Optimal Survival Trees With the Iterative Breslow Estimator and the Integrated Brier Score Objective

Izzy van der Giessen[1]

Supervisor(s): Jacobus G. M. van der Linden[1]**, Emir Demirović**[1]

[1]**EEMCS, Delft University of Technology, The Netherlands**

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 22, 2024

Name of the student: Izzy van der Giessen
Final project course: CSE3000 Research Project
Thesis committee: Jacobus G. M. van der Linden, Emir Demirović, David Tax

An electronic version of this thesis is available at http://repository.tudelft.nl/.

Abstract

Survival analysis predicts survival functions that give the probability of survival until a given time. Many applications of survival analysis involve health care, which requires interpretability of the models used to predict the survival function. Provably optimal decision trees have shown to be an interpretable alternative to so-called black box models. However, these algorithms often choose estimators that are fast, yet not necessarily most accurate. Moreover, the objective functions of optimal decision tree algorithms tend to make (possibly incorrect) assumptions about the survival function.

In this paper, we tackle both problems. We implement the iterative Breslow estimator in an already existing optimal survival tree algorithm in order to iteratively improve the Nelson-Aalen estimator. This approach has great potential, as we show by using it on artificial datasets, but we do not see an improvement in accuracy on real world data. To eliminate the assumptions made by the objective function, we implement the Integrated Brier Score objective, which causes a significant improvement on training accuracy. However, we see no improvement on out-of-sample accuracy.

1 Introduction

Survival analysis analyzes the expected time of 'death' for a given subject, where death refers to the occurrence of a one-time event. Medical problems intuitively fit such descriptions of death very well; one could think of biological death, but also recovery from a disease, as a one-time event. Although survival analysis is often used in medical fields, this is not its only application. For example, the failure of a machine and recidivism are also one-time events that can be analyzed. In contrast to regression, survival analysis does not try to predict labels for instances, but survival functions that give the probability of surviving past a time t.

Machine learning can be used to create high accuracy survival models [7], but the problem of many machine learning approaches is that they produce uninterpretable black box models [6], while interpretability is crucial for many fields, especially health care [21]. For this reason, decision trees are often the preferred option for survival analysis. An example of a survival tree, a decision tree adapted to survival analysis, is shown in figure 1.

The interpretability of decision trees decreases as the number of nodes in the tree grows, so the trees used in survival analysis are generally sparse trees. It has been shown that sparse survival trees can be as accurate as black box models for tabular datasets [22]. Greedy heuristics have also been used to create decision trees [4], [16], but they do not guarantee optimality: any decision in a current node does not take into account the decision that its children will make.

Optimal decision trees have optimal accuracy on the training data, given a maximum depth or a maximum number of nodes. It has been shown that optimal decision trees outperform regular decision trees not only on training data, but also on out-of-sample data [9]. By exploiting the structure of the tree, dynamic programming can be used to create optimal decision trees within seconds [21], [18].

Huisman et al. present SurTree [12], an adaptation of MurTree [9] for survival analysis. SurTree is optimal with regard to its objective function. The objective function used in SurTree depends on the proportional hazards model. This model assumes that the hazard functions of all instances are proportional to each other. The hazard function gives the rate of death for any time. Its integral is the cumulative hazard function (CHF). Consequently, CHFs are also assumed to be proportional. As we show in this paper, this assumption of proportionality does not suit all types of datasets.

The CHF is used to calculate the survival function, so it is important to estimate it well. SurTree uses the Nelson-Aalen estimator to do this. This is an estimator that weighs every instance in the dataset equally. Although this makes the Nelson-Aalen estimator relatively fast, it may be less accurate than an estimator that gives a (unique) weight to each instance.

We propose two changes to the SurTree algorithm: 1) we replace the objective function by the Integrated Brier Score, which does not depend on the proportional hazards model, and 2) we replace the Nelson-Aalen estimator by the iterative Breslow estimator.

The Integrated Brier Score (IBS) is the integration of the Brier Score. The Brier Score is similar to the mean squared error. In contrast to the objective function used by SurTree, the IBS does not use the proportional hazards model, and therefore does not make assumptions about the hazard functions. This is advantageous for datasets that do not follow the assumption of proportionality [2].

The Breslow estimator can be used in an iterative process to incrementally improve the Nelson-Aalen estimate [16], but, to the best of our knowledge, it has not been used for optimal survival trees yet.

In this paper, we compare the accuracy of the iterative Breslow estimator to the Nelson-Aalen estimator. We show that the iterative Breslow estimator creates more accurate trees than the Nelson-Aalen estimator on artificially constructed datasets, but leads to no significant improvement on real data. We also analyze the difference in accuracy between the IBS and the partial likelihood loss function used by SurTree, and conclude that using the IBS as the objective function can lead to significant improvements on artificial datasets, but does not increase the out-of-sample accuracy, and comes at the cost of higher runtimes.

Our contributions are:

1. An explanation of the type of dataset for which the iterative Breslow estimator would give a better estimate of the cumulative hazard function than the Nelson-Aalen estimator.
2. An implementation of the iterative Breslow estimator in the SurTree algorithm.
3. Experiments on this implementation with real and artificial data.

4. An explanation of the type of dataset for which the Integrated Brier Score objective gives significantly different scores than the partial likelihood objective.
5. An implementation of the Integrated Brier Score objective in the SurTree algorithm.
6. Experiments on this implementation with real and artificial data.

First, we discuss related work in section 2. In section 3, we introduce the terminology and mathematics upon which this research was based. In sections 4 and 5, we explain the main contributions of this paper. In section 6, we show how different configurations of our new estimator and objective function compare to each other and to the original SurTree algorithm. In section 7, we discuss the reproducibility and ethical implication of our research. Finally, in section 8, we interpret the results, and in section 9 we conclude this paper and propose areas for further research.

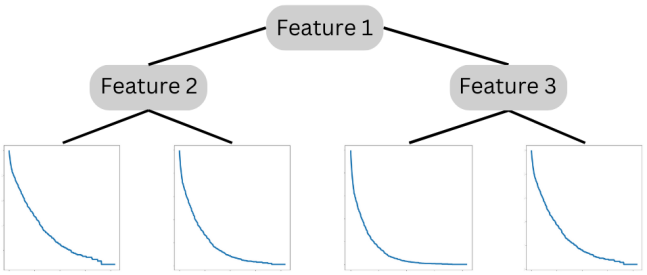


Figure 1: Example of a survival tree

2 Related Work

Survival Analysis The goal of survival analysis is to find a good survival function for a dataset. In the process of gathering data, it may occur that a subject leaves the observation group before the one-time event has happened. Such cases are called right-censored. The time at which their observation started is known, but the time-of-event is unknown. The Kaplan-Meier method is an early estimator of the survival function that takes censored data into account [15]. Nelson and Aalen developed an indirect estimator of the survival function that estimates the cumulative hazard function first. Their method of estimating the cumulative hazard function is similar to the Kaplan-Meier method.

Meanwhile, Cox proposed a proportional hazards model and a regression algorithm based on this model [8]. Inspired by Cox's proportional hazards model, Breslow created a new estimator of the cumulative hazard function [5] which can be seen as a proportional hazards version of the Nelson-Aalen estimator. More recently, many machine learning techniques have been used for survival analysis, such as random forests and decision trees [13], [12].

Decision Trees Decision trees have often been created using greedy heuristics [4]. In order to limit the size of the tree to avoid having small sets of instances in each node, a cost-complexity metric was introduced that penalizes tree growth [4]. LeBlanc and Crowley adapt this greedy approach by introducing Cox's proportional hazards model into their survival tree algorithm and estimating the risk coefficients of the model with a maximum likelihood estimator [16]. Recently, Bertsimas et al. proposed Optimal Survival Trees, a coordinate descent algorithm to find locally optimal trees [2]. By repeating the algorithm, they increase the probability of finding a good tree, but there is no guarantee of global optimality.

Optimal Decision Trees Many approaches towards finding optimal decision trees have been tried, such as constraint programming [20], (maximum) satisfiability [14], [11], and Mixed Integer Programming [3]. However, these methods struggle with large datasets and medium-sized trees. On the other hand, dynamic programming algorithms have undergone big runtime improvements. A major focus of these dynamic programming approaches is to branch and bound in order to speed up the search [21], [9].

Zhang et al. created an algorithm to compute optimal sparse regression trees, and used equivalent points to calculate a tighter lower bound [22]. They extend this algorithm for survival analysis and call it Optimal Sparse Survival Trees (OSST) [21]. OSST has the option of using reference models that are believed to make similar mistakes as the optimal survival tree. However, using these reference models means that the resulting tree is no longer guaranteed to be optimal.

Demirović et al. contributed the similarity-based lower bounding approach, which uses solutions to previously computed subproblems with similar data, while guaranteeing optimality [18]. This algorithm was adapted for survival analysis by Huisman et al. [12], and it was named SurTree. SurTree uses the proportional hazards model and the Nelson-Aalen estimator to estimate a survival function in each leaf node.

3 Preliminaries

In this section, we define terms and notation used throughout this paper, and we explain fundamental concepts upon which our research builds.

Instances (i) in datasets (D) have a label, and a *feature vector* with features. We only consider binary features and binary decision trees. The labels consist of two values: the *time-of-event* (t_i) and the *censoring indicator* (δ_i). The censoring indicator is a binary value stating whether time t_i signifies censoring ($\delta_i = 0$) or the time-of-event ($\delta_i = 1$). Events can be *left-censored* or *right-censored*. Left-censored means that the starting time is unknown. Right-censored means that the time-of-event is unknown. In this study, we only consider right-censored events.

The goal of survival analysis is to find the *survival function* $S(t)$ that gives the probability for a subject to survive past a given time: $P(T > t)$, where T is the true time-of-event. We can estimate the survival function with the Kaplan-Meier estimator [15]:

$$\hat{S}(t) = \prod_{t' \leq t} (1 - \frac{d(t')}{n(t')}) \tag{1}$$

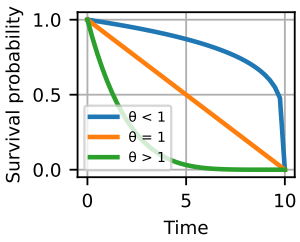


Figure 2: The effect of θ on the survival function

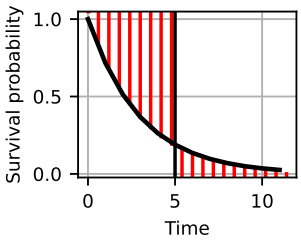


Figure 3: Intuition of area of loss for IBS

Where:

$$d(t) = \sum_{i \in D, t_i = t} \delta_i \tag{2}$$

$$n(t) = \sum_{i \in D, t_i \geq t} 1 \tag{3}$$

The opposite of the survival function is the *death distribution*: $F(t) = 1 - S(t)$. The derivative of the death distribution is the *death density function*: $f(t) = \frac{\mathrm{d}}{\mathrm{d}t}F(t)$. Using the death density function and the survival function, we can calculate the *hazard function*, which gives the rate of an event happening at a given time, provided that it has not occurred before: $\lambda(t) = \frac{f(t)}{S(t)}$. Since the death density function can be written as $f(t) = -\frac{\mathrm{d}}{\mathrm{d}t}S(t)$, the hazard function can be rewritten as follows:

$$\begin{aligned} \lambda(t) &= \frac{f(t)}{S(t)} \\ &= -\frac{\frac{\mathrm{d}}{\mathrm{d}t}S(t)}{S(t)} \\ &= -\frac{\mathrm{d}}{\mathrm{d}t} ln(S(t)) \end{aligned} \tag{4}$$

When we integrate this hazard function, we get

$$\Lambda(t) = -ln(S(t)) \tag{5}$$

which is the *cumulative hazard function* (CHF). Now we can write $S(t) = e^{-\Lambda(t)}$. Following the assumption of proportionality [8], we can also write this as

$$S_i(t) = e^{-\theta_i \Lambda_0(t)} \tag{6}$$

for any instance i, where Λ_0 is the baseline CHF, shared by all instances. The assumption of proportionality says that the CHF of every instance is proportional to the baseline CHF: $\Lambda_i(t) = \theta_i \Lambda_0(t)$. The effect of the risk coefficient (θ_i) on the survival function is shown in figure 2.

Following SurTree [12], we want to estimate the risk coefficient for each leaf, and the baseline CHF for the whole tree. The baseline CHF can be estimated by the Nelson-Aalen estimator [17], [1]:

$$\hat{\Lambda}_0(t) = \sum_{t' \leq t} \frac{d(t')}{n(t')} \tag{7}$$

There is also a more general estimator similar to the Nelson-Aalen, which builds on the proportional hazards model and takes the risk coefficient of each instance into account. We call it the iterative Breslow estimator and the formula is as follows [5]:

$$\hat{\Lambda}_0^j(t) = \sum_{t' \leq t} \frac{d(t')}{\sum_{t_i \geq t} \theta_i^j} \tag{8}$$

If $\theta_i^j = 1$ for all i, this turns into the Nelson-Aalen estimator. We do this for $j = 0$ to get an initial estimate. Most useful about the Breslow estimator is that we can estimate the baseline CHF in multiple iterations j. To do this, we also need θ_i^j.

LeBlanc and Crowley show that we can derive the maximum likelihood estimator of the risk coefficient using the likelihood of a dataset, given the CHF and making the assumption of proportionality [16]. This likelihood is:

$$L = \prod_{i \in D} (\lambda_0(t_i)\theta_i)^{\delta_i} e^{-\Lambda_0(t_i)\theta_i} \tag{9}$$

This likelihood function has been derived from the likelihood function that does not make the assumption of proportionality:

$$L = \prod_{i \in D} \lambda_i(t_i)^{\delta_i} S_i(t_i) \tag{10}$$

The maximum likelihood estimator of θ_i, as was shown by LeBlanc and Crowley, is as follows.

$$\hat{\theta}_h = \frac{\sum_{i \in h} \delta_i}{\sum_{i \in h} \hat{\Lambda}_0(t_i)} \tag{11}$$

Where h is the set of instances in a leaf node. We can use this to calculate θ_i^j:

$$\hat{\theta}_i^{j+1} = \frac{\sum_{i \in h} \delta_i}{\sum_{i \in h} \hat{\Lambda}_0^j(t_i)} \tag{12}$$

The risk coefficient calculated over a dataset with only one instance is called the *saturated risk coefficient*. Following the formula above, the saturated risk coefficient for any instance i is:

$$\theta_i^{sat} = \frac{\delta_i}{\hat{\Lambda}_0(t_i)} \tag{13}$$

We can calculate the loss for any given instance by taking the difference between the log-likelihoods of the saturated risk coefficient and the estimated risk coefficient ($\hat{\theta}_i$). Huisman et al. show that the loss over a dataset D can be written in the following form:

$$\mathcal{L}(D, \hat{\theta}) = \sum_{i \in D} \hat{\Lambda}_0(t_i)\hat{\theta} - \delta_i \log \hat{\Lambda}_0(t_i) - \delta_i \log \hat{\theta} - \delta_i \tag{14}$$

This loss is used as the objective function in SurTree.

Other metrics can be used to calculate the loss of a tree. The Integrated Brier Score is of particular interest to us in this paper. As the name suggests, it is the integral of the Brier Score:

$$BS = \frac{1}{|D|} \sum_{i \in D} (1 - \hat{S}_i(t_i))^2 \tag{15}$$

$$IBS = \frac{1}{t_{max}} \frac{1}{|D|} \sum_{i \in D} \int_0^{t_i} \frac{(1 - \hat{S}_i(t))^2}{\hat{G}(t)} dt + \delta_i \int_{t_i}^{t_{max}} \frac{(\hat{S}_i(t))^2}{\hat{G}(t_i)} dt \tag{16}$$

Where $\hat{G}(t)$ is the censoring distribution estimated by the Kaplan-Meier estimator (equation 1). The Brier Score can be compared to the mean squared error. The Integrated Brier Score weighs the terms with $\hat{G}(t)$ or $\hat{G}(t_i)$ and takes censored data into account. Excluding the weights, figure 3 shows an intuition of the area of the integral. A perfect score would be achieved by a survival function that has a probability of 1 before the time of death t_i and a probability of 0 afterward:

$$S(t) = \begin{cases} 1 & \text{if } t < t_i \\ 0 & \text{if } t \geq t_i \end{cases}$$

The Concordance Index (CI) is an accuracy metric that takes all *comparable* pairs and calculates how many of them are *concordant*. A pair is comparable if it is certain that one of the subjects died before the other. This means that neither of them is censored, or one died before the censoring event of the other. Concordance means that the subject that died first of the pair is also the subject with the lower predicted risk (ρ). We can define *discordance* as the opposite. Pairs that are comparable, yet have the same risk, are counted among the tied risk pairs. The number of concordant, discordant and tied risk pairs are given by

$$CC = \sum_{i,j} I_{(t_i > t_j)} I_{(\rho_i < \rho_j)} \delta_j \tag{17}$$

$$DC = \sum_{i,j} I_{(t_i > t_j)} I_{(\rho_i > \rho_j)} \delta_j \tag{18}$$

$$TR = \sum_{i,j} I_{(t_i > t_j)} I_{(\rho_i = \rho_j)} \delta_j \tag{19}$$

and the Concordance Index is given by

$$CI = \frac{CC + \frac{TR}{2}}{CC + DC + TR} \tag{20}$$

4 Iterative Breslow estimator

In this section, we explain the theoretical advantages of the iterative Breslow estimator (equation 8) over the Nelson-Aalen estimator (equation 7).

Since the survival function is estimated according to $\hat{S}_i(t) = e^{-\theta_i \hat{\Lambda}_0(t_i)}$, it is crucial to get a good estimate of the baseline CHF ($\hat{\Lambda}_0$). The SurTree algorithm uses the Nelson-Aalen estimator, where every risk coefficient is implicitly set to 1 while estimating the baseline CHF. However, we can use equations 8 and 12 to iterate on the cumulative hazard function. This weighs every instance by the respective risk coefficient. The effect of these weights is shown in figure 5. Only the instance with t_i on the vertical line has a different risk coefficients, and all others are the same. We can see that the baseline CHF for $t \leq t_i$ is reduced, while it is not affected for $t > t_i$.

The value of the risk coefficient, as we see in equation 12, is determined by the cumulative hazard of instances in the leaf node (h). When all instances in the leaf node have a relatively low cumulative hazard at their time of death, their shared risk coefficient will be relatively high. Since low cumulative hazards are the consequence of early times of death, we can likewise say that leaf nodes with relatively early deaths have a relatively high risk coefficient. However, as we saw in figure 5, the risk coefficient of an instance with a low time of death has only a small effect on the baseline CHF, because the space $t \leq t_i$ is small.

Following these observations, we can conclude that the instance that has the most weight in the Breslow estimator is an instance with two properties: 1) it has a late time of death, and 2) the other instances in the same leaf node have an early time of death. Such an instance can be thought of as a subject that is expected to have an early death based on its features, yet still survives for a long time. In other words, this instance is an outlier. Although the Nelson-Aalen estimator treats these outliers the same as every other instance, the iterative Breslow estimator gives them a higher weight.

We have implemented the iterative Breslow estimator in SurTree. We start the algorithm with $\theta_i = 1$ for all $i \in D$. A baseline CHF is estimated using these initial risk coefficients, and an optimal survival tree is built. After the algorithm has found the optimal survival tree, it uses the risk coefficients calculated in the leaf nodes (the predicted labels) as input for equation 8. This generates a new baseline CHF that is used to build a new tree. The process is repeated for a variable amount of times.

5 Integrated Brier Score objective

In this section, we discuss the advantages of the Integrated Brier Score (equation 16) compared to the partial likelihood loss function (equation 14), and we explain how we have implemented the Integrated Brier Score as the objective function in SurTree.

Comparison with partial likelihood loss

The partial likelihood loss function is based on the proportional hazards model and the assumption of proportionality, which assumes that the CHFs of all instances are proportional. Proportionality of hazard functions is also assumed when we calculate the CHF in each leaf node. As we explain in section 3, we calculate the survival function according to $\hat{S}_i(t) = e^{-\theta_i \Lambda_0(t)}$. The saturated risk coefficient is taken as the optimal risk coefficient in the partial likelihood loss function, but it is only optimal if the assumption of proportionality holds. Whenever the assumption of proportionality does not hold, a perfect score is achieved with a suboptimal estimate of the risk coefficient.

Figure 4 shows two survival functions for a dataset with copies of one instance with the same time of death. We can see that the survival probability of both functions is $\frac{1}{e}$, which shows that the estimated risk coefficient is equal to the

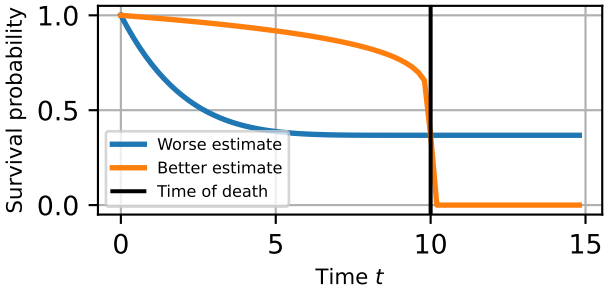


Figure 4: Two survival functions with perfect loss according to the partial likelihood loss function on a dataset with only instances at one time of death

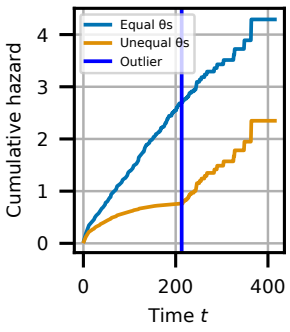


Figure 5: Exaggerated effect of one high theta value on a baseline CHF

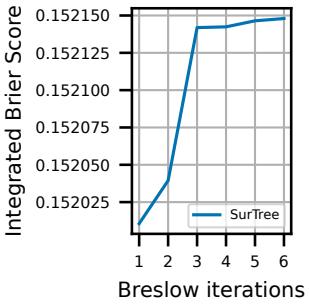


Figure 6: Integrated Brier Score per Breslow iteration on test data

saturated risk coefficient:

$$\begin{aligned} S_i^{sat}(t_i) &= e^{-\theta_i^{sat}\hat{\Lambda}_0(t_i)} \\ &= e^{-\frac{\delta_i}{\hat{\Lambda}_0(t_i)}\hat{\Lambda}_0(t_i)} \\ &= e^{-\delta_i} \end{aligned}$$

Since both functions have the same optimal risk coefficient, the only difference between the survival functions is the baseline CHF. It is clear from the dataset that the true probability of survival is high before the time of death, and then makes a sudden drop, but the partial likelihood loss does not distinguish between these two functions.

In contrast to the partial likelihood loss, the IBS looks at the survival probability before and after the instance in consideration. It gives a higher (worse) score for survival functions that have a low probability of surviving before the actual time of death. Consequently, the IBS has a clear preference for the high survival function with a sudden drop in figure 4.

Implementation in SurTree

A great contributor to the speed of SurTree and the underlying MurTree algorithm is the depth-two solver [9], which solves trees of depth two by splitting the loss function into multiple components and calculating the contribution of each instance to these components. In other words: "the cost of a leaf node can be expressed as a function over the contributions of individual instances." [18] We define the following terms as sums over the contributions of single instances:

$$ES = \sum_{i \in h} \delta_i \tag{21}$$

$$HS = \sum_{i \in h} \Lambda_0(t_i) \tag{22}$$

Now we can write the survival function as follows:

$$S_i(t) = e^{-\frac{ES}{HS}\Lambda_0(t)} \tag{23}$$

The loss can be calculated as follows:

$$\begin{aligned} IBS = \frac{1}{t_{max}} \frac{1}{|D|} \sum_{i \in h} \int_0^{t_i} \frac{(1 - e^{-\frac{ES}{HS}\Lambda_0(t)})^2}{\hat{G}(t)} dt + \\ \delta_i \int_{t_i}^{t_{max}} \frac{e^{-2\frac{ES}{HS}\Lambda_0(t)}}{\hat{G}(t_i)} dt \end{aligned} \tag{24}$$

In order to use this function in the depth-two solver, it has to be additive: the loss of the full tree has to be equal to the sum of the losses of the leaf nodes. When calculating the loss over the entire tree at once, without summing over the leaf node losses, $|D|$, t_{max} and $\hat{G}(t)$ are calculated over the whole dataset, D, and not just over h, the subset in the leaf node. Therefore, when calculating the loss of each leaf node, we have to make sure that these terms are still calculated over the entire dataset. We can pre-compute these values and reuse them in each leaf node.

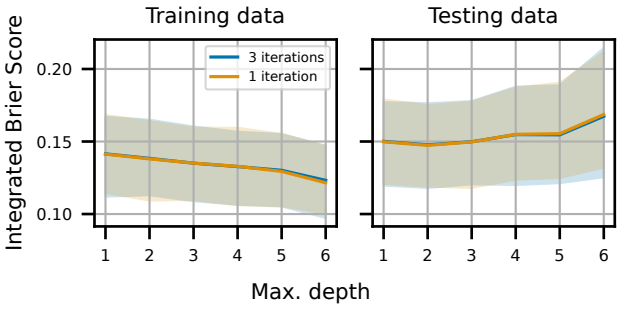
The time complexity of this function is $O(|h| * |D|)$. We go over very element of h in the sum, and use every unique time in D to compute the integral. Since we can pre-compute all values for $\hat{G}(t)$ and $\Lambda_0(t)$, the worst case time complexity is $O(|h| * |D|)$.

6 Experimental Setup and Results

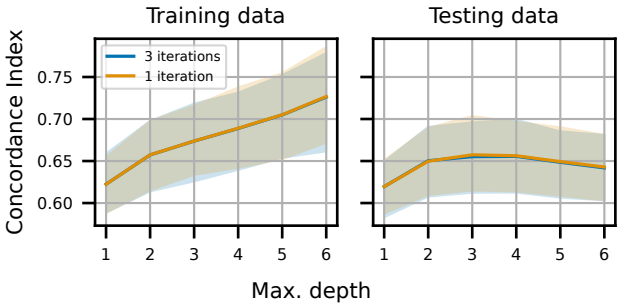
The results are divided into two parts, one for each of the two topics: the iterative Breslow estimator and the Integrated Brier Score objective. For both of these contributions, we show how its accuracy relates to the SurTree algorithm, and for the IBS objective experiments, we also compare to OSST [21], a survival tree algorithm that also uses the IBS as the objective function. The focus of our experiments is on accuracy, since more time can be spent optimizing the runtime of our contributions in future research.

Survival trees return a survival function as the classification result, and there are many ways to calculate the accuracy of such a function [2]. In order to give a good picture of the accuracy of a model, different kinds of metrics have to be used [19]. Following the metrics by Huisman et al. [12], we have decided to use the Concordance Index and the Integrated Brier Score to measure our accuracy. For both metrics, the accuracy for the resulting model is calculated for a list of datasets and the results are averaged. Unless stated otherwise, we use a maximum depth of 3 and a maximum number of nodes of 7.

Data We use real data from SurvSet [10] to perform our experiments. We binarize all feature data, because both SurTree and OSST work with binarized data. For categorical



(a) Average Integrated Brier Score on the SurvSet datasets, lower is better



(b) Average Concordance Index on the SurvSet datasets, higher is better

Figure 7: Accuracy of SurTree with 1 and 3 Breslow iterations

variables, we use one-hot encoding. Categorical variables are divided into ten categories. The nine most frequent variables are each one category, and all the others are put in the tenth category. We remove identical variables and variables that identify less than 1% of the data. For testing out-of-sample accuracy, we perform five-fold cross-validation. Unless stated otherwise, the IBS and Concordance Index are calculated for all datasets in SurvSet, and the average of these results is presented.

6.1 Iterative Breslow Estimator Results

Since the Nelson-Aalen estimator is the same as the iterative Breslow estimator with one iteration, we run our algorithm with one iteration to simulate the original SurTree algorithm. We run the algorithm with multiple iterations to see the results of the iterative Breslow estimator. In order to know how many iterations are enough for the iterative Breslow estimator, we test for convergence. Figure 6 shows the average IBS on trees of a maximum depth of 3 per Breslow iteration. We conclude that three iterations of the iterative Breslow estimator are enough to achieve convergence, since we no longer see real change in the IBS after more iterations. Henceforth, we will use three iterations in our experiments.

In figure 7, we see the average IBS and CI for one and three iterations. Neither on testing nor on training data, do we see that the algorithm with three iterations outperforms the algorithm with one in terms of the IBS. Concerning the Concordance Index, three iterations achieve slightly lower

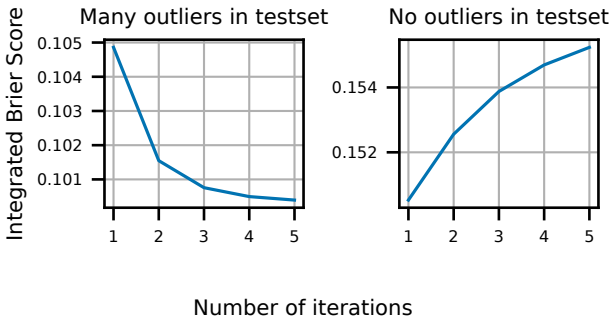


Figure 8: IBS of SurTree with varying Breslow Iterations on artificially constructed train set and test set

accuracy than one iteration. The p-value of the Wilcoxon signed rank test for the IBS results on depth 3 is 0.58.

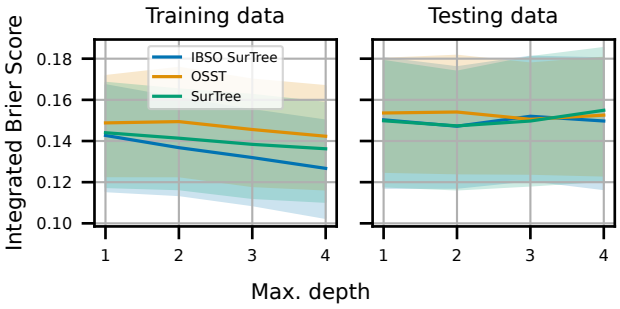
In section 4, we theorized that the iterative Breslow estimator would be better for outliers within leaf nodes. Although we do not see this improvement reflected in the accuracy of our model on real data, our hypothesis is confirmed when we run the algorithm on a dataset particularly constructed to show the strength of the iterative Breslow estimator.

We create a training set with copies of two different instances, i_1 and i_2. i_1 has a low time of death, and i_2 has a high time of death. We add a new instance i_3, whose feature vector is identical to the feature vector of i_1 to make sure it ends in the same leaf node, but the time of death is much higher, yet lower than the time of death of i_2. This represents an extreme version of the scenario sketched in section 4. We run our algorithm on the training set with a maximum depth of 1, so the resulting tree has two leaves, one for i_3 and the copies of i_1, and one for the copies of i_2. We also create two test sets. One test set has instances similar to i_1 and the other has instances similar to i_3. Figure 8 shows the results. We see that increasing Breslow iterations results in a better IBS on the test set with instances similar to the outlier, while it achieves worse results on the test set without outliers.

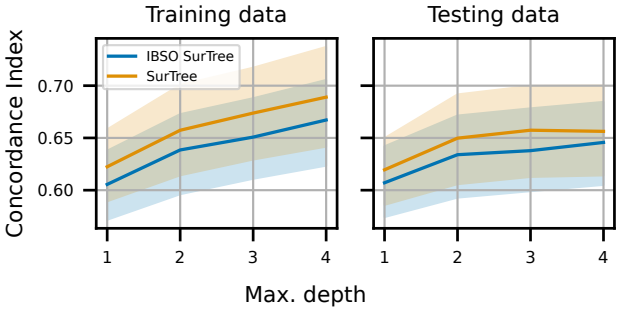
6.2 Integrated Brier Score Objective Results

We see in figure 9 that the IBS objective helps to achieve a better score on training data, as is expected. However, the test scores show that there is no clear improvement on out-of-sample data. In terms of the IBS metric, using the IBS as objective function produces worse trees than the partial likelihood objective for a maximum depth of three. The Concordance Index shows similar results. For every maximum depth that was used in the experiment, the original SurTree algorithm scored better than the SurTree with IBS objective algorithm.

Table 1 shows the runtime of SurTree with the IBS objective compared to the partial likelihood objective and OSST. It is clear from the runtimes that the IBS is much slower. Because of the slow runtime of IBS Objective SurTree, some datasets (*flchain*, *hdfail*, and *Framingham*) time out when the maximum depth is greater than 3. These datasets were consequently not used while calculating the



(a) IBS, lower is better



(b) Concordance Index, higher is better

Figure 9: Accuracy of SurTree with and without the IBS objective

results in figure 9. The table also shows a clear difference in runtime between OSST and IBS Objective SurTree.

In order to confirm the theoretical advantage of the IBS over the proportional likelihood loss as described in section 5, we test both methods on an artificial dataset. We set the maximum depth of the tree to 1, and use two features in the dataset. Our goal is to create a scenario where splitting on one feature seems best when considering the shape of the survival function as the IBS does, while splitting on the other feature seems best when assuming proportional hazards. In practice, this means that the first split results in a tree with two differently shaped survival functions in the two leaf nodes, while the second split results in survival functions with proportional hazards.

Since we assume proportionality while estimating the survival function, the only way to create a survival function in a leaf node that differs in shape from the others in the way described in section 5 is to add a set of censored instances with the same features and a similar time of censoring. When creating a dataset with this set of censored instances and two other sets of non-censored instances, we can construct a scenario where the IBS objective leads the algorithm to split the censored instances off into a separate node, while the proportional likelihood loss prefers to split the uncensored instances into two groups. On datasets like these, the IBS makes decisions that the proportional likelihood function would not have made, and achieves a lower IBS. However, the Concordance Index is also lower for the IBS objective algorithm.

7 Responsible Research

In this section, we reflect on the reproducibility and the ethical implications of our research.

7.1 Reproducibility

Our experiments were run as described in this paper. Wherever randomization was used during the experiment, we used a seed that can be found in our codebase. Anyone with access to the latest SurTree code will be able to reproduce all results found in this paper. All experiments were run on an AMD Ryzen 5 2600 processor. Although runtimes will differ for each subsequent run of the experiments, we believe that they will paint the same picture about the algorithms that we compared: OSST, SurTree, and IBSO SurTree.

We use the SurvSet dataset, as this was also used by Huisman et al. who created SurTree. When testing the IBSO SurTree algorithm, we did not experiment with maximum depths beyond 4, for the mentioned reason that the algorithm would time out. It is possible that results for depth 5 and higher would lead to new insights, but we do not think that it would affect the conclusion. We care most about the trees with lower depths, because these are more interpretable and attain better out-of-sample accuracy. We do not expect maximum depths above 5 to achieve higher out-of-sample accuracy than depths below 5.

While testing the IBS for different maximum depths for SurTree, OSST and IBSO SurTree, we left out the three datasets that timed out at a maximum depth of 4. We did include the results on these datasets in table 1. There we see that the datasets that were left out in the other experiment follow the same pattern as the other datasets. Therefore, we do not think that leaving them out significantly changed the final results.

Code availability The code to run our experiments, and a script to download the necessary datasets are available on GitHub[1]. Although we do not yet have permission to make the SurTree code public at the time of writing, it will be made public once we get permission. All code used to run the experiments set out in this paper, and the code to generate the plots, are already available.

7.2 Ethical implications

Survival trees can be used for unethical purposes, for example by insurance companies to make money from predicting someone's death. Improvement in the accuracy and runtime of optimal survival trees would likely increase the frequency of such unethical behavior. Despite its potential utility for unethical behavior, survival trees are most common in health care, where they are used to help patients. It is important in these scenarios that no life-altering decisions are made solely based on the results of machine learning algorithms that are not understood by the person taking the decision. Our survival trees achieve best out-of-sample accuracy with depths lower than 5, so they are relatively easy to interpret. This mitigates the risk of irresponsible use.

[1] https://github.com/IzzyVanDerGiessen/evalsurtree

	Runtime (seconds)			Integrated Brier Score		
	OSST	SurTree	IBSO SurTree	OSST	SurTree	IBSO SurTree
UnempDur	1	<1	72	**0.17**	0.18	0.18
flchain	<1	<1	8205	0.05	0.05	0.05
aids2	<1	<1	140	0.16	0.16	0.16
acath	<1	<1	21	0.11	0.11	0.11
rott2	1	<1	2169	0.19	**0.17**	0.18
nwtco	<1	<1	77	0.11	0.11	0.11
Framingham	3	<1	4934	0.13	**0.12**	**0.12**
dataDIVAT3	<1	<1	210	0.07	0.07	0.07
dataDIVAT1	<1	<1	151	0.16	**0.16**	**0.16**
Dialysis	1	<1	69	0.19	**0.18**	**0.18**
csl	1	<1	244	**0.18**	0.23	0.25
divorce	<1	<1	4	0.22	**0.21**	**0.21**
prostateSurvival	<1	<1	4	0.08	**0.07**	**0.07**
oldmort	1	0	1953	0.18	0.18	**0.17**
hdfail	2	0	9179	0.12	**0.11**	**0.11**

Table 1: IBS scores and runtime of SurTree, IBS Objective SurTree and OSST with maximum depth 3

8 Discussion

In figure 7a and figure 7b we saw that the IBS of our trees with three Breslow iterations is not better than the trees produced by the Nelson-Aalen estimator. Similarly, the Concordance Index does not show any improvement. The scenario where the iterative Breslow estimator stands out is one where the test set has many outliers while the training set does not. Since the training data used in our experiments was representative of the test data, we were not able to see an improvement in our results.

The IBS objective is significantly slower than the original partial likelihood objective of SurTree. This was expected, since the partial likelihood has a constant time function to compute the loss, while we use equation 24, which is $O(|h| * |D|)$ in the worst case. Compared to OSST, the difference in runtime is also significant. Since OSST was specifically created with the IBS objective in mind, it has many runtime improvements in its implementation. However, the OSST code is poorly documented, and redesigning SurTree to fit the OSST implementation of the IBS objective would not be worthwhile for the scope of our experiments, which focused on accuracy.

In section 5, we discussed the advantage of the IBS over the partial likelihood objective when the assumption of proportionality is not appropriate for a dataset. However, this advantage is hard to notice when this assumption is still made while estimating the survival functions. On artificially constructed data, the IBS objective achieves a higher IBS, but also a lower Concordance Index. This mirrors the results we saw on real data.

9 Conclusions and Future Work

We have extended the SurTree optimal survival tree algorithm with an option to use multiple iterations in estimating the cumulative hazard function. This is done by using the iterative Breslow estimator, which incrementally improves the estimate of the cumulative hazard function with each iteration. We have shown that this has theoretical advantages over the Nelson-Aalen estimator by creating artificial datasets that exaggerate the difference between the two estimators. On these datasets, SurTree with the iterative Breslow estimator has proven to achieve higher accuracy than SurTree with the Nelson-Aalen estimator. On real data, we see no improvement in training accuracy and out-of-sample accuracy resulting from three iterations of the Breslow estimator.

Furthermore, we have experimented with the Integrated Brier Score as objective function. This score has an advantage over the partial likelihood objective used by SurTree, since it does not make the assumption of proportionality. We think that the Integrated Brier Score would be useful as an objective function when working with datasets for which the assumption of proportionality is not appropriate. However, with these datasets, it would be best to use a different method of estimating the survival function, because the method used in SurTree assumes proportional hazards. In general, the Integrated Brier Score objective in SurTree does not achieve a higher out-of-sample accuracy than the proportional likelihood objective.

We suggest further research into pre-processing and manipulating datasets in order to get closer to the exaggerated theoretical datasets that we have created, as we would expect SurTree with the iterative Breslow estimator to achieve higher accuracy on real data if this can be achieved. In general, it would be worthwhile to research the real world use cases of the iterative Breslow estimator in optimal decision tree algorithms.

The Integrated Brier Score objective does not achieve higher out-of-sample accuracy than the partial likelihood objective, but we think that it can still be useful in combination with a survival function estimator that does not assume proportionality of hazard functions. We think it would be helpful to experiment with an algorithm that takes the best scoring estimate of two estimates per leaf node: one that uses the proportional hazards model, and one that does not. Moreover, there is a lot of room for improving the

runtime of the IBS objective implementation. We suggest ordering the datasets on the time-of-event to allow faster computations of the IBS.

References

[1] Odd Aalen. Nonparametric Inference for a Family of Counting Processes. *The Annals of Statistics*, 6(4):701–726, 1978.

[2] Dimitris Bertsimas, Jack Dunn, Emma Gibson, and Agni Orfanoudaki. Optimal survival trees. *Machine Learning*, 111(8):2951–3032, 2017.

[3] Matheus Guedes Vilas Boas, Haroldo Gambini Santos, Luiz Henrique de Campos Merschmann, and Greet Vanden Berghe. Optimal decision trees for the algorithm selection problem: integer programming based approaches. *International Transactions in Operational Research*, 28(5):2759–2781, 2021.

[4] L. Breiman, J. Friedman, R.A. Olshen, and C.J. Stone. *Classification and Regression Trees*. Chapman and Hall/CRC, 1984.

[5] N. Breslow. Contribution to the discussion of paper by D. R. Cox. *Journal of the Royal Statistical Society, Series*, 34:216–217, 1972.

[6] Diogo V. Carvalho, Eduardo M. Pereira, and Jaime S. Cardoso. Machine Learning Interpretability: A Survey on Methods and Metrics. *Electronics*, 8(8), 2019.

[7] Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. Recurrent Neural Networks for Multivariate Time Series with Missing Values. *Scientific reports*, 2018.

[8] D. R. Cox. Regression Models and Life-Tables. *Journal of the Royal Statistical Society: Series B (Methodological)*, 34(2):187–202, 1972.

[9] Emir Demirović, Anna Lukina, Emmanuel Hebrard, Jeffrey Chan, James Bailey, Christopher Leckie, Kotagiri Ramamohanarao, and Peter J. Stuckey. MurTree: Optimal Classification Trees via Dynamic Programming and Search. *Journal of Machine Learning Research*, 23(26), 2022.

[10] Erik Drysdale. SurvSet: An open-source time-to-event dataset repository, 2022. arXiv preprint arXiv:2203.03094.

[11] Hao Hu, Mohamed Siala, Emmanuel Hébrard, and Marie-José Huguet. Learning Optimal Decision Trees with MaxSAT and its Integration in AdaBoost. In *IJCAI-PRICAI 2020, 29th International Joint Conference on Artificial Intelligence and the 17th Pacific Rim International Conference on Artificial Intelligence*, 2020.

[12] Tim Huisman, Jacobus G. M. van der Linden, and Emir Demirović. Optimal Survival Trees: A Dynamic Programming Approach. In *Proceedings of AAAI-24*, 2022.

[13] Hemant Ishwaran, Udaya B. Kogalur, Eugene H. Blackstone, and Michael S. Lauer. Random survival forests. *The Annals of Applied Statistics*, 2(3):841 – 860, 2008.

[14] Mikoláš Janota and António Morgado. SAT-Based Encodings for Optimal Decision Trees with Explicit Paths. In Luca Pulina and Martina Seidl, editors, *Theory and Applications of Satisfiability Testing – SAT 2020*, pages 501–518. Springer International Publishing, 2020.

[15] E. L. Kaplan and Paul Meier. Nonparametric Estimation from Incomplete Observations. *Journal of the American Statistical Association*, 53(282):457–481, 1958.

[16] Michael LeBlanc and John Crowley. Relative Risk Trees for Censored Survival Data. *Biometrics*, 48(2):411–425, 1992.

[17] Wayne Nelson. Theory and Applications of Hazard Plotting for Censored Failure Data. *Technometrics*, 14(4):945–966, 1972.

[18] Jacobus van der Linden, Mathijs de Weerdt, and Emir Demirović. Necessary and Sufficient Conditions for Optimal Decision Trees using Dynamic Programming. *Advances in NeurIPS-23*, 2023.

[19] Iulii Vasilev, Mikhail Petrovskiy, and Igor Mashechkin. Sensitivity of Survival Analysis Metrics. *Mathematics*, 11(20), 2023.

[20] Hélène Verhaeghe, Siegfried Nijssen, Gilles Pesant, Claude-Guy Quimper, and Pierre Schaus. Learning optimal decision trees using constraint programming. *Constraints*, 25:226–250, 2020.

[21] Rui Zhang, Rui Xin, Margo Seltzer, and Cynthia Rudin. Optimal Sparse Survival Trees. arXiv preprint arXiv:2401.15330.

[22] Rui Zhang, Rui Xin, Margo Seltzer, and Cynthia Rudin. Optimal Sparse Regression Trees. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.