BEST : A decision tree algorithm that handles missing values

Cédric Beaulac [1] Jeffrey S. Rosenthal [2]

November 28, 2018

Abstract

In this article we propose a new decision tree algorithm. The proposed approach allows the algorithm to interact with some predictors that are only defined in subspaces of the predictor space. One way to utilize this new algorithm is to create or use a predictor to keep track of missing values. This predictor can later be used to define the subspace where predictors with missing values are available for the data partitioning process. By doing so, this new classification tree can handle missing values for both modelling and prediction. The algorithm is tested against simulated and real data. The result is a classification procedure that efficiently handles missing values and produces results that are more accurate and more interpretable than most common procedures.

Keywords : Classification and Regression Tree, Missing Data, Applied Machine Learning, Data analysis, Interpretable Models, Variable Importance Analysis

[1]University of Toronto
Department of Statistical Sciences
Sidney Smith Hall, 100 St George St, Toronto, ON M5S 3G3
E-mail : beaulac.cedric@gmail.com
ORCID : 0000-0002-6050-5313
[2]University of Toronto
Department of Statistical Sciences
Sidney Smith Hall, 100 St George St, Toronto, ON M5S 3G3
E-mail : jeff@math.toronto.edu

1 Introduction

Machine learning algorithms are used in many exciting real data applications, but may have problems handling predictors with missing values. Many solutions have been proposed to deal with observations that are missing completely at random (MCAR). Since this is a restrictive assumption we propose a solution to missing values that uses the tree structure of Classification and Regression Trees (CART) to deal in an intuitive manner with observations that are missing in patterns which are not completely at random.

Our proposed new tree construction procedure was inspired by a data set where the missing pattern of one subset of predictors could be perfectly explained by another subset (see Section 4.1). A typical decision tree is an algorithm that partitions the predictor space based upon a predictor value, splitting it into two subspaces and repeats this process recursively. Our proposed algorithm is different as it allows the researcher to impose a structure on the variables available for the partitioning process. By doing so, we construct Branch-Exclusive Splits Trees (BEST). When a predictor X_j contains missing values, we can use other predictors to identify the region where the predictor X_j contains no missing value. Therefore we can use the proposed algorithm to consider splitting on a predictor only when it contains no missing value based on previous partitioning. BEST can be easily adapted to any splitting rule and any forest forming procedure [4, 5, 10]. BEST also has other applications; it can be used by researchers that would like to utilize some knowledge they have on the data generating distribution in order to guide the algorithm in selecting a more accurate and more interpretable classifier.

In this article we will briefly discuss the classification problem and its notation and we will explain how classification trees solve that problem. We will then do a quick review of the missing values treatments that are currently being used. Afterwards, we will introduce the proposed algorithm and some motivating examples before explaining in detail how the algorithm functions. Finally, some tests will be performed on simulated data sets and on the real data that inspired this new algorithm.

2 The classification problem

In a typical supervised statistical learning problem we are interested in understanding the relationship between a response variable $\mathbf{Y}$ and an associated m-dimensional predictor vector

$\mathbf{X} = (X_1, ..., X_m)$. When the response variable is categorical and takes k different possible values, this problem is defined as a k-class classification problem. In that set up, an interesting challenge is to use a data set $S = \{(y_i, x_{i,1}, ..., x_{i,m}); i = 1, ..., n\}$ in order to construct a classifier h. Most of the time, it is assumed that the observations within our data set were drawn independently from the same unknown and true distribution $\mathcal{D}$, i.e. $\mathbf{X} \times \mathbf{Y} \sim \mathcal{D}$. A classifier is built to emit a class prediction for any new data point X that belongs in the predictor space $\mathcal{X} = \mathcal{X}_1 \times ... \times \mathcal{X}_m$. Therefore a classifier divides the predictor space $\mathcal{X}$ into k disjoint regions $R_1, ...R_k$, one per class, such that $\cup_{q=1}^{k} R_1 = \mathcal{X}$, i.e. $h(x) = \sum_{q=1}^{k} q\mathbf{1}\{x \in R_q\}$.

2.1 Classification and Regression Trees

A classification tree [3] is an algorithm that forms regions in the predictor space by recursively dividing it, more precisely, this procedure performs recursive binary partitioning. Beginning with the entire predictor space, the algorithm selects the variable to split upon and the location of the split that minimizes some impurity measure. Then the resulting two regions are each split into two more regions until some stopping rule is applied. The classifier will label each region with one of the k possible classes.

The traditional labelling process goes as follows; let $p_{rq} = \frac{1}{n_r} \sum_{x_i \in R_r} \mathbf{1}\{y_i = q\}$, the proportion of the class q in the region r where n_r is the number of observations contained in region r. Then, the label of the region r is the majority class in that region, i.e. if $x \in R_r$, $h_S(x) = \text{argmax}_q(p_{rq})$. For regression trees, the output mean within a leaf node is used as prediction for observations that belong in that node. The impurity measure function for region r is defined as Q_r and can take many forms such as the *Gini index*, the *deviance* or the *misclassification error*. For regression trees, the *mean squared error* is one possible region impurity measure.

When splitting a region into two new regions R_r and R_t the algorithm will compute the total impurity of the new regions ; $n_r Q_r + n_t Q_t$ and will pick the split variable X_j and split location s that minimizes that total impurity. If the predictor X_j is continuous, the possible splits are of the form $X_j \leq s$ and $X_j > s$ which usually results in $n_r - 1$ possible splits. For a categorical predictor having c possible values, we usually consider all of the $2^{c-1} - 1$ possible partitions.

The partitioning continues until a stopping rule is applied. In some cases, the algorithm

stops whenever every terminal node of the tree contains less than β observations, in other cases it stops when all observations within a region belong to the same class. To prevent overfitting, a deep tree is built and then the tree can be pruned. Tree-pruning is a cost-complexity procedure that relies on considering that each leaf, region, is associated with a cost α. The procedure begins by collapsing leaves that produce the smallest increase in total impurity and this technique will collapse leaves as long as the increase in impurity is less than the cost α of the additional leaf. The α parameter can be determined by cross-validation or with the use of a validation set.

3 Background about missing values

As described in the previous section, a standard assumption in data analysis is that all observations are distributed according to the true data generation distribution $\mathcal{D}$. We could think of the missingness itself as a random variable $\mathbf{M}$ also of dimension m that is distributed according to some missingness generating distribution which is a part of $\mathcal{D}$, i.e. $\mathbf{X} \times \mathbf{M} \times \mathbf{Y} \sim \mathcal{D}$. Formally, if $\mathbf{M}$ represents the missingness of the vector of predictors $\mathbf{X}$ it means that $M_j = 1$ if X_j is observed and $M_j = 0$ if X_j is missing.

Three different dependence relationships between $\mathbf{M}$ and $\mathbf{X}$ were defined by Rubin [16] and by Little and Rubin [14]. Seaman [18] later untangled the many definition inconsistencies of these relationships. In here, we will rely on simple definitions for an easy understanding of the structure we will consider. First, missing completely at random (MCAR) is the simplest dependence structure we consider: $\mathbf{M} \perp \mathbf{X}$.

Second, missing at random (MAR) is much more complicated; it essentially means that the missingness $\mathbf{M}$ is independent of missing observations but can still depend on observed predictors. More rigorously, we define $\mathrm{X}^o = \{x_{ij} \in S | x_{ij} \text{ is observed}\}$ as the set of all observed predictors value, and $\mathrm{X}^{na} = \{x_{ij} \in S | x_{ij} \text{ is missing}\}$ as the set of missing predictors value. We say that data is MAR if the distribution of the missingness is conditionally independent of missing values given observed values : $\mathbf{M} \perp \mathrm{X}^{na} | \mathrm{X}^o$. As pointed by Seaman [18], MAR has not always been used consistently and the definition above is the one we settled on for this project.

Finally, if the missingness $\mathbf{M}$ depends on missing values $\mathbf{M} \not\perp \mathrm{X}^{na}$, we say that the data is missing not at random (MNAR). We will see that the dependence relationship between $\mathbf{M}$ and $\mathbf{X}$ has a considerable effect on the efficiency of the many missing values techniques that exist. In

the next few sections we will establish under which of these relationship our proposed algorithm outperforms the current techniques.

3.1 Missing values techniques for decision trees

In order to handle missing values, a wide variety of solutions have been proposed for classification trees. Recent surveys ([17, 6, 9, 21]) define in detail most of the techniques that are currently used and compare them using various simulated and real data sets. Some techniques are only suitable for training, some for prediction and finally, some can deal with missing value in both. Under the assumption that both the observations obtained for training and the observations that need to be predicted are distributed according to the same true data generating distribution $\mathcal{D}$, we would like to use a technique that can handle missing values for both training and prediction.

Predictive value imputation (PVI) methods are popular approaches to deal with missing value. They aim at estimating the missing value and impute them within both the training and the test set. The simplest imputation consists of replacing the missing values with the mean for numerical predictors or the mode for categorical predictors. More advanced prediction models have also been proposed, such as linear model, k-nearest neighbours or expectation-maximization (EM). These models use the known predictors to impute values for the missing ones, therefore one weakness arise if the predictors are independent as these approaches will have close to no predictive power. Even though Gavankar [9] and Saar-Tsechansky and Provost [17] raise other problems concerning those techniques, they tend to perform well when there exist correlation between the predictors. Twala [21] demonstrated using simulated data sets the great performances of EMMI [13], an expectation-maximization based imputation algorithm that produces multiple imputations and aggregates the results.

The surrogate variable (SV) approach [3] is a special case of predictive value imputation. As explained in [11], during the training process, when considering a predictor for a split, only the observations for which that predictor is not missing are used. After the primary predictor and split point have been selected, a list of surrogate predictors and split points is constructed. The first surrogate split is the predictor and split point pair that best mimic the split of the training data achieved by the primary split. Then the second surrogate split is determined among the leftovers predictors and so on. When splitting the training set during the tree-building procedure or when sending and observation down the tree during prediction, the surrogate splits are used

in order if the primary splitting predictor value is missing. Many articles ([7, 6, 17, 21]) showed that the results are not satisfactory in many cases and Kim and Loh [12] noted the variable selection biased caused by this approach.

The Separate Class (SC) method replaces the missing value with a new value or a new class for all observations. For categorical predictors we can simply define *missing value* as a category on its own and for continuous predictors any value out of the interval of observed value can be used. This technique is proved to be the best by Ding and Simonoff [6] when there is missing values in both the training and the test set and when observations are missing not at random (MNAR). Twala et al. [22] also came up with similar results with a generalization of the separate class method named Missing Incorporated in Attribute (MIA).

The popular C4.5 implementation [15] has its own way to manage missing data, defined as a distribution-based imputation (DBI). When selecting the predictor to split upon, only the observations with known values are considered. After choosing the best predictor to split upon, observations with known values are split as usual. Observations with unknown values are distributed among the two child nodes proportionately to the split on observed values. Similarly, for prediction, a new test observation with missing value is split intro branches according to the portions of training example falling into those branches. The prediction is then based upon a weighted vote among possible leaves.

Finally, reduced-feature models are suggested by Saar [17] when missing values appear only in the prediction process. This technique relies on using only know predictors of the new observation we are trying to classify. A tree is built using only the know predictors of the new observation. If multiple observations contain different missing pattern then multiple trees are built to classify the various observations. It shares a great deal of similarities with lazy decision trees [8] as both models tailor a classifier to a specific observation and uses only known predictors to do so.

4 Branch-Exclusive Splits Trees (BEST)

We now introduce the proposed algorithm, BEST. The purpose of BEST is to utilize the tree structure itself in order to manage some missing data or some special structure among predictors.

As we explained in section 2.1, a classification tree aims at partitioning the predictor space and labelling the resulting regions. CART does so by looking through all the possible splits

and selecting the one that minimizes some prespecified error measure. When using BEST some predictors are available to split upon only within some regions of the predictor space. These regions are defined according to other predictors values, for example, predictor X_l could be only available for the partitioning process in the region defined by $X_j < 0$. Therefore if BEST selects X_j as the splitting variable and if s, the splitting value selected, is such that $s \leq 0$, then the predictor X_l would be available for partitioning in the subspace defined by $X_j < s$. From a tree perspective, some split variables are exclusive to some branches in the classification tree. By doing so, predictors with missing values can be handled easily as BEST will partition the data according to that predictor only in regions where it does not contain missing value. Similarly, some insight on the data structure can be used to force some variable to be partitioned upon before others. The result is a tree-structured classification model where certain split variables are branch-exclusive. All of the construction described below could be used for regression trees as well.

Therefore, our proposed algorithm differs from imputation methods as it only uses known information to build the classifier instead of using prediction to replace missing values. It also differs from reduced-feature models as it not only uses the known values but also utilize the fact that we know some predictors are missing instead of discarding this information. Finally, our algorithm shares similarities with separate class models as they can both lead to the same tree structure. On the other hand, BEST identifies the missing pattern using other predictors rather than including this information about missingness within the predictor containing missing value. Doing so, our approach leads to more interpretable results but also the ability to identify the importance of the missingness itself.

4.1 Motivating Example

Let us now explain which data structure BEST is suitable for by introducing the motivating data set. It contains information regarding the academic performances of students. The data set was provided to us by the Univeristy of Toronto and was first introduced and analysed by Bailey et al. [1]. It was later analysed by Beaulac and Rosenthal [2] where the goal was to predict whether or not a student would complete its program. The predictors represent the number of credits and grades obtained in all the departments during the first two semesters. Understanding the importance of these predictors was also a question raised by the authors. Obviously a student

did not obtain grades in many departments as he can only register to a limited number of courses within a year. In this situation, many grade variables were missing for every student. BEST handles that problem by considering the averaged grade obtained in a department only for students who took courses in that department. For example, BEST will force the classification tree algorithm to split upon the *Number of credits* predictors to begin. Then, suppose *Number of credits in Statistics* is selected and 2 is the split point for the partitioning, BEST will then allow splits on the *Grade in Statistics* predictor for the group of students in the region defined by *Number of credits in Statistics* > 2. Therefore, the *Number of credits* variables are used to define the region where the respective *Grade* variables are available for the partitioning process, we define *Number of credits* as *gating variables.*

If a data set contains missing values on predictor X_j but no predictor can help define the region with no missing value, we can add a new predictor X_{m+1} to the model as our *gating variable.* This new predictor is a dummy variable such that $X_{i,m+1} = 0$ if $X_{i,j}$ is missing and 1 if not. Doing so, we effectively add M_j as defined in section 3, as a predictor in the model and thus will be defined as follows in the rest of the text. Then, BEST will only consider splitting on X_j in the subspace defined by $M_j = 1$. Multiple dummy variables are added to the model if multiple predictors contain missing values. Doing so allows us to analyse the individual importance of the missing patterns M.

4.2 Theoretical intuition

Formally, the loss of a classifier h is defined as :

$$L_{\mathcal{D}}(h) = \mathbf{P}_{\mathcal{D}}[h(x_i) \neq y_i], \tag{1}$$

which is the probability under the true data generating distribution $\mathcal{D}$ that the classifier h misclassifies an observation x_i. Since the data generating distribution $\mathcal{D}$ is unknown, the empirical loss computed with the data set S is typically used as an estimator of the true loss :

$$L_S(h) = \frac{|\{i \in [n] : h(x_i) \neq y_i\}|}{n}, \tag{2}$$

which is the proportion of misclassified observations in the training set S. Usually a set of models $\mathcal{H}$, the hypothesis class [19], is selected in advance and most learning algorithms are trying to identify the classifier $h \in \mathcal{H}$ that minimizes the empirical loss $L_S(h)$. The true loss can be decomposed in a manner to observe a bias-complexity tradeoff. Suppose $h_S = \underset{h \in \mathcal{H}}{\operatorname{argmin}} L_S(h)$, then :

$$\begin{aligned} L_D(h_S) &= \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h) + (L_D(h_S) - \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h)). \\ &= e_{\text{app}}(\mathcal{H}) + e_{\text{est}}(h_S). \end{aligned} \tag{3}$$

The approximation error, $\min_{h \in \mathcal{H}} L_{\mathcal{D}}(h) = e_{\text{app}}$, is the minimum achievable loss within the hypothesis class. The second term , $(L_D(h_S) - \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h)) = e_{\text{est}}$, is the estimation error and is caused by the use of the empirical loss instead of the true loss when selecting the best classifier h. Since the goal is to minimize the total loss a natural tradeoff emerges from equation 3. A vast, large and complex hypothesis class $\mathcal{H}$ leads to a wider choice of functions and therefore reduces e_{app}, but the classifier is more prone to overfitting, which increases e_{est}. Inversely, a small hypothesis class $\mathcal{H}$ reduces e_{est} but increases e_{app}.

Our proposed algorithm aims at obtaining a better classifier by restricting the hypothesis class to a smaller one without increasing the approximation error. Suppose $\mathcal{H}_T$ is defined as the set of all tree-structured classifiers. Then, BEST is a new algorithm that aims to find the best classifier in a new hypothesis class $\mathcal{H}_B$ that contains all the tree-structured classifiers that respect a set of conditions regarding the order that variables can be partitioned upon. Therefore, we have $\mathcal{H}_B \subset \mathcal{H}_T$. Since the complexity of $\mathcal{H}_B$ is smaller than the complexity of $\mathcal{H}_T$ the estimation error of BEST will be smaller. Next, let us take a look at the approximation error : $\min_{h \in \mathcal{H}_B} L_{\mathcal{D}}(h)$. When using BEST, we make multiple assumptions on how the partitioning should be processed. For example, we assume it is better to partition the data using the missing indicator M_j before partitioning the data using X_j. Doing so, we assume that the best tree-structured classifier among all classification tree $\mathcal{H}_T$ is contained within the set of tree-structured classifiers that respect the partition ordering that defines $\mathcal{H}_B$. In other words, we assume $\underset{h \in \mathcal{H}_T}{\operatorname{argmin}} L_D(h) \in \mathcal{H}_B$. Suppose S is a data set, $h_S(\mathcal{H}_T)$ is the classifier that minimizes the empirical loss on $\mathcal{H}_T$ and $h_S(\mathcal{H}_B)$ is the classifier that minimizes the empirical loss on $\mathcal{H}_B$, then :

$$
\begin{aligned}
L_D(h_S(\mathcal{H}_T)) &= \min_{h \in \mathcal{H}_T} L_{\mathcal{D}}(h) + e_{\text{est}}(h_S(\mathcal{H}_T)). \\
&= \min_{h \in \mathcal{H}_B} L_{\mathcal{D}}(h) + e_{\text{est}}(h_S(\mathcal{H}_T)). \\
&\geq \min_{h \in \mathcal{H}_B} L_{\mathcal{D}}(h) + e_{\text{est}}(h_S(\mathcal{H}_B)). \\
&= L_D(h_S(\mathcal{H}_B)),
\end{aligned}
\tag{4}
$$

which implies that the under the assumption we have made we would not only naturally manage missing values but also reduce the loss. If our assumption $\underset{h \in \mathcal{H}_T}{\text{argmin}} L_D(h) \in \mathcal{H}_B$ is false, we might increase the loss, and the assumption itself is impossible to verify. Therefore, the behaviour of the algorithm under multiple scenarios will be tested in section 5 with simulated data.

4.3 Algorithm

Let us now explain how the algorithm functions. BEST takes as input the full data set S, the tuning parameter β and a list containing the predictor availability structure V. First S is set as the root node, the first set of observations to go through the following steps. The algorithm verifies a set of conditions before proceeding with the partitioning process. The first condition (C1) is that the region contains more than β observations, this is the main stopping rule. Then, the next condition (C2) is that the observations in the region have different labels; this condition makes sure that the algorithm has a reason to partition the data. Finally, the last condition (C3) is that at least one of the available predictors takes different values among the observations in the region; this is to guarantee that the algorithm can actually partition the data.

If at least one condition is false, then the region is defined as a leaf node, a label is assigned to that leaf for prediction purposes and the partitioning process is stopped. Usually the class that represents the majority in a leaf node is selected as label for that region, but one could define different label assignment rules.

If all the conditions (C1, C2 and C3) are respected then the partitioning process begins. The algorithm will go through all the available predictors. For a predictor j, the algorithm will go through all the possible partition s of the node with respect to the predictor j and will compute the total impurity of the resulting two regions $n_{r_1}Q_{r_1} + n_{r_2}Q_{r_2}$. Any region impurity measure Q can be used. BEST then selects the predictors j and the split s that minimizes the total

impurity and create two children regions by splitting the data according to s.

The last step is to update the list of available predictors for the children regions. There exist multiple possible structures that can contain this information but within the R-language we have settled on a list V where $V[0]$ represents the set of predictors available for the partitioning process in the root node and $V[i]$ for $i \in 1, ...m$ represent the update for the predictors available on the children node after the data has been partitioned using the ith predictor. More specifically, $V[0]$ is a vector of size m where $V[0][j] = 1$ if the jth predictor is available to be split upon in the root node and $V[0][j] = 0$ otherwise. Suppose now that the jth predictor is selected as the splitting predictor. If j is a continuous predictor, $V[j][0]$ contains a threshold value, if the splitting point s is greater (or less) than the threshold, then the child node containing the observations such that $X_j > s$ ($X_j \leq s$) have their available predictors updated according to the vector contained in $V[j][1]$ for example. If j is a categorical predictor, then $V[j]$ contains a matrix where the ith line represents the update needed on the node containing the observations such that $X_j = i$ after the partitioning. This formatting was easy to implement in R since lists are very flexible, but other structure could have been used.

Here is a pseudo-code of the proposed algorithm :

Algorithm : BEST(S,β,V)

1. Starting with the entire data set S and the set of available predictors $V[0]$.
2. Check conditions (C1, C2 and C3).
3. **if** (any condition is false) :
   Assign a label to the node and exit.
   else if (all conditions are true) :
   for (j in all available predictors):
   for (s in all possible splits) :
   Compute total impurity measure.
   Select the variable j and the split s with minimum impurity measure.
   Split the node into two children nodes.
   Update the available predictors for both children nodes using $V[j]$.
   Repeat steps 2 & 3 on the two children nodes.

In addition, the resulting tree can be pruned, and constructed with any splitting rule, any stopping rule and any label assignment rule. Since one of the goals of this new algorithm is

to produce accurate but also interpretable models we did not discuss forests so far, but the proposed tree construction procedure can be used to build any type of forest.

5 Experiments : Simulated data sets

In the following set of simulations, we will compare 6 methods; (1) the Distribution Based Imputation (DBI) proposed by Quinlan [15], (2) a simple single variable imputation (SVI), either the mode for a categorical predictor or the mean for a numerical one, (3) a refined predictive value imputation (PVI) using know predictors; EM for numerical predictors and multinomial logistic regression for categorical one [23], (4) the separate class (SC) approach, (5) the surrogate variable technique introduced by Breiman et al. [3] and finally (6) BEST, our proposed approach. Since the Reduced-Feature Model was the least accurate in every single experiment we have done, we decided not to include it in the following tables to improve readability. Multiple Imputation methods were also left out as they actually create forests.

We will simulate data sets with different missing patterns, details on the data structures are included in the respective subsections. Our experimental procedure is the following; to begin, we will generate a data set with a missing pattern. Then, we will fit a pruned decision tree using each of the six methods mentioned in the previous paragraph. This procedure will be repeated 50 times for every training data size we have established. The generated data set will include either 100, 500, 2000, 5000 or 20000 training observations and will contain 500 validation observations and 2000 testing observations. We will examine the averaged accuracy of the 50 decision trees produced by the 6 compared missing values management techniques for every experimental set up. The data is generated according to a small decision tree of depth 4 with 4 different labels. In every leaf nodes, 70% of observations are assigned to the right labels and 30% are randomly assigned to other labels.

5.1 MAR : Missingness depends on observed predictors

This first experimental set up is meant to test the missing pattern structure BEST was designed for. In this set up, the missing pattern of a predictor is fully explained by another, fully observed, predictor. A binary predictor X_i was designated as gating variable for a continuous predictor X_j. When $X_i = 0$, X_j is missing, otherwise X_j is generated according to its distribution. The data is generated according to a decision tree where X_j is only utilized in branches such that

$X_i = 1$.

# of obs	100		500		2000		5000		20000	
Methods	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.
DBI	0.4118	0.0553	0.5546	0.0322	0.5963	0.0169	0.6179	0.0107	0.6345	0.0092
SVI	0.4099	0.0509	0.5531	0.0330	0.5962	0.0169	0.6179	0.0107	0.6345	0.0092
PVI	0.4025	0.0502	0.5384	0.0299	0.5893	0.0171	0.6138	0.0143	0.6300	0.0096
SC	0.4088	0.0514	0.5539	0.0319	0.5962	0.0171	0.6178	0.0107	0.6345	0.0092
Surrogate	0.3875	0.0636	0.5611	0.0440	0.6032	0.0214	0.6158	0.0216	0.6316	0.0125
BEST	0.4112	0.0651	0.5688	0.0362	0.6302	0.0147	0.6382	0.0145	0.6434	0.0092

Table 1: Mean accuracy and standard deviation when a predictor is MAR given the other predictors.

This simulation highlights the great behaviour of BEST under an ideal scenario. As we can see from the mean accuracy columns, every method have similar performances with few training observations. BEST is outperforming every other missing data handling methods on data set of size 2000 and 5000 by a large margin. For these intermediate size training set the data is sufficient for a good fit and the restricted set of models utilized by BEST seems to contribute towards more accurate classifiers. Finally, for the large data set simulation, BEST still outperforms other algorithm but the performance gap has shrunk down a lot.

5.2 MAR : Missingness depends on the response

According to Ding et al. [6], the relationship between the missing pattern and the response variable has a great effect on the results obtained from different missing value treatments. The relation between the predictor with missing value and the response is also important since the less correlated they are the more information we relatively gain from analysing the missing pattern. In this simulation, one of the predictors is randomly selected, let us say X_j, every iteration and the censoring process is then applied. The censoring process goes as follows; one of the four response labels is randomly selected, and X_j is missing for all observation with that selected label. Table 2 contains the results when X_j is continuous and table 3 when X_j is categorical. In this experiment we have used a dummy variable as the *gating variable* for the BEST algorithm.

# of obs	100		500		2000		5000		20000	
Methods	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.
DBI	0.4591	0.0699	0.6105	0.0301	0.6649	0.0267	0.6805	0.0216	0.681	0.0176
SVI	0.6289	0.0423	0.713	0.0298	0.7772	0.0144	0.7907	0.0102	0.7972	0.0079
PVI	0.458	0.0732	0.6024	0.0299	0.6573	0.0271	0.6737	0.0262	0.675	0.0222
SC	0.6388	0.0505	0.7324	0.0253	0.7854	0.0104	0.7945	0.0106	0.798	0.0079
Surrogate	0.4303	0.0735	0.6183	0.0249	0.6596	0.0185	0.6697	0.0202	0.6687	0.0184
BEST	0.6555	0.0517	0.7432	0.0236	0.7885	0.0115	0.7927	0.0106	0.7954	0.0102

Table 2: Mean accuracy and standard deviation when a continuous predictor is MAR given the response.

# of obs	100		500		2000		5000		20000	
Methods	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.
DBI	0.4297	0.0608	0.5827	0.0567	0.6229	0.0691	0.646	0.0638	0.6479	0.0625
SVI	0.544	0.1204	0.6866	0.0853	0.7293	0.0738	0.7348	0.0626	0.7345	0.0578
PVI	0.4202	0.0601	0.5731	0.0532	0.6135	0.0655	0.6384	0.0632	0.6459	0.0565
SC	0.6312	0.0603	0.7538	0.0204	0.7897	0.0124	0.7972	0.0082	0.7992	0.0095
Surrogate	0.4154	0.0528	0.5964	0.0466	0.6261	0.052	0.6337	0.0493	0.6375	0.05
BEST	0.6392	0.0594	0.7595	0.0225	0.7934	0.0103	0.7931	0.0109	0.7966	0.0115

Table 3: Mean accuracy and standard deviation when a categorical predictor is MAR given the response.

In this scenario, the missing pattern is actually a variable with predictive power and therefore, models like BEST and SC shine as they utilize the fact that there is missing values instead of trying to impute them. BEST and SC approaches have similar results and their performances is way higher than any other techniques except SVI which will be discussed next. It seems like using a variable strictly to represent the missing pattern grants BEST the ability to utilize this new predictor with fewer observation as BEST outperforms any other techniques in the small training size simulations. The SC approach seems to have better performances on larger data sets as it can utilize the missing pattern like BEST without restricting itself to a pre-specified partitioning order. Finally, it is interesting to notice the high performances of the simple single value imputation. Our experiments revealed that when the predictor containing missing value is continuous, replacing missing values with the mean actually behave like the separate class approach because only the missing values will exactly take the value of the mean. If the predictor with missing value is categorical, replacing missing values with the mode will make the observation with missing value undistinguishable from observations that truly belong to that class which explains why SVI is less competitive in table 3.

5.3 MNAR : Missingness depends on missing values

Let us now proceed with simulation when the predictors are missing not at random. In the first simulation a continuous predictor, let us say X_j, is randomly selected. Then, a random value t which serves as threshold within the domain of X_j is selected. Finally, a Bernoulli variable b is drawn. If $b = 0$, then if $X_j < t$ it is set missing, otherwise if $b = 1$ then X_j is missing if its value is greater than t. In the second one, a categorical prediction, X_j is selected. Then a subset of categories is randomly selected. X_j is considered missing for observations for which X_j is one of the selected categories. Since the missingness of X_j depends on the value of X_j itself, this is considered MNAR.

# of obs	100		500		2000		5000		20000	
Methods	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.
DBI	0.4451	0.0791	0.6101	0.0397	0.6603	0.0418	0.659	0.0411	0.6737	0.036
SVI	0.4399	0.0765	0.6044	0.0385	0.6589	0.0357	0.6703	0.0312	0.683	0.0275
PVI	0.4356	0.0784	0.6038	0.0399	0.6503	0.0426	0.6553	0.0429	0.6708	0.0375
SC	0.4479	0.0744	0.6127	0.0316	0.6675	0.0333	0.6714	0.0302	0.6843	0.0275
Surrogate	0.4302	0.069	0.6181	0.0268	0.6491	0.0333	0.6505	0.0298	0.6626	0.0312
BEST	0.4683	0.0683	0.6229	0.034	0.6808	0.0242	0.6746	0.0263	0.6779	0.0279

Table 4: Mean accuracy and standard deviation when a continuous predictor is MNAR.

# of obs	100		500		2000		5000		20000	
Methods	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.
DBI	0.4636	0.0712	0.5678	0.0749	0.6103	0.0825	0.6175	0.0768	0.6245	0.0682
SVI	0.4609	0.0675	0.5541	0.0644	0.5852	0.0665	0.6102	0.0714	0.6172	0.0628
PVI	0.4586	0.0665	0.5548	0.0613	0.5914	0.0714	0.6114	0.0732	0.6221	0.066
SC	0.4921	0.0709	0.6396	0.0362	0.6822	0.0143	0.6918	0.0118	0.6995	0.0086
Surrogate	0.4622	0.0688	0.605	0.0519	0.6369	0.046	0.632	0.051	0.6309	0.0535
BEST	0.5108	0.0537	0.659	0.026	0.6845	0.0189	0.6821	0.0294	0.6915	0.0146

Table 5: Mean accuracy and standard deviation when a categorical predictor is MNAR.

Once again we observe that BEST slightly outperforms every other techniques with few observations. It seems that the more data available the less guidance the decision tree needs.

5.4 Random forests and variable importance

Let us now build a small example where random forests are used to analyse the variable importance. Random forests are popular in exploratory analysis [20] as the variable importance tools that were developed for this model became popular.

As we have seen in the previous experiments, when BEST performs well, so does the SC approach as both of these techniques do not impute missing values. Here we will quickly discuss how BEST produces more accurate variable importance computations than the SC approach. We have created an example where the missing pattern depends on the response, used either the SC approach or BEST to handle missing values and we have built a forest with those trees.

When the values for a predictor are conditionally missing at random given the response, the missing pattern is itself a good predictor. We would like a variable importance analysis that distinguishes between the importance of the predictor with missing value, say X_j, from the importance of its missing pattern M_j. A random forest of trees built under the SC approach would fail to distinguish between the effect of the observed value for that predictor and the effect of the missing pattern. Since BEST actually uses a variable to define the region with missing values, either with another predictor or a user-created dummy variable, this gating variable importance will better represent the predictive power of the missing pattern.

Data	X_1	X_2	X_3	X_4	$\mathbf{X_5}$	X_6	X_7	X_8	$\mathbf{M_5}$
Complete	25.74	34.15	20.24	30.33	**4.98**	78.81	46.43	45.73	-
SC	31.83	43.18	28.17	33.15	**155.12**	69.53	39.40	35.09	-
BEST	30.31	40.89	26.10	33.02	**3.09**	64.56	45.04	35.72	**145.84**

Table 6: Variable Importance table : Computed using the GINI decrease importance

We have built a random forest using the complete data set and computed the GINI decrease importance. Then we have randomly selected one of the four labels, and the predictor X_5, a predictor of low importance according to the GINI decreases under the complete data set, is rendered missing depending on the value of the response. Since the SC approach uses the predictor containing missing value to identify observations containing missing value then it identifies X_5 as the most important predictor. Using BEST, we can easily observe that the missing pattern, M_5 is the important predictor and that X_5 is actually of low importance when observed as it should be according to the complete data variable importance. We do believe that this is a benefit from using BEST over the SC approach.

5.5 Simulations takeaways

Throughout various simulation experiments we have been able to highlight the success of BEST under various scenarios. It seems that forcing the algorithm to consider the missing pattern

before the variable with missing values is useful with small to medium data size. As the data size grows larger, the imputations are more precise which helps imputation-based techniques. Regarding the SC approach, since it does not impute the missing values, it can create the same partitioning that BEST creates but the SC approach does not need to first isolate the missing value in order to partition upon the variable containing missing values which can sometimes be valuable on large data sets.

Our simulation revealed that BEST suffers from a weakness when the gating variable is of low importance. This can happen if only a small proportion of data is missing, if the missing pattern is simply non-informative or in some cases when data is MCAR. In that case, BEST will never partition upon the gating variable and thus will never partition upon the branch-exclusive variable which will almost surely reduce the accuracy of the resulting tree. This weakness is intrinsic to the algorithm as it is caused by the greedy nature of decision trees. Since it can only see the reduction in impurity gained with a single partition, a classic decision tree approach cannot perceive the accuracy gained by the combination of two successive partitioning. Considering pairs of consecutive splits would be a great improvement and would negate this limitation but would drastically increase the run time of the algorithm.

6 Experiments : grades data set

The data set mentioned in section 4.1 was analysed using BEST. Once again, the accuracy of the proposed algorithm is compared to other techniques that handle missing values. To begin, we predict if a student completes its program using its first year of courses and results. The data set contains 38842 observations. Our set of predictors consists of the number of credits attempted in all the departments and the average grade obtained in those respective departments. The number of credits is a numerical variable that serves as the gating variable for the respective average grade. If the number of credits attempted in a department is greater than 0 for every observation in a region then BEST acquires access to the grade variable. We have randomly sampled training sets of different sizes and used all the remaining observations to assess the accuracy. This process was repeated 50 times and we have averaged the results. We did not include the single value imputation because we expect this technique to produce the same result as the SC approach since all predictors are numerical. We did not include the imputation produced by the mice package [23] as the package was incapable of handling the data set.

# of obs	5000		10000		15000		20000	
Methods	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.
DBI	0.7223	0.0035	0.7336	0.0038	0.7385	0.0092	0.7402	0.0095
SC	0.7307	0.0066	0.7387	0.0044	0.7427	0.0036	0.7462	0.0033
Surrogate	0.7291	0.0053	0.7301	0.0046	0.7311	0.0034	0.7300	0.0032
BEST	0.7333	0.0062	0.7424	0.0045	0.7457	0.0037	0.7479	0.0033

Table 7: Mean accuracy and standard deviation when a categorical predictor is MNAR.

In table 7, we observe that BEST produces the most accurate decision trees for that data set for all training data sizes. This data set also provides a good example for the increased interpretability of the BEST classifier as we previously discussed. For the SC approach we have replaced missing grades by a value outside of the domain, 101. Frequently in this experiment, the tree constructed under the SC approach partitions upon the grade in departments before the number of credits. For example, if *Grades in Mathematics* is selected and 60 is selected as the split point, then the partitioning is of the following form; *students with grades below 60 in Mathematics* are partitioned from *students with grades above 60 and students with missing grades*. What interpretable information is to be understood from that partition ? It is hard to say. Does this partition implies students with no experience in Mathematics behave similarly to students with good results in Mathematics ? BEST achieves similar or higher accuracy while keeping the partitions logical and interpretable. BEST will begin by partitioning students who attempted at least 1 credit in Mathematics from those who did not. Then, among students who attempted at least 1 credit in Mathematics, BEST will partition them according to their grades, which leads to a more interpretable sequence of partitions. If interpretability is considered a strength of decision trees, then BEST is better than the SC approach at preserving this strength.

Another reason why we might prefer using BEST in this analysis is its ability to rightfully identify the variable importance. As we discussed in section 4.1, the researchers were interested in the importance of the predictors. Therefore BEST is an improvement as it truly identifies the importance of the gating variables, the number of credits, as we have shown in section 5.4. In this case we were able to distinguish the importance of the number of credits in a department of the importance of the grade obtained in that department.

In the second analysis, we will look at the 26488 students who completed their program. Using the same set of explanatory variables, we will try to predict the department they majored

in :

# of obs	5000		10000		15000		20000	
Methods	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.
DBI	0.3581	0.0055	0.376	0.0043	0.3871	0.0039	0.3936	0.0057
SC	0.3992	0.0079	0.4172	0.0069	0.4262	0.0047	0.4332	0.0044
Surrogate	0.3639	0.0121	0.3661	0.011	0.3723	0.012	0.3742	0.012
BEST	0.3952	0.0074	0.4164	0.0051	0.4265	0.0043	0.4319	0.0043

Table 8: Mean accuracy and standard deviation when a categorical predictor is MNAR.

Here we observe closer results from the two best performing algorithm. BEST and SC have almost indistinguishable performances and are the top performers. Even though the results are a lot closer between BEST and SC, our proposed algorithm still produces trees that are more interpretable and could be used to produce a non-biased variable importance analysis.

7 Conclusion

We have constructed a modified tree-building algorithm that lets the users decide the regions of the predictor space where variables are available for the data partitioning process. We have focused on using this feature to manage missing values. BEST has the elegant property of analysing a variable only when values are known without assuming any missingness dependence structure. It produces highly interpretable trees and achieves higher accuracy than most missing value handling techniques in cases we have identified using simulated data sets. Even though BEST shares similarities with the separate class technique, BEST leads to a more accurate variable importance analysis and produces more interpretable and intuitive trees.

BEST suffers from a weakness when the gating variable has no predictive power. In those cases, the algorithm will never choose to split upon the gating variable and thus will never be allowed to use the branch-exclusive variable. This problem can lead to a decrease in accuracy in some simple cases where the data is MCAR. Fortunately, as we have previously discussed, there already exist multiple techniques to handle data MCAR and we can count on cross-validation in order to help us select the best missing data handling technique. Nonetheless, in the simulated experiments we have performed, results were mostly positive as BEST outperforms other techniques when data is MAR and MNAR and when the data size is not too large.

The results produced by BEST were also satisfactory when the algorithm was used on the real motivating *grades data set*. We were able to achieve higher accuracy than with most other techniques while obtaining a more interpretable classifier. Since variable importance was a concern in the *grades data set* analysis, BEST was an improvement as it answers that research question by providing a more reliable variable importance analysis than the separate class approach previously used [2].

Acknowledgements

We are very grateful to Glenn Loney and Sinisa Markovic of the University of Toronto for providing us with the anonymised students grade data. The authors also gratefully acknowledge the financial support from NSERC of Canada.

References

[1] Michael A. Bailey, Jeffrey S. Rosenthal, and Albert H. Yoon. Grades and incentives: assessing competing grade point average measures and postgraduate outcomes. *Studies in Higher Education*, 41(9):1548–1562, 2016.

[2] C. Beaulac and J. S. Rosenthal. Predicting University Students' Academic Success and Choice of Major using Random Forests. *ArXiv e-prints*, February 2018.

[3] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA, 1984. new edition [**?**]?

[4] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[5] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[6] Yufeng Ding and Jeffrey S. Simonoff. An investigation of missing data methods for classification trees applied to binary response data. *J. Mach. Learn. Res.*, 11:131–170, March 2010.

[7] A. J. Feelders. Handling missing data in trees: Surrogate splits or statistical imputation. In *PKDD*, 1999.

[8] Jerome Friedman, Ron Kohavi, and Yeogirl Yun. Lazy decision trees. 1, 09 1997.

[9] S. Gavankar and S. Sawarkar. Decision tree: Review of techniques for missing values at training, testing and compatibility. In *2015 3rd International Conference on Artificial Intelligence, Modelling and Simulation (AIMS)*, pages 122–126, Dec 2015.

[10] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine Learning*, 63(1):3–42, Apr 2006.

[11] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2 edition, 2009.

[12] H. Kim and W.-Y. Loh. Classification trees with unbiased multiway splits. *Journal of the American Statistical Association*, 96:589–604, 2001.

[13] Joseph L. Schafer and Maren K. Olsen. Multiple imputation for multivariate missing-data problems: A data analyst's perspective. 33, 07 2000.

[14] Roderick J. A. Little and Donald B. Rubin. *Statistical Analysis with Missing Data*. John Wiley & Sons, Inc., 2002.

[15] J. Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[16] Donald B. Rubin. Inference and missing data. *Biometrika*, 63(3):581–592, 1976.

[17] Maytal Saar-Tsechansky and Foster Provost. Handling missing values when applying classification models. *J. Mach. Learn. Res.*, 8:1623–1657, December 2007.

[18] Shaun Seaman, John Galati, Dan Jackson, and John Carlin. What is meant by âĂIJmissing at randomâĂİ? *Statist. Sci.*, 28(2):257–268, 05 2013.

[19] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, New York, NY, USA, 2014.

[20] Carolin Strobl, Anne-Laure Boulesteix, Achim Zeileis, and Torsten Hothorn. Bias in random forest variable importance measures: Illustrations, sources and a solution. *BMC Bioinformatics*, 8(1):25, 2007.

[21] Bhekisipho Twala. An empirical comparison of techniques for handling incomplete data using decision trees. *Appl. Artif. Intell.*, 23(5):373–405, May 2009.

[22] Bhekisipho Twala, M.C. Jones, and David Hand. Good methods for coping with missing data in decision trees. 29:950–956, 05 2008.

[23] Stef van Buuren and Karin Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in r. *Journal of Statistical Software*, 45(3):1–67, 2011.