 0.143, SVM - 0.127. Even with missing data BART outperforms the other models in terms of error rate.

Summary

Whether regression or classification purposes, there are many types of models to choose from when wanting to create a model. And when making the decision there are various measures to consider when making the selection. Hill (2019) compared regression models based on their bias and RMSE. Carnegie (2019) compared BART regression models with and without propensity scores. While these papers highlighted the use of BART for regression, this paper explored BART and other common models for classification purposes. From this analysis we support Hill's findings for the superiority of BART, but as a classification model. It is important to note that Hill explored BART in terms of causal inference using a regression model, so an extension of this project for further research would be to do the same for a classification model. Other facets of BART research to explore is its use with incomplete datasets. We used a dataset with randomized missing values and BART performed best among the models. It would be useful to see how BART performs with more data removed when compared to other models.

Project Bibliography

Carnegie, Nicole Bohme. "Comment: Contributions of Model Features to BART Causal Inference Performance Using ACIC 2016 Competition Data." *Statistical Science* 34, no. 1 (February 2019): 90–93. https://doi.org/10.1214/18-STS682.

Chipman, Hugh A., Edward I. George, and Robert E. McCulloch. "BART: Bayesian Additive Regression Trees." *The Annals of Applied Statistics* 4, no. 1 (March 2010): 266–98. https://doi.org/10.1214/09-AOAS285.

Hill, Jennifer L. "Bayesian Nonparametric Modeling for Causal Inference." *Journal of Computational and Graphical Statistics* 20, no. 1 (January 1, 2011): 217–40. https://doi.org/10.1198/jcgs.2010.08162.

Tan, Yaoyuan Vincent, and Jason Roy. "Bayesian Additive Regression Trees and the General BART Model." arXiv, January 22, 2019. https://doi.org/10.48550/arXiv.1901.07504.

You can find the code used in this analysis on my github.
github.com/mathneighborhood/BART_classification_comparison

Carnegie, Nicole Bohme. “Comment: Contributions of Model Features to BART Causal Inference Performance Using ACIC 2016 Competition Data.” *Statistical Science* 34, no. 1 (February 2019): 90–93. https://doi.org/10.1214/18-STS682.

Chipman, Hugh A., Edward I. George, and Robert E. McCulloch. “BART: Bayesian Additive Regression Trees.” *The Annals of Applied Statistics* 4, no. 1 (March 2010): 266–98. https://doi.org/10.1214/09-AOAS285.

Dorie, V. (2017). Aciccomp2016: Atlantic Causal Inference Conference Competition 2016 Simulations. R Package Version 0.1-0. https://github.com/vdorie/aciccomp

Dorie, Vincent, Jennifer Hill, Uri Shalit, Marc Scott, and Dan Cervone. “Automated versus Do-It-Yourself Methods for Causal Inference: Lessons Learned from a Data Analysis Competition.” *Statistical Science* 34 (July 9, 2017). https://doi.org/10.1214/18-STS667.

Hill, Jennifer L. “Bayesian Nonparametric Modeling for Causal Inference.” *Journal of Computational and Graphical Statistics* 20, no. 1 (January 1, 2011): 217–40. https://doi.org/10.1198/jcgs.2010.08162.

Rosenbaum, Paul R., and Donald B. Rubin. “The Central Role of the Propensity Score in Observational Studies for Causal Effects.” *Biometrika* 70, no. 1 (1983): 41–55. https://doi.org/10.2307/2335942.

Tan, Yaoyuan Vincent, and Jason Roy. “Bayesian Additive Regression Trees and the General BART Model.” arXiv, January 22, 2019. https://doi.org/10.48550/arXiv.1901.07504.

Jon Southam
726 Project Full Methods

Below is a slimmed down bibliography from the bibliography I originally submitted, and a brief description of how I intend to use them.

Carnegie, Nicole Bohme. "Comment: Contributions of Model Features to BART Causal Inference Performance Using ACIC 2016 Competition Data." *Statistical Science* 34, no. 1 (February 2019): 90–93. https://doi.org/10.1214/18-STS682.

Hill, Jennifer L. "Bayesian Nonparametric Modeling for Causal Inference." *Journal of Computational and Graphical Statistics* 20, no. 1 (January 1, 2011): 217–40. https://doi.org/10.1198/jcgs.2010.08162.

Dorie, V. (2017). Aciccomp2016: Atlantic Causal Inference Conference Competition 2016 Simulations. R Package Version 0.1-0. https://github.com/vdorie/aciccomp

Jennifer Hill's paper was the jumping off point for my work. She explains what BARTs are and some of the theory behind them. Carnegie outlines a bit more on the Atlantic Causal Inference Conference competition from 2016 where researchers submitted different types of models and their models' ability to accurately detecting causes. This was assesses by looking at the model's bias and root mean squared error. Dorie's GitHub page contains quite a bit of code that is difficult to decipher as it includes the code for other methods from the ACIC competition.

While we have not discussed Regression Trees yet in this course, these are going to be discuss in the upcoming module on classification.

Tan, Yaoyuan Vincent, and Jason Roy. "Bayesian Additive Regression Trees and the General BART Model." arXiv, January 22, 2019. https://doi.org/10.48550/arXiv.1901.07504.

Tan and Roy's work explains the BART process a bit more to make the theory explained by Hill more concrete.

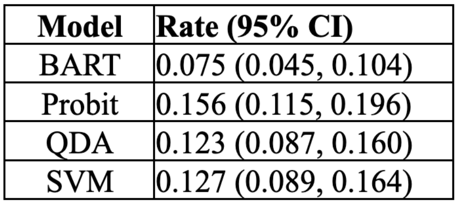
Yoshida, Kazuki. "Bayesian Additive Regression Trees." October, 16, 2018. https://rpubs.com/kaz_yos/bart1

Yoshida gives some sample code on creating a BART which is a bit more accessible for an introduction to BARTs. Much of my time the past few weeks has been digging through Dorie's code to no avail. This work from Yoshida should help alleviate the struggles navigating with Dorie's GitHub.

Model (missing data)	**Rate**
BART	0.084
Probit	0.156
QDA	0.143
SVM	0.127

BART -, Probit -, QDA -, SVM -.

Model (10 variables)	**Rate (95% CI)**	**CI Coverage**
BART	0.114 (0.078, 0.149)	0.071
Probit	0.159 (0.118, 0.200)	0.082
QDA	0.143 (0.104, 0.182)	0.039
SVM	0.133 (0.095, 0.171)	0.076



Model	**Rate (95% CI)**
BART	0.075 (0.045, 0.104)
Probit	0.156 (0.115, 0.196)
QDA	0.123 (0.087, 0.160)
SVM	0.127 (0.089, 0.164)

Model (Full dataset)	**Rate (95% CI)**	**CI Coverage**
BART	0.075 (0.045, 0.104)	0.059
Probit	0.156 (0.115, 0.196)	0.081
QDA	0.123 (0.087, 0.160)	0.073
SVM	0.127 (0.089, 0.164)	0.075