Generating Globally Optimized Multivariate-Split Survival Trees using GP-GOMEA

Author:
Kenzo Boudier
Student number: 5333024

To obtain the degree of Master of Science in:
Computer Science, track Artificial Intelligence,
at the Delft University of Technology,
Faculty of Electrical Engineering, Mathematics, and Computer Science.

Thesis committee:

Chair:	Prof. Dr. Mathijs M. de Weerdt, TU Delft
Core member:	Dr. David M.J. Tax TU Delft
Advisor:	Prof. Dr. Peter A.N. Bosman, CWI, TU Delft
Advisor:	Dr. Tanja Alderliesten, LUMC
External Advisor:	MCs. Thalea schlender, LUMC

Preface

This thesis marks the culmination of my 5-year journey at the Delft University of Technology. When I started my Bachelor of Computer Science in 2020, I never expected to earn a Master's in Computer Science five years later. Throughout this process, I was never once alone. Over the years, I have grown from all the wonderful encounters I have had. I am truly grateful for the opportunity given by TUDelft and would like to thank all for my personal and academic growth.

I would like to thank Prof. Dr. Mathijs M. de Weerdt, who accepted to be my chair on a short delay. I would like to express my gratitude to MSc Thalea Schlender, who guided me through the entire process, always providing me with valuable assistance and insight. Additionally, I would like to thank Prof. Dr. Peter Bosman and Dr. Tanja Alderliesten for their supervision throughout the project.

I would like to thank my family and friends, who have always been there for me. I would like to extend a special thanks to my CWI fellow master's students for making this experience less intimidating and helping me through the process, despite the productivity bar not always being at its highest. Finally, I would like to thank my partner, who has helped me throughout the entire journey, through thick and thin, always having my back.

Kenzo Boudier
November 17, 2025

Abstract

Survival trees are a statistical modeling technique used to predict the time until an event occurs. They are widely valued for their interpretability, as they allow practitioners to understand how different variables influence outcomes. However, traditional survival trees struggle to capture nonlinear relationships and rely on greedy splitting strategies, which limit their performance in complex settings.

This thesis proposes a novel approach that addresses these limitations by generating globally optimized survival trees using Genetic Programming Gene-pool Optimal Mixing Evolutionary Algorithm (GP-GOMEA). By integrating a state-of-the-art evolutionary algorithm into the tree construction process, the resulting survival trees optimize both the structure and the decision nodes at a global level.

The method was evaluated on a synthetic dataset designed to require nonlinear decision boundaries—the XOR problem. Our approach consistently outperformed traditional survival trees, achieving optimal or near-optimal results in the noise-free setting. Moreover, the results show that GP-GOMEA survival trees can maintain a high performance even with a smaller population size and limited data, demonstrating the method's suitability for problems involving nonlinear interactions.

These findings suggest that GP-GOMEA survival tree is a promising direction for advancing survival tree methodology. Future work should include evaluating the method on real-world survival datasets and further tuning key hyperparameters, such as the number of decision nodes.

Contents

List of Figures

Chapter 1

Introduction

Cancer is one of the leading causes of death worldwide, responsible for millions of deaths every year [47]. Survival analysis can play an essential role in supporting clinical decisions. Survival analysis allows for the estimation of patient outcomes, the identification of key risk factors, and the tailoring of treatment strategies accordingly.

With the increasing adoption of artificial intelligence (AI) in healthcare, clinicians have access to powerful tools that can assist in these tasks. However, the use of AI models must be approached with caution; clinicians must understand the model's predictions to ensure trust and transparency. This necessity has given rise to explainable AI (XAI), which aims to make AI decisions transparent and interpretable.

1.1 Problem Statement

Survival analysis plays a critical role in domains such as medicine, where predicting time-to-event outcomes is essential. Survival analysis aims to have the most accurate survival function. This function predicts the probability that the time of an event T is later than time t. The survival function is formally denoted as S and defined as $S(t) = P(T > t)$. Survival analysis deals with survival data. A distinguishing feature of survival data is the presence of censoring [22, 14]. In this work, only right-censoring is considered—cases where the starting date is known; however, the event of interest has not been observed. Right-censoring can arise for various reasons, such as when a study concludes before the event occurs, a participant withdraws, or follow-up data is lost. The presence of censored data poses challenges for traditional predictive models, such as those minimizing the mean squared error, as they are ill-suited to handle incomplete event information. Due to the scarcity of survival data, it is essential to incorporate censored observations, as it leads to more accurate models.

A survival analysis model that handles right-censored data well is the Cox proportional hazard model, also known as the Cox regression model [6]. The Cox regression model is the most widely used survival analysis model and has been a standard model in clinical research. The model aims to relate the time-to-event outcome to one or more *predictor variables*. Predictor variables, also known as covariates in the clinical domain, can refer to factors such as age, treatment group, or genetic markers. The Cox regression model estimates *hazard ratios*, which quantify how a change in a predictor variable affects the outcome relative to the baseline. This model is widely used as it is interpretable through the hazard ratio. Despite these advantages, the Cox regression model rests on assumptions that limit its use in more complex scenarios. One of these assumptions is that it assumes a linear relationship between covariates and the hazard. Consequently, when effects are nonlinear, this model fails to work [20]. More flexible approaches, like survival trees, have been developed to address these limitations.

Survival trees are an alternative model that can capture nonlinear relationships and interactions between predictor variables without requiring a parametric form [13]. They are binary decision trees specifically designed for survival analysis. They aim to maximize the difference in survival between groups by splitting across decision nodes. Survival trees use a univariate split, where decision nodes typically take the form $x_i > t$, comparing a single feature x_i against a threshold t. The tree structure itself makes results easy to visualize. Their ability to capture nonlinear relationships and ease of interpretability are the main reasons survival trees are frequently used in survival analysis. Nonetheless, survival trees have some noticeable drawbacks.

Their first drawback is that single survival trees are unstable and tend to underperform [35]. A remedy for this drawback is to use an ensemble method, such as the Random Survival Forest (RSF) [19], which aggregates predictions over many survival trees. Due to ensemble averaging, RSF reduces variance and is less prone to overfitting, improving robustness. However, RSFs do not solve all the drawbacks of survival trees. In particular, the quality of the splits determines a tree's performance. In traditional survival trees, splitting is done iteratively, starting from the root node, by an iterative greedy local heuristic. The local iterative nature of survival trees is an issue, as it hinders the capture of global traits. While this limitation is barely noticeable in simpler problems, it becomes problematic in more complex scenarios involving nonlinear interactions between multiple variables. For example, traditional survival trees often struggle with problems that exhibit such interactions, such as the exclusive OR (XOR) problem. Survival trees would be trapped in local optima, resulting in poor predictive performance, or they would need to grow large to capture nonlinear interactions, which would inhibit interpretability. Noise can exacerbate these limitations, significantly degrading split quality.

Some state-of-the-art survival trees try to remedy such a limitation using dynamic constraint programming [18]. While this approach partially solves the issue, the splits are still univariate, meaning these trees must be significant to capture nonlinear interactions between predictor variables. Another approach uses an evolutionary algorithm (EA)[24] to optimize the splits, which encounters a similar issue. The resulting Globally Induced Survival Tree (GIST) uses global learning to optimize the survival tree. In that study, what is unique is that the EA optimizes the splits and structure. While both methods can optimize the tree structure and content of the decision nodes, the decision nodes remain univariate. This means these trees still struggle if there is a nonlinear relation between multiple predictor variables. This research aims to define a new type of tree capable of overcoming the aforementioned limitations and featuring a multivariate split to prevent the formation of large survival trees, as well as being able to identify nonlinear relationships between the predictor variables.

To address the limitations of traditional survival trees, the proposed approach integrates symbolic regression (SR) to optimize the tree splitting. SR searches for explicit mathematical expressions that best describe patterns in the data, thus capturing the global structure of the data. The search and optimization for such expressions is performed by the gene-pool optimal mixing evolutionary algorithm (GOMEA)[42], adapted for genetic programming (GP)[44], known as GP-GOMEA.

GP-GOMEA evolves multiple candidate expressions jointly while exploiting learned dependencies between predictor variables, referred to as the *genotype* in EAs. In this framework, each expression represents a mathematical function that defines a decision split. Collectively, the set of expressions determines the structure of the survival tree. By jointly constructing and optimizing these multivariate expressions, GP-GOMEA is capable of capturing complex, nonlinear relationships within the data. As a result, GP-GOMEA circumvents the iterative, greedy nature of conventional survival trees and performs global optimization across the tree structure. The final model, a GP-GOMEA-based survival tree (GP-GOMEA-ST), optimizes both the tree structure and the splitting functions simultaneously. This creates an opportunity to investigate whether a globally optimized survival tree using GP-GOMEA can deliver improved splitting strategies, predictive accuracy, and robustness in nonlinear survival problems.

1.2 Research Question

This research aims to implement a survival tree that employs GP-GOMEA to optimize its decision nodes and evaluate its performance on problems where traditional survival trees typically encounter difficulties. Ultimately, this thesis seeks to address the following main research question:

Main Research Question: *Can a GP-GOMEA-based survival tree achieve a more effective splitting strategy than traditional greedy survival trees, thereby improving structural optimization and predictive performance?*

The research is divided into two stages. The first stage evaluates GP-GOMEA-ST on a synthetic problem to provide initial insight into its capabilities. The selected synthetic problem is the exclusive OR (XOR) problem, where traditional survival trees generally struggle due to their greedy splitting strategies. Comparisons will be made with a traditional greedy survival tree. Additionally, GP-GOMEA-ST is tested on a noisy XOR problem to assess its robustness to noise. This stage addresses the following research questions:

Research Question 1: *Can GP-GOMEA-ST discover optimal splitting strategies in synthetic problems where greedy survival trees typically fail, and how does its performance compare to these traditional methods?*

Research Question 2: *How does noise in the data affect GP-GOMEA-ST?*

The second stage focuses on optimizing GP-GOMEA-ST by examining the effects of population size and available training data. The objective is to identify the best overall hyperparameter settings. This stage investigates:

Research Question 3: *What is the impact of dataset size and population size on the performance of GP-GOMEA-ST?*

1.3 Outline of the Thesis

The remainder of this thesis is organized as follows. Chapter 2 presents the relevant background explaining the GP-GOMEA framework and survival trees. Chapter 3 details the methodology for constructing the proposed survival tree, GP-GOMEA-ST. Chapter 4 compares GP-GOMEA-ST to other survival tree approaches on a synthetic problem. Chapter 5 investigates optimal hyperparameters for GP-GOMEA-ST. Chapter 6 concludes the thesis, outlining potential limitations in the study and directions for future research.

Chapter 2

Background

This chapter will review the key concepts and background necessary of the thesis. This chapter is split into two sections: the first section is on survival analysis and different survival models, and the second section provides a background on the gene-pool optimized mixing evolutionary algorithm (GOMEA) used and how GOMEA can optimize a survival tree.

2.1 Survival Analysis

Survival analysis is a statistical method used to analyze and model time-to-event data[4, 21]. The primary objective is to estimate the survival function, $S(t) = P(T > t)$, which represents the probability that the event of interest has not occurred by time t. A key feature of survival analysis is the presence of censored data. Censored data introduces incomplete information [34]. This generally happens due to the study design of an experiment. Survival analysis is typically done in domains where there is a limited number of observations. Censored data prevents traditional models from correctly predicting, as they assume complete data. There are three main types of censoring:

- **Right censoring:** The most common form occurs when an individual's event time is unknown. For example, a patient still alive at the end of a clinical trial is right-censored. This thesis only contains data with right censoring.
- **Left censoring:** Happens when the event of interest has already happened, but the exact time is unknown. An example would be a patient who has already developed a condition before the study's start date, but whose onset time is not recorded.
- **Interval censoring:** Arises when the event occurs within an interval between two observation times, but the exact event time remains unknown. For example, a patient's disease progression is only known to have occurred between two clinical visits.

Survival analysis aims to estimate the survival distribution while taking censoring into account. Survival models are often employed to quantify the effects of predictor variables on the survival distribution. The Cox proportional hazards regression model and survival trees are two widely studied approaches. The upcoming sections discuss their use and benefits in survival analysis.

2.1.1 Cox proportional hazards regression

Cox proportional hazards regression, also known as Cox regression, is a widely applied model in survival analysis due to its semi-parametric nature and interpretability. It estimates the hazard, which represents the instantaneous rate at which an event occurs at time t, given that the individual has survived up to that time, based on a set of covariates [6]. Cox regression assumes that the hazard ratios between individuals are proportionate over time, referred to as the proportional hazard assumption. The following Equation 2.1 defines how the hazard is calculated for a given time t

$$h(t|\mathbf{X}) = h_0(t) \cdot e^{\beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_n X_n} \tag{2.1}$$

The hazard $h(t|\mathbf{X})$ at time t and coviarates $\mathbf{X} = (X_1, X_2, ..., X_n,)$ depends on the baseline hazard h_0 at time t and covariate coefficients $\beta_1, \beta_2, ..., \beta_n$. The baseline hazard is the value when all covariates are set to zero. The effect of each covariate can be quantified by its coefficient, rendering the Cox regression model interpretable.

Despite its popularity, the Cox proportional hazards model has significant limitations. It relies on the proportional hazards assumption and models covariate effects linearly on the log-hazard scale. This restricts its capacity to capture complex or nonlinear relationships between predictors. These assumptions make Cox regression less suitable for datasets with complex feature interactions. For this reason, Cox regression is not employed in this thesis. Instead, the focus is placed on survival trees, which offer a more flexible, nonparametric framework. Unlike Cox models, survival trees can accommodate nonlinear dependencies and interactions, making them more robust across diverse data types and distributions.

2.1.2 Survival Trees

Survival trees are a commonly used machine learning technique in survival analysis [13, 27]. They aim to predict the survival function based on censored data. Survival trees are binary decision trees specifically designed for survival analysis. The trees recursively partition the solution space through decision nodes. A decision node contains a function of $x < t$, where x is a feature and t is a threshold value. The leaf nodes of the survival tree represent groups of individuals with unique survival distributions. Figure 2.1 shows an example of a bare survival tree. The most common way to create a survival tree is by greedily splitting the solution space according to a metric. The metrics used are generally log-rank statistics [32] or log-likelihood ratios [46] that aim to divide the solution space into groups with different survival experiences. Survival trees are evaluated based on their ability to accurately predict an individual's survival probability given their predictor variables. Another strength of survival trees is that they are interpretable. This is due to their transparent and logical structure, which allows one to understand the decisions made.

Nonetheless, survival trees have three significant limitations. The first limitation is the instability of a survival tree. Small changes in the data can translate into a different survival tree, thus not generalizing well. The second limitation is that traditional survival trees employ a local greedy metric, which does not capture global trends. The last limitation is the iterative nature of univariate splits, which can capture nonlinear relations between predictor variables; however, this subsequently increases the size of the survival tree, thereby reducing its interpretability.

New methods for constructing survival trees have been developed to overcome these limitations. The following sections provide an in-depth discussion of three state-of-the-art models: the Optimal Sparse Survival Tree, the Oblique Survival Tree, and the Globally Induced Survival Tree, highlighting their innovations and limitations.

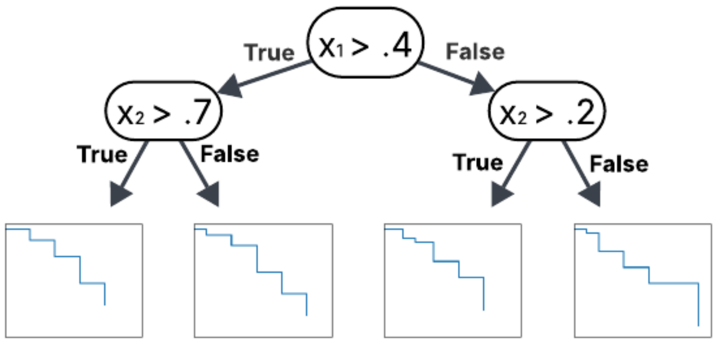


Figure 2.1: Example of a survival tree with two predictor variables where the graphs represent the survival function associated to each group

Optimal Sparse Survival Tree

The Optimal Sparse Survival Tree (OSST) [18] is a state-of-the-art survival tree that utilizes dynamic programming. More precisely, OSST employs a dynamic programming with bounds approach to identify sparse survival trees. This enables the model to exhibit improved predictive capabilities and capture more complex relationships. These survival trees are ”sparse” as OSST produces smaller survival trees with fewer splits. This is a desirable feature as it allows for more interpretable solutions. Another key feature of OSST is its non-greedy splitting strategy.

Instead of using a greedy splitting strategy, OSST uses dynamic-programming-with-bounds, an algorithm that aims to prune the search space until it finds an optimized survival tree. The algorithm begins with a single leaf node and recursively explores all possible feature splits, generating sub-problems. Each sub-problem maintains a lower and upper bound of its objective value. The search space is pruned if the bounds show that a sub-problem cannot improve the solution. The optimization is complete once all sub-problems are solved, at which point the optimal survival tree can be extracted. The resulting survival tree is optimized in a non-greedy way and is also sparse.

Despite these advantages, OSST also has limitations. The OSST is a single-tree method. This implies that it still has the same issues as traditional survival trees. Notably, they are sensitive to noise and data perturbations, leading to instability in some scenarios. Additionally, the OSST uses univariate splits just like traditional survival trees. Therefore, if the OSST needs to capture a nonlinear feature, the depth of the OSST will need to be large, lowering the tree's capacity and interpretability. Finally, while sparsity improves interpretability, it may come at the cost of predictive accuracy when modeling highly nonlinear or complex interactions. The following section introduces another state-of-the-art survival tree that uses an EA and outlines how it addresses some of the shortcomings of OSST.

Globally induced Survival Tree

The Globally Induced Survival Tree (GIST) [24] is a state-of-the-art survival tree that utilizes EAs to achieve global learning. GIST overcomes the instability and local suboptimality limitations of traditional and greedy tree-based methods. Unlike conventional approaches, GIST globally optimizes the entire survival tree, aiming for both stability and improved predictive performance.

GIST constructs trees using an evolutionary induction framework. GIST does not rely on greedy recursive splitting; instead, it treats the entire tree as an individual and evolves a population of trees using evolutionary operators, keeping the best-performing survival tree as the final solution. GIST utilizes a standard EA, the primary process of which is defined in Section 2.2. This design makes GIST less sensitive to noise and small perturbations in the dataset, addressing a significant weakness of traditional survival trees. Additionally, GIST employs a penalty term related to the size of the survival tree to favor smaller, more interpretable trees.

One of the key limitations is that the splits done by GIST are univariate. Hence, GIST requires deeper trees to capture highly nonlinear effects. This reduces the interpretability of the tree. Another limitation is the variation operation used in GIST. GIST does not consider how the genotype interacts with one another and applies variation randomly. Hence, variation operations often result in worse survival trees. The next section introduces the last state-of-the-art survival tree that expands on GIST to create globally optimized survival trees with multivariate splits using an EA.

Oblique survival Tree

The Oblique Survival Tree (OST)[23] is a state-of-the-art survival tree that utilizes EAs to induce global optimization. OST uses multivariate splits to overcome limitations of traditional survival trees, mainly their linear splits and susceptibility to being stuck in local optimas. By employing multivariate splits, OST can model complex, nonlinear relationships between predictors, improving both predictive accuracy and robustness to noise.

OST constructs trees using the same process as GIST. The primary difference is that each split in the tree is defined as a linear combination of multiple predictors, enabling oblique decision surfaces that can capture multidimensional relationships. Additionally, OST incorporates a regularization term to penalize overly complex trees, favoring simpler, more interpretable structures.

A key limitation of OST lies in its variation operators 2.2.1, operations that modify a tree. OST performs minimal crossover and relies mainly on mutation to introduce diversity. The reason behind that choice is that improper crossover can easily disrupt the tree structure and degrade model quality [12]. Similar to GIST, OST does not account for dependencies between genes, meaning that variation is applied without exploiting structural relationships in the data. Consequently, search efficiency is reduced, and beneficial building blocks may be lost during the mutation process. Furthermore, although OST supports multivariate splits, these splits are restricted to linear combinations of predictors. This constraint prevents OST from modelling nonlinear decision boundaries that require interactions between variables. Collectively, these limitations inhibit OST's ability to discover complex global structures, particularly in highly nonlinear problems.

These limitations have led to the development of the GP-GOMEA-based survival tree. The following section will first aim to explain the overall process of an EA, explain how GP-GOMEA differs from standard EAs, such as the one used in GIST [24], and why it is preferable to use GP-GOMEA over other EAs.

2.2 Evolutionary Algorithms

This section will discuss the fundamental principles of EAs. EAs are algorithms that draw their name from Darwin's widely known theory of evolution [7]. The objective of EAs are to adapt a population to its environment. Like in nature, EAs have a *population* comprised of *individuals*; in this case, an individual corresponds to a solution in the EA. EAs optimize their population through the iterative process of selection and variation. The EA used in this project is GP-GOMEA which can be view in detail in Section 2.2.5

2.2.1 Principle components of EAs

Typically, the population of an EA is initialized randomly, and throughout the generations, the solutions are optimized until a termination criterion is met, like a time limit or when all individuals have converged to the same solution. A generation consists of two steps: selection and variation.

Selection is one of the key components of an EA. Its biological counterpart is often referred to as ”survival of the fittest”. This process ensures that the population is optimized and converges. To do so, every individual has a fitness, a numerical way of determining the goodness of a solution, which is obtained through a fitness function. The selection process involves identifying the most dominant individuals in a generation with above-average fitness. By choosing these high-performing individuals, their ”good” genes will likely be passed down to the next generation. Therefore, future generations keep these favorable traits. This combination of favorable traits is referred to as *building blocks* and is the cornerstone of a good solution. The most common way to do this is with *tournament* selection, as opposed to other methods such as proportionate selection and rank/domination-based selection. Tournament selection is straightforward: a few individuals are selected, and only the best individual, also called the *tournament winner*, remains. This process is repeated multiple times to get enough individuals for the next generation. Tournament selection can be performed with or without replacement. An advantage of tournament selection is that it exerts high selection pressure, meaning there is a fast convergence of the population.

Variation is the other component of a traditional EA. In nature, no two individuals have the same DNA. This is due to variation, which ensures diversity within the population. This factor is esdsential in EAs as it allows them to explore more of the solution space. In EAs, two operators manage variation: *crossover* and *mutation*. Crossover is the process in which *parent* individuals recombine their parameters to create new *offspring* individuals. This operation aims to produce fitter offspring that keep the favorable building blocks from the parents. A limitation of the crossover step is that it constrains the search space to the initial population. If the optimal building blocks are not present in the initial set of individuals or are lost in the selection process, it would be impossible to find the optimal solution; hence, the need for another variation operator: mutation. In EAs, the mutation phase mimics the random mutation that can occur within an individual in nature. In the context of EAs, it is done by slightly altering an individual, such as changing a bit from a 0 to a 1. Mutation allows for the introduction of new building blocks previously not in the population. It slows down the convergence process and maintains diversity among the population.

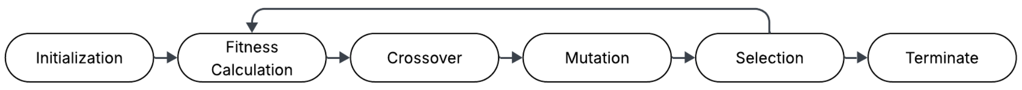


Figure 2.2: The optimization process of an EA

While all EAs have a selection and variation step, how these steps are performed differs from one EA to another. A state-of-the-art EA, GOMEA, uses a special type of variation to obtain better individuals.

2.2.2 Gene-pool Optimized Mixing Evolutionary Algorithm

GOMEA is a state-of-the-art evolutionary algorithm [41]. One of the main characteristics is that it utilizes Gene-pool Optimal Mixing (GOM) for the variation process. GOM allows for intermediate function evaluation, which enables considerable speedups over traditional EAs, as it tests various sets of modifications and accepts them if they improve the solution. This is due to the fact that GOM can exploit the linkage between variables by performing crossover over groups of variables with high linkage.

The sets utilized during GOM are contained in a structure called *family of subsets* (FOS). The FOS, denoted as $\mathcal{F}$, is a set of subsets. Assuming l different variables, a FOS is mathematically defined in Equation 2.2:

$$\mathcal{F} = \{\mathbf{F}^0, \mathbf{F}^1, ..., \mathbf{F}^{|\mathcal{F}|-1}\} \text{ where } \mathbf{F}^j \subseteq \{0, 1, ..., l-1\} \wedge j \in \{0, 1, ..., |\mathcal{F}|-1\} \tag{2.2}$$

GOM uses a FOS called the linkage tree (LT-FOS). The LT-FOS is a hierarchical structure that contains all the variables of the optimization problem. The LT-FOS is built from the bottom up, starting with univariate FOS, an FOS where every variable is modeled as independent of every other variable. Then, the two subsets that exhibit the highest similarity according to a chosen metric are merged. This process is repeated until all the variables are combined into a single joint set. The LT-FOS follows the following rule: any subset in the LT-FOS with more than one variable is composed of the union of two smaller subtests in that LT-FOS. Equation 2.3 mathematically describes that rule:

$$\forall \mathbf{F^j} \in \mathcal{F} \left(|\mathbf{F^j}| > 1 \rightarrow \left(\exists \mathbf{F^x} \, \exists \mathbf{F^y} \left((\mathbf{F^x} \cap \mathbf{F^y} = \emptyset) \wedge (|\mathbf{F^x}| < |\mathbf{F^j}|) \wedge (|\mathbf{F^x}| < |\mathbf{F^j}|) \wedge (\mathbf{F^x} \cup \mathbf{F^y} = \mathbf{F^j})\right)\right)\right) \tag{2.3}$$

Mutual information measures similarity and is used to decide which subsets in an LT-FOS should be merged together. By iteratively joining the subsets with the highest similarity, the LT-FOS is obtained. Before every generation, GOMEA will continue to refine the LT-FOS. By capturing variable dependencies through the LT-FOS, GOMEA can perform more effective recombination than traditional EAs, which assume variable independence.

GOM uses subsets of the LT-FOS to transform a parent solution into a new offspring for the upcoming generation. The process goes as follows: a parent solution is first copied into a temporary offspring. Then, for each subset in the LT-FOS, processed in random order, the corresponding genes are injected from a randomly selected donor parent within the population. The donor variables are retained if they do not worsen the fitness of the solution; otherwise, the variables from the parents are restored. This process is repeated for every individual in the population, ensuring each parent creates one offspring. By focusing on mixing linked gene subsets, GOM effectively exploits problem structure, balancing exploration and exploitation.

While GOMEA provides a powerful mechanism for capturing variable dependencies and improving variation efficiency, it traditionally focuses on optimizing a single objective. However, in many real-world problems, multiple metrics need to be optimized. Hence, there is a need for multi-objective (MO) optimization.

2.2.3 Multi-Objective Optimization

MO optimization aims to optimize multiple, often conflicting objectives simultaneously. In many scenarios, the objectives can not be reduced to a single scalar objective without the introduction of bias, as the improvement of one objective could lead to the deterioration of another. Instead of having a single optimal solution, MO optimization provides a set of solutions with varying trade-offs. The set of solutions, also called the approximation front, represents different compromises between the objectives.

The approximation set is constructed using the principle of *Pareto dominance*. Assuming a problem where all objectives need to be minimized, a solution x^0 is said to Pareto dominate a solution x^1 if and only if for all objectives i and for at least one objective, solution x^0 is strictly better than x^1. This is formally defined in Equation 2.4. This is also denoted as $\mathbf{x}^0 \succ \mathbf{x}^1$. The mathematical expression can be seen in Equation 2.5. An even stronger condition is *Pareto optimal* solutions. A solution $\mathbf{x}^0$ is Pareto optimal if and only if no solution dominates it. Formally noted in Equation 2.6. The set $\mathscr{P}$, known as the *Pareto* *set*, is the set of all Pareto optimal solutions formally defined in Equation 2.7. The Pareto set gives the Pareto front, the set of all objective function values corresponding to the solutions in the Pareto set.

$$\min f(x) = (f_0(x), f_1(x), \ldots, f_{m-1}(x)) \tag{2.4}$$

$$(\forall i \in \{0, 1, \ldots, m-1\} : f_i(x^0) \leq f_i(x^1)) \wedge (\exists i \in \{0, 1, \ldots, m-1\} : f_i(x^0) < f_i(x^1)) \tag{2.5}$$

$$\neg \exists x^1 : x^1 \succ x^0 \tag{2.6}$$

$$\mathscr{P} = \{x^0 | \neg \exists x^1 : x^1 \succ x^0\} \tag{2.7}$$

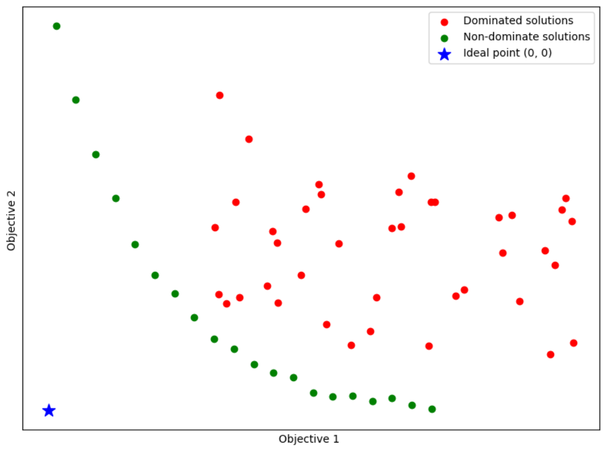


Figure 2.3: Visualization of an approximation front with two objectives

An issue with MO optimization is front degradation: the Pareto front can worsen due to discarding a solution in the Pareto set. To address this, elitist archives are utilized [28]. Elitist archives store non-dominated solutions. A solution is stored in the archive if one of the following conditions is met: the solution dominates at least one solution in the archive, or no solution dominates it. Any dominated solution is removed from the archive. Elitist archive allows for diverse and stable solution front

A typical use case of multi-objective optimization is symbolic regression, where both the regression's accuracy and the size of the expression are considered. Balancing these objectives is essential to ensure that the resulting models are accurate and interpretable.

2.2.4 Genetic Programming and Symbolic Regression

Genetic programming (GP) is a specialized subset of EAs. In the context of GP, individuals are symbolic expressions, typically represented as trees. Symbolic Regression (SR) is a type

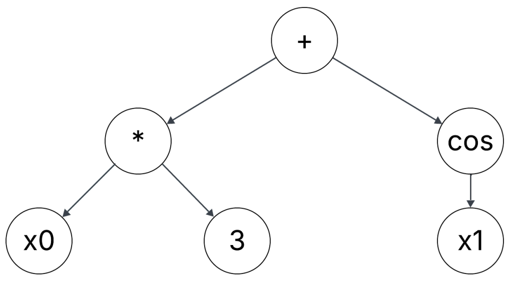


Figure 2.4: Visualization of a symbolic regression tree

of regression analysis that aims to uncover the mathematical expression that best describes a data set. The aim of GP is to, formally, given a data set of d features and n samples and a target variable y, find a relationship f such that $f : R^d \rightarrow R$. One advantage of SR is that the resulting models are inherently interpretable. In the context of this thesis, the goal is to develop a glass-box model, meaning that the underlying decision process is fully transparent and the explanations provided are precise and comprehensible to humans. SR directly supports this objective by producing explicit mathematical expressions that can be analyzed without requiring external interpretation methods. Figure 2.4 shows a symbolic regression tree for the expression: $(x_0 * 3) + cos(x_1)$

This thesis applies a tree-based GP framework to solve a symbolic regression problem. Each tree represents a candidate program. Internal nodes denote functions and leaves denote terminals. Three sets define the components used to construct these trees:

- **Function set**: Contains the atomic functions, arithmetic or mathematical operations that GP can combine to form programs.
- **Terminal set**: Contains the variables and constants that serve as the inputs or fixed values in the programs.
- **Primitive set**: The union of the function and terminal sets.

The GP search process operates over a vast, combinatorial space of possible symbolic expressions. Therefore, it is key to have variation operators that are capable of navigating the entire space. Just like in a regular EA, the main variation operators are crossover, which exchanges subtrees between individuals, and mutation, which alters the expression of a node. Selection in a GP drives the population toward fitter individuals. This balance between exploration and exploitation enables GP to discover nonlinear and complex relationships in data without the need to predefine the model structure. Compared to traditional regression techniques such as linear or Cox regression, SR is more flexible, as it can automatically uncover nonlinear interactions and produce closed-form mathematical expressions that remain interpretable.

Despite these strengths, GP-SR also faces several limitations. A well-known issue is *bloat* [36], where trees tend to grow excessively large without any improvements in performance. Thus, bloat leads to reduced interpretability. Another limitation lies in the treatment of constants [37]. GP often struggles to find the precise value a constant should take. This can lead to suboptimal approximations of the true underlying function. Furthermore, variation operators such as crossover and mutation are applied in a largely random manner and may separate correlated variables, disrupting useful structural relationships and degrading solution quality[5].

These challenges motivate the exploration of more advanced GP variants. In particular, this thesis employs GP-GOMEA, a recent approach that enhances the variation process by modeling and exploiting dependencies between program components.

2.2.5 Genetic Programming Gene-pool Optimized Mixing Evolutionary Algorithm (GP-GOMEA)

GP-GOMEA is a genetic programming framework that incorporates GOMEA. The first limitation in traditional GP is that variation operators may separate correlated genes. Unlike standard GP, which applies variation randomly and often destructively, GP-GOMEA uses linkage learning to perform more informed search operations. This prevents the destruction of individuals, as only changes that yield an increase in fitness are considered. The version of GP-GOMEA used in this research stems from Virgolin et al.'s project [44].

This version of GP-GOMEA is multi-tree [39], meaning each individual is a collection of GP trees. This collection of trees is optimized simultaneously by GP-GOMEA, allowing for the exploration of a wider solution space while maintaining diversity of the GP trees.

Furthermore, GP-GOMEA is capable of handling both categorical and numerical data. This versatility broadens its applicability to real-world survival analysis problems, where datasets often contain various variable types. This reduces the amount of pre-processing or data transformation needed.

An issue in GP is the optimization of constants often represented as Ephemeral Random Constant (ERC) [31]. GP-GOMEA tackles this issue by using on-line binning of constants to optimize the ERCs efficiently. On-line binning limits the constants considered during linkage learning to a maximum of γ. Once γ unique constants have been encountered, any new constant is assigned to the bin of the closest existing constant.

A third major limitation in GP is bloat. GP-GOMEA deals with this issue in three ways. First, GP-GOMEA limits the number of nodes in an individual and the depth of each node. This creates a strict upper bound that prevents individual solutions from expanding indefinitely. Secondly, this version of GP-GOMEA utilizes MO optimization, where two objectives are considered: size and predictive performance. By explicitly optimizing for size, the algorithm produces smaller, more interpretable individuals, while maintaining predictive performance. This is important for applications where interpretability is necessary, such as survival analysis. In addition, large individuals incur a higher size-objective value and are therefore implicitly penalized, functioning as a pruning mechanism that discourages excessively complex solutions. Furthermore, MO optimization is employed because it is undesirable to have only one model. It is more desirable to have a set of different compromises between complexity and performance that depict varying trade-offs. Lastly, GP-GOMEA is capable of using high-arity operators [38]. High-level arity operators allow the creation of complex yet shallow and interpretable trees; this, in turn, prevents bloat.

Chapter 3

GP-GOMEA Survival Tree

This section will discuss the methodology of the proposed GP-GOMEA survival tree (GP-GOMEA-ST). GP-GOMEA-ST is implemented on top of Virgolin et al.'s project [44]. That specific version of GP-GOMEA is described in Section 2.2.5. As GP-GOMEA employs MO optimization, it utilizes multiple evaluation criteria. The first evaluation criterion is the size metric 3.1.1. This metric is identical to Virgolin et al's initial project. The other criterion is fitness metrics taking into account the censored data. Specifically, the Concordance Index (C-Index) and Integrated Brier ScoreIBS (IBS) 3.1.3 were chosen to assess the predictive accuracy of the GP-GOMEA survival trees. As no current C++ library has implemented these metrics, they were implemented manually. To ensure the proper functioning of these metrics, they were compared to the Python *scikit-survival* library to verify that they yield identical results on the same input.

The second part of this chapter discusses how a GP-GOMEA-based survival tree is represented and the changes made to Virgolin et al.'s project to achieve that. The first step is to impose a tree structure by imposing a hierarchical order to the nodes in the **Multitree** class. The second step is to change how data is processed over a given GP-GOMEA-ST to reflect to new tree structure.

3.1 Evaluation Criteria

This section will cover the various metrics used for MO optimization. The first metric size is a proxy for interpretability, favoring more interpretable solutions over complex ones. The second metric is a survival metric that favors solutions with high predictive accuracy. Two commonly used methods were considered for the survival metric: the C-Index and the IBS. This section will also discuss why the IBS is preferred over the C-Index.

3.1.1 Size

The first metric considered is the size of a GP-GOMEA-ST individual. The size metric S_T for an individual survival tree T is defined in Equation 3.1.

$$S_T = \sum_{t \in T} |t| \tag{3.1}$$

In Equation 3.1, $|t|$ denotes the number of components in the expression at each decision node t. Thus, S_T represents the cumulative size of all expressions across the survival tree T.

The size of a survival tree is defined by two hyperparameters: the number of decision nodes n and the maximum depth h of the tree. The size is bounded within the interval $[n, n \cdot (2^{h+1} - 1)]$. While the number of decision nodes n scales linearly, increases in h lead to exponential growth in the potential tree size. Consequently, restricting the upper bound of h is important to prevent exponential growth and retain interpretability. GP-GOMEA aims to minimize the size objective to favor smaller, more interpretable trees.

The depth of the survival trees directly impacts model complexity and the risk of overfitting [17], supporting the need to limit depth for interpretability. Some state-of-the-art survival trees employ depth-limited dynamic programming approaches to strike a balance between performance and interpretability [18]. While the size of a solution is essential to maintain the interpretability of a GP-GOMEA-ST, it does not determine the predictive performance of a GP-GOMEA-ST. Therefore, an additional evaluation metric is needed, most notably the C-Index and the IBS.

3.1.2 The predictive performance metrics

The Concordance Index (C-Index) and the Integrated Brier Score (IBS) are the two metrics considered to evaluate the predictive performance of a GP-GOMEA-ST. These metrics are the primary indicators of the predictive accuracy of a survival tree. The C-index [15] measures the model's ability to correctly rank the survival times based on predicted risk scores. To predict the risk score, the C-index utilizes the Nelson-Aalen Estimator (NAE), which estimates the cumulative hazard function in a nonparametric way [30, 1]. Recently, numerous shortcomings of the C-index were found [16, 2]. While both metrics were initially implemented, this study ultimately prioritizes the IBS, providing a more comprehensive assessment of survival prediction performance, particularly under censoring.

3.1.3 Integrated Brier Score

The Integrated Brier Score (IBS) is a metric mainly used in survival analysis due to its ability to deal with censored data. The IBS evaluates a survival model's overall accuracy over a specified period. The time frame in this thesis is dependent on the data set and is set between $[t_1, t_{max}]$ where t_1 is the lowest and t_{max} is the highest time an individual survived in the dataset. The IBS is bound between $[0, 1]$. Lower IBS values indicate better predictive performance, suggesting that the survival probabilities predicted by the model are close to the actual outcome on average over time. To get the overall performance, the IBS evaluates the model's performance over the entire follow-up period rather than at a single time. In this thesis, the models evaluated are the survival trees. The IBS is estimated via the trapezoidal rule. The trapezoidal rule numerically integrates the observed Brier Score (BS) over time by approximating the area under the score curve as a series of trapezoids. The BS will be discussed more deeply in section 3.1.4. The exact formula used to calculate the IBS is shown in Equation 3.2:

$$\mathrm{IBS} = \frac{1}{t_{\max} - t_1} \int_{t_1}^{t_{\max}} BS(t)\, dt \tag{3.2}$$

3.1.4 Decomposition of the IBS

Three subcomponents must first be calculated to calculate the IBS: the BS, the survival function, and the inverse probability of censoring weights. This section will explain these components and their use in computing the IBS.

The survival function & the Kaplan-Meier estimator

The survival function is a function that gives the probability that an individual will survive beyond a specific time. The survival function is the foundational quantity that a model must estimate. Often, it is neither feasible nor desirable to specify a parametric model for the survival function; therefore, a nonparametric approach is used. A commonly used nonparametric estimator to estimate the survival function $\hat{S}$ is the Kaplan–Meier estimator (KME). The following Equation 3.3 describes how the KME is calculated:

$$\mathrm{KME}(t) = \hat{S}(t) = \prod_{t_i < t} \frac{r_i - d_i}{r_i} \tag{3.3}$$

In Equation 3.3 r_i is the number of individuals at risk at time t_i. It is defined as the sum of all individuals whose event time is greater than t_i. d_i is the number of uncensored individuals whose event time equals time t_i. The KME is a strictly decreasing stepwise function. It is one of the key components in calculating the Brier Score.

Brier Score

The BS is a metric that measures the accuracy of a probabilistic prediction. The original BS is quite similar to the mean square error. The original BS is defined as follows:

$$\mathrm{BS} = \frac{1}{N} \sum_{x=1}^{N} \sum_{y=1}^{R} (f_{xy} - o_{xy}) \tag{3.4}$$

In the Equation 3.4 N is the total number of different instances, R is the number of different possible classes, f is the probability given to an individual, and o is the outcome of that said event. A shortcoming of this version of the Brier score is that it does not consider possible data censoring. While the initial BS cannot handle censored data, the integration of the inverse probability of censoring weights 3.1.4 (IPCW) into it makes it consider the censoring of data.

$$\mathrm{BS}(t) = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{(0 - \hat{S}(t, x_i))^2 \cdot I(y_i \le t \wedge \delta_i = 1)}{\hat{G}(y_i)} + \frac{(1 - \hat{S}(t, x_i))^2 \cdot I(y_i > t)}{\hat{G}(t)} \right) \tag{3.5}$$

The lower the $BS(t)$, the more accurate the model is for that specific time. The Brier Score used in this thesis is defined by the following formula 3.5. The key components in the BS are the survival function $\hat{S}$, the probability of censoring weight $\hat{G}$, and two terms. The first term is the case term $I(y_i \leq t \wedge \delta_i = 1)$, the case term is a boolean value zero if event i is censored or has a time greater than the time stamp t. The other is the control term $I(y_i > t)$. Just like the case term, it is a boolean value. It takes a value of 1 if the event time of individual i is strictly greater than the timestep t.

Inverse probability of censoring weight

The IPCW is one of the terms needed to calculate the Brier Score. The IPCW corrects the BS for right-censoring. Just like the survival function, the IPCW uses the KME. However, in this case, the KME is applied to the censored times instead of event times. This function helps adjust for censored observations, ensuring that individuals who were censored earlier do not bias the risk estimation. It is calculated as follows:

$$\hat{G}(t) = \prod_{T_i < t} (1 - \frac{c_i}{r_i}) \tag{3.6}$$

The IPCW, $G(t)$, represents the probability of not being censored beyond time t. The first term c_i is the number of censored individuals at time t. The second term is risk r_i.

3.2 Representation

The GP-GOMEA-ST follows the classical evolutionary algorithm framework, where a population is optimized across generations through the processes of selection and variation. It is a survival tree, where every decision node is a tree-based GP optimized using GP-GOMEA.

First, the output of every decision node is constrained to be a Boolean. A GP-GOMEA-ST is like a binary decision tree; hence, a continuous output is unnecessary, as each decision node has two children. The root node of every decision node is enforced to be a comparison operator. This constraint applies to trees with a height of one or more. Doing so assures an inequality or equality, and thus a Boolean output. In the case where the tree of a decision node is of height zero, the root node is a Boolean value (True or False).

Like in the reference project, decision nodes are stored in a list called *nodes*. The decision nodes do not need to be stored differently, as imposing a decision tree structure on a list is possible. The tree hierarchy is imposed on the *nodes* list through index-based organization, following these three structural rules:

- The root node of the survival tree is located at index 0 in *nodes*.
- The left child of a node at position i is located at index $2i+1$
- The right child of a node at position i is located at index $2i+2$

A limitation of this indexing approach is that survival trees generated this way are always left-filled. Nodes are assigned in order, which leads to less flexible tree structures. However, this representation allows for efficient top-down traversal of the tree, which is useful when processing the data.

Additionally, in this project, the height of GP-GOMEA-ST is constrained. GP-GOMEA-ST has a maximum height of three; therefore, a full GP-GOMEA-ST will have at most seven decision nodes and eight child nodes. This additional constraint is due to the fact that every extra decision node adds exponential time to optimizing the GP-GOMEA-ST. Additionally, increasing the number of decision nodes penalizes the interpretability of a solution.

Algorithm 1: Recursive Tree Splitting (RTS) Procedure

```
Input: $X$ : data matrix, $p$ : current node position
Output: List of data subsets (terminal nodes)
$\mathbf{o} \leftarrow \emptyset$
if $p \geq \texttt{max\_nodes}$ or $X = \emptyset$ then
    return o
$\mathbf{b} \leftarrow$ Boolean output of decision node $p$ on $X$;
$I_{\text{true}} \leftarrow$ indices where $\mathbf{b} = 1$;
$I_{\text{false}} \leftarrow$ indices where $\mathbf{b} = 0$;
$X_{\text{true}} \leftarrow X[I_{\text{true}}]$;
$X_{\text{false}} \leftarrow X[I_{\text{false}}]$;
$p_{\text{left}} \leftarrow 2p + 1$;
$p_{\text{right}} \leftarrow 2p + 2$;
if $X_{true} \neq \emptyset$ then
    $\mathbf{o} \leftarrow \mathbf{o} +$ RTS($X_{\text{true}}, p_{\text{left}}$);
if $X_{false} \neq \emptyset$ then
    $\mathbf{o} \leftarrow \mathbf{o} +$ RTS($X_{\text{false}}, p_{\text{right}}$);
return o;
```

Another change made is to adapt the way data is processed to suit a GP-GOMEA-ST. In Virgolin et al.'s original project, data is traversed sequentially across all nodes, which is not directly applicable to a tree-based survival model. In the context of survival trees, this sequential approach fails to respect the hierarchical ordering of the decision nodes. Therefore, a new procedure tailored to the survival tree structure is required. The process can be seen in Algorithm 1. It aims to partition the initial dataset into the appropriate child nodes of the decision tree by recursively splitting the data. The procedure begins at the root decision node of the tree and continues until the terminal nodes are reached. Separating the dataset enables the computation of the survival function3.1.4 in each child node. The survival function is one of the key components to computing an individual's fitness.

Figure 3.1 shows the main steps from initialization to completion of the GP-GOMEA-ST algorithm. Throughout this process, GP-GOMEA is responsible for initializing the population. Furthermore, GP-GOMEA replaces the worst-performing individuals and applies variation to these individuals. Once created, these survival trees need a metric to evaluate them. As the algorithm has multiple objectives, the output is a set of survival trees, all of which are part of the approximation front.

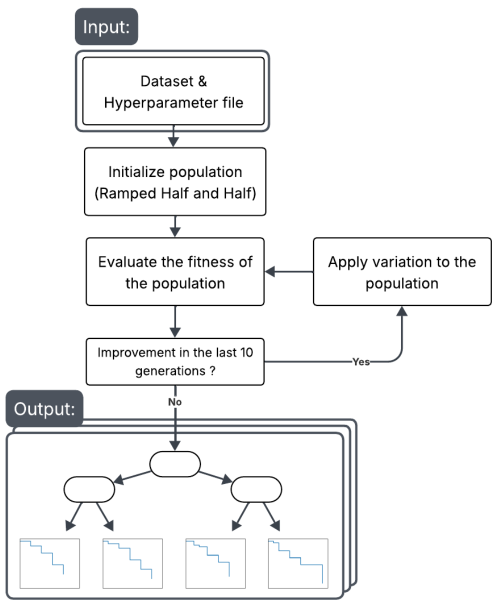

Input:
Dataset & Hyperparameter file
Initialize population (Ramped Half and Half)
Evaluate the fitness of the population
Apply variation to the population
Improvement in the last 10 generations ?
Yes
No
Output:

Figure 3.1: Schema of the process of the genetic programming gene-pool optimized mixing evolutionary algorithm survival tree

Chapter 4

Experiment on synthetic data

The objective of the 1st experiment is to determine the performance of GP-GOMEA-ST on synthetic data and how robust GP-GOMEA-ST is to non-essential information. The performance of a survival tree is given by the IBS defined in Section 3.1.3, where a lower score indicates a better performance. GP-GOMEA-ST will be compared to two different survival trees. One of the survival trees is the `SurvivalTree` model from the `sksurv.tree` module of the `scikit-survival` library [33], it will be referred to as the *Sksurv survival tree* (SST). The other survival tree is the *"Ideal" Survival Tree* (IST) that is the best possible survival tree given a dataset. The use of synthetic data allows for making an IST as the ideal splits are known. These two different survival trees allow for benchmarking of the implemented algorithm. Using the IST enables us to see if GP-GOMEA-ST finds the correct boundaries. Section 4.1 does a deeper dive into the datasets.

4.1 Test Problems

Two synthetic datasets were employed in this experiment. Both datasets contain 10,000 data points with two feature vectors, x_1 and x_2, drawn from a uniform distribution in $[-1,1]$. Additionally, these datasets are split $75-25$ training and testing. The first dataset only has these two features. On the other hand, the second dataset has these two features and three noisy features. Each of noisy features is generated from a different distribution: the first noisy distribution is a uniform random integer from $[0,5]$, the second noisy feature is random values from an exponential distribution with a scale of 1, and the last noisy feature is a normal distribution with a center of 1 and a standard deviation of 1.

The motivation for using synthetic datasets is twofold. First, they provide complete control over the sample size and feature distributions. Second, synthetic datasets allow for the construction of datasets with known optimal solutions. For both datasets in this experiment, the optimal survival tree, referred to as the ideal survival tree (IST), can be explicitly defined. The IST allows a direct performance comparison between the GP-GOMEA-ST and the optimal model.

Both datasets are designed to mimic the XOR problem with continuous features, a known challenge for greedy survival trees. Such splitting strategies often fail to capture the non-linear relationships between features, particularly when only univariate splits are allowed. In both of these problems, a strong performance requires discovering two hidden intermediate features, defined in the Equation 4.1

$$x_3 = x_1 \times x_2, \quad x_4 = x_1^2 + x_2^2. \tag{4.1}$$

The ideal survival tree for both XOR problems is $x_3 \leq 0.0$ XOR $x_4 \leq 0.664$. Figure 4.1 illustrates the decision boundaries for the XOR problem with x_4 plotted in red and x_3 in blue. These datasets allow us to test whether GP-GOMEA-ST can discover an optimal splitting strategy, requiring a global and multivariate splitting strategy where greedy methods struggle to.

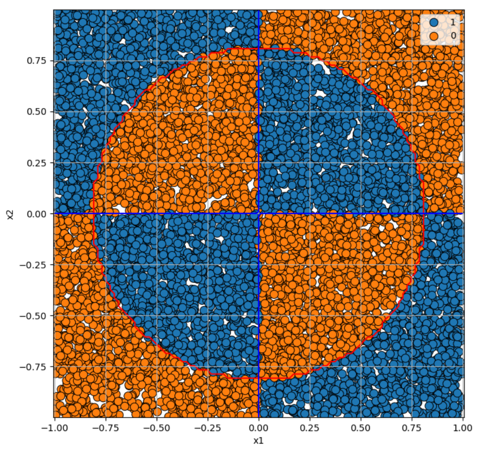


Figure 4.1: Plot showing the distribution of the data as well as the boundaries

4.2 Experimental Setup

Three different types of survival trees were evaluated to run this experiment. First, the ideal survival tree (IST) is described in Section 4.1. The second survival tree is the SST. The maximum height of the SST was determined through hyperparameter tuning and set to 20, as this yielded the best results. This implementation employs a greedy splitting strategy, with each split being univariate. The third model is the GP-GOMEA-based survival tree 3, configured with a population size of 1024. The algorithm was allowed to run for up to 50 generations, or terminate early if no improvement in the hypervolume was observed for ten consecutive generations. Each GP-GOMEA survival tree was limited to a maximum of three decision nodes to ensure interpretability, with the expression inside each decision node restricted to a maximum height of four.

All three survival tree models were executed on 20 different random seeds for both the noisy and non-noisy XOR datasets. The same seeds were used across all models to ensure consistency in the introduced randomness. From these runs, the mean and standard deviation of the IBS were computed for both the training and testing sets. For the SST, the IBS was calculated using the `sksurv.metrics` module. To calculate the IBS, the SST needs to first predict the survival functions for the test dataset. The predicted survival functions, the corresponding event indicators, and time-to-event values are then used to calculate the IBS for the given PSA. Once all the IBS values are collected from the three different models, a statistical comparison will be done. Two statistical tests are used: the Paired t-test [40] and the Wilcoxon test [45]. This is done to ensure that the variation between two models is significant.

Additionally, for all 40 experimental runs involving the GP-GOMEA-ST, the approximation fronts were recorded. These fronts contain sets of solutions characterized by training and testing IBS values, the size, and expressions of the survival trees. Furthermore, additional parameters were recorded for each seed, including the number of generations executed, computation time, number of evaluations performed, and hypervolume. Analyzing these fronts enables a detailed examination of the decision boundaries produced by the GP-GOMEA-ST.

4.3 Results

The IBS performance of the SST and the GP-GOMEA-ST across 20 independent seeds on both non-noisy and noisy XOR datasets can be seen in Table 4.1. All the IBS values are multiplied by 100 for clarity. In the non-noisy case, GP-GOMEA-ST achieves a mean test IBS of 6.37, compared to 9.29 for the SST. The difference is of 3.49 corresponding to a relative improvement of approximately 37% on the test set. In the noisy case, the improvement was slightly larger, with a mean reduction of 3.64 equivalent to a 34% improvement. The Δ metric represents the deviation from the ideal IBS. It is close to zero for GP-GOMEA-ST in both settings, particularly in the non-noisy case (0.03 on the test set). For the SST, Δ is larger in both datasets, with values exceeding 2.9. Looking at the Test IBS, there is a deterioration of 0.35, approximately a deterioration of 5.5% when noise is added to GP-GOMEA-ST. On the other hand, the SST has a decline of 1.42. On the non-noisy training set, GP-GOMEA-ST achieved a lower standard deviation (0.13) than the SST (0.26), whereas on the test set, GP-GOMEA-ST recorded a higher standard deviation (0.07) than SST (0.01).

Survival Tree	**Dataset**	**Train IBS**	**Test IBS**	**Train Δ**	**Test Δ**
Sksurv survival tree	Non-noisy	9.45 ± 0.26	9.29 ± 0.01	3.48 ± 0.26	2.94 ± 0.01
Sksurv survival tree	Noisy	9.97 ± 0.40	10.71 ± 0.03	4.02 ± 0.40	4.36 ± 0.03
GP-GOMEA survival tree	Non-noisy	5.96 ± 0.09	6.37 ± 0.07	$\mathbf{0.04} \pm 0.09$	$\mathbf{0.03} \pm 0.07$
GP-GOMEA survival tree	Noisy	6.33 ± 0.25	6.72 ± 0.23	0.38 ± 0.25	0.38 ± 0.23
Ideal survival tree	Non-noisy	5.92 ± 0.1	6.34 ± 0.0	Na	Na
Ideal survival tree	Noisy	5.92 ± 0.1	6.34 ± 0.0	Na	Na

Table 4.1: Integrated Brier Score of two types of survival tree: the SST, and GP-GOMEA-ST. The IBS ($\times 100$) is expressed as the mean and the standard deviation, calculated over 20 seeds. The Δ is the difference over the 20 seeds from the ideal IBS

Figure 4.2 depicts the IBS on the test set across 20 different seeds for two models, GP-GOMEA-ST and SST, across both the XOR and Noisy XOR datasets. The figure illustrates that the IBS values for GP-GOMEA-ST on both XOR problems are more tightly clustered than those of the SST. Additionally, the visualizations show that only GP-GOMEA-ST could find optimal solutions. On the XOR problem seeds: 2, 3, 5, 7, 10, 13, 14, 17, 18, and 19 have an optimal decision tree. On the noisy XOR problem seeds, 13 and 17 found the optimal solution. This figure enables the identification of outliers within a model. There are no outliers among both SST. On the other hand, in the GP-GOMEA-ST non-noisy condition (indicated by blue squares), two outliers corresponding to seeds 4 and 12 are observed. On the noisy XOR problem, GP-GOMEA-ST (as red circles) has six outliers: seeds 7, 9, 11, 12, 16, and 18.

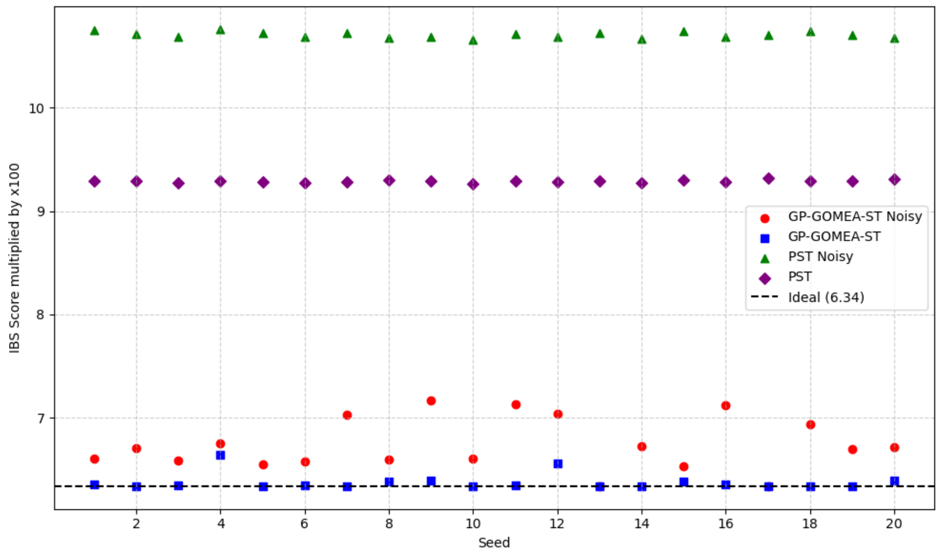
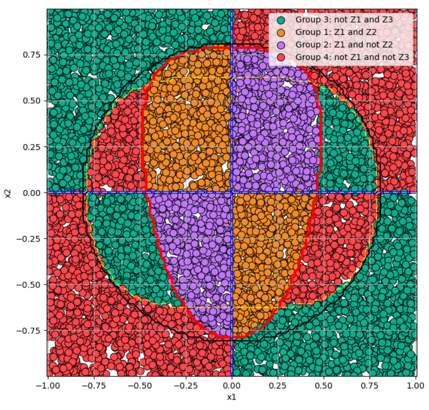


Figure 4.2: Scatter plot of IBS ($\times$100) for GP-GOMEA-ST and SST across 20 seeds on XOR and Noisy XOR datasets

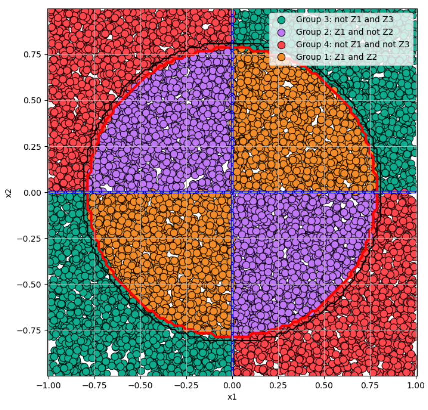
Building upon identifying outliers in Figure 4.2, the following section examines the decision boundaries associated with these outlier seeds to gain insight into their behavior. On every plot is a black outline that indicates the ideal circle size. Figure 4.3 shows the boundaries of a GP-GOMEA-ST with the lowest train IBS in the solution front for seeds 4 and 12. The red function depicts the boundary created with the function Z_1, Z_2 is the blue contour, and Z_3 is the yellow contour. The best GP-GOMEA-ST in seed 4 4.3a has an IBS of 6.75 and the solution size is 49. When simplified, the three decision nodes translate to the following expressions:

$$Z_1 = x_0^2(3 - 2x_1 - x_1^2) + x_1^2 \leq 0.635209 \quad Z_2 = \frac{x_0}{x_1} \leq x_0^2 \quad Z_3 = x_0^4 + x_1^2 \geq 0.391$$

For seed 12, the GP-GOMEA-ST with the lowest IBS has a value of 6.56 and a size of 28. Its expressions are as follows: $Z_1 = (x_0^2 + x_1^2) \leq 0.616$, $Z_2 = \frac{x_0^3}{x_1} \geq 0.0$, and $Z_3 = x_0 \leq \frac{x_0}{x_1}$.



(a) Boundaries of the best performing GP-GOMEA-ST in the approximation front on the non-noisy data on seed 4



(b) Boundaries of the best performing GP-GOMEA-ST in the approximation front on the non-noisy data on seed 12

Figure 4.3: Outliers in the non-noisy XOR problem

Among the boundaries of the best GP-GOMEA-ST per seed in the XOR problem, seed 3 stands out. Figure 4.4 displays the bounds of that special individual. It has a perfect IBS of 6.34 and a size of 23. The three decision nodes are the following:

$$Z1 = \frac{x_0 \cdot x_1}{1.57} \leq (x_0 \cdot x_1)(x_1^2 \cdot x_0^2) \quad Z_2 = 1 \quad Z_3 = 1$$

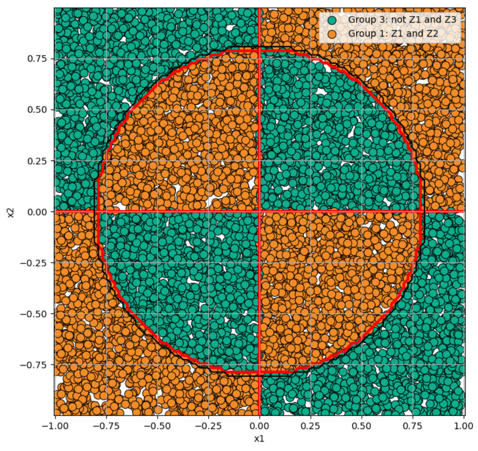


Figure 4.4: Boundaries of the best performing GP-GOMEA-ST in the approximation front on the non-noisy data on seed 3

Figure 4.5 shows the GP-GOMEA-ST outliers in the noisy XOR problem. The red function depicts the boundary created with the function Z_1, Z_2 is the blue contour, and Z_3 is the yellow contour. For seed 7, the GP-GOMEA-ST with the lowest IBS has a value of 7.03 and a size of 44. Its expressions, once simplified, are as follows:

$$Z_1 = x_0^2 \leq \left(x_0 \leq \frac{x_0}{x_1}\right)^2, \quad Z_2 = \left(1 - x_1^2 - x_1^4\right) \leq \left(x_0^2 \cdot 1.778\right), \quad Z_3 = 1 \leq (x_1 - x_0)^2 - (x_1 x_0)^2.$$

For seed 9, the GP-GOMEA-ST with the lowest IBS has a value of 0.0717 and a size of 42. Its expressions are as follows:

$$Z_1 = \left(x_1^4 - \frac{\frac{x_0}{x_1}}{x_1^2}\right) \leq (x_1 - x_0)^4, \quad Z_2 = 1 \leq (x_1 + x_0)^4, \quad Z_3 = (x_0 - x_1)^4 \leq (1 + x_1^2)^2.$$

For seed 11, the GP-GOMEA-ST with the lowest IBS has a value of 0.0713 and a size of 51. Its expressions are as follows: $Z_1 = \frac{\left(\frac{29.201}{x_3}\right)^2}{x_1^2 \cdot \frac{x_0}{x_1}} \leq 0$, $Z_2 = \left((x_0 - x_1) + \mathbf{1}_{x_1 \leq -0.868}\right)^2 \leq \left(\mathbf{1}_{\frac{x_1}{x_0} \leq x_1 x_0}\right)^2$, and $Z_3 = (x_1 + x_0)^4 \leq \frac{x_1}{x_1} + x_0 x_1 - (x_0 - x_1)^2$.

For seed 12, the GP-GOMEA-ST with the lowest IBS has a value of 7.04 and a size of 35. Its expressions are as follows:

$$Z_1 = \left(\frac{x_0}{x_1} \leq x_0\right)^2, \quad Z_2 = 1 - (x_1 + x_0)^2 \leq (x_0 - x_1)^4, \quad Z_3 = (x_1 + x_0)^4 \leq 1 - (x_1 - x_0)^2.$$

For seed 16, the GP-GOMEA-ST with the lowest IBS has a value of 7.12 and a size of 41. Its expressions are: $Z_1 = x_0 x_1 \leq (x_0^2)^4$, $Z_2 = (x_1 + x_0)^4 \leq 1 - (x_1 - x_0)^2$, and $Z_3 = 1 - (x_1 - x_0)^2 \leq (x_0 + x_1)^4$.

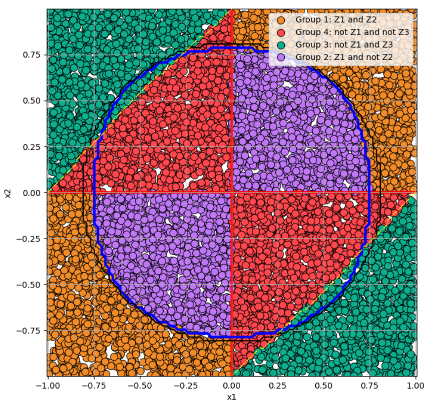
For seed 18, the GP-GOMEA-ST with the lowest IBS has a value of 6.94 and a size of 46. Its expressions are: $Z_1 = (x_1 x_0) \leq 0$, $Z_2 = (x_1 + x_0)^4 \leq 1 - (x_1 - x_0)^2$, and $Z_3 = \left((x_0 - x_1) + \frac{x_0}{-14.069}\right)^2 \leq 1 - (x_0 x_1) - (x_1^2 + x_0^2)$.

Table 4.2 presents the results of statistical comparisons between the three different survival tree models using paired t-tests and Wilcoxon signed-rank tests on both training and test datasets.

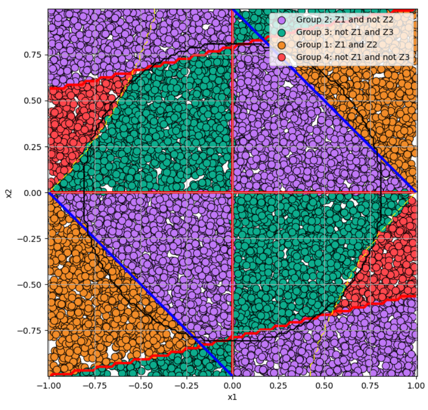
When the SST is compared to GP-GOMEA-ST, both paired t-tests and Wilcoxon tests show highly significant differences ($p < 0.001$) in performance on training and test sets for both the XOR and noisy XOR problems. The difference in IBS between GP-GOMEA-ST and the SST is approximately 3.47 (train) and 2.91 (test) on the XOR problem, and 3.64 (train) and 3.97 (test) on the noisy XOR problem.

Comparing GP-GOMEA-ST on noisy versus non-noisy datasets reveals statistically significant differences ($p < 0.001$) with a small but positive mean difference of approximately 0.37 (train) and 0.36 (test), indicating a slight performance deterioration due to noise.

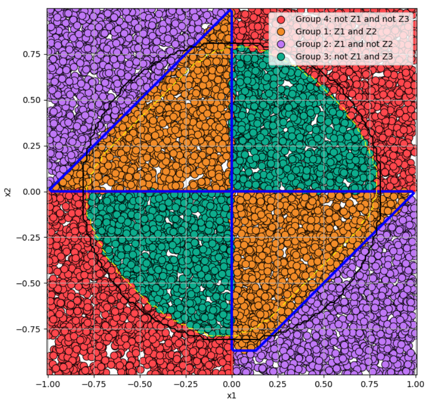
The final comparison is the IST to GP-GOMEA-ST. These comparisons reveal smaller but still statistically significant differences in performance. On the test set, the mean IBS difference is approximately $+0.04$ for the non-noisy data and $+0.42$ for the noisy data. Additionally, the p-values indicate significance with a value of $p = 0.003$ and $p < 0.001$ on the test set for the XOR and Noisy XOR, respectively.



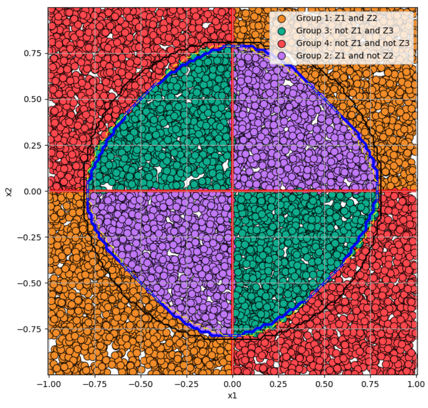
(a) Boundaries of the best performing GP-GOMEA-ST in the approximation front on the noisy data on seed 7



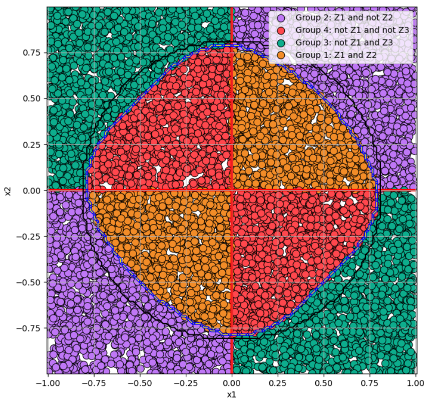
(b) Boundaries of the best performing GP-GOMEA-ST in the approximation front on the noisy data on seed 9

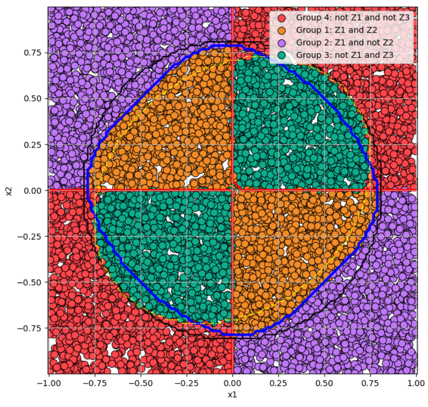


(c) Boundaries of the best performing GP-GOMEA-ST in the approximation front on the noisy data on seed 11



(d) Boundaries of the best performing GP-GOMEA-ST in the approximation front on the noisy data on seed 12





(e) Boundaries of the best performing GP-GOMEA-ST in the approximation front on the noisy data on seed 16

(f) Boundaries of the best performing GP-GOMEA-ST in the approximation front on the noisy data on seed 18

Figure 4.5: Outliers on noisy XOR problem

Comparison	**Test**	**Train p-value**	**Test p-value**	**Train Mean Diff (95% CI)**	**Test Mean Diff (95% CI)**
Sksurv vs GP-GOMEA	Paired t-test	$< .001$	$< .001$	-3.47 $[-3.62, -3.31]$	-2.91 $[-2.94, -2.87]$
	Wilcoxon	$< .001$	$< .001$	–	–
Sksurv vs GP-GOMEA (noisy data)	Paired t-test	$< .001$	$< .001$	-3.64 $[-3.89, -3.38]$	-3.97 $[-4.09, -3.85]$
	Wilcoxon	$< .001$	$< .001$	–	–
GP-GOMEA vs GP-GOMEA on the noisy data	Paired t-test	$< .001$	$< .001$	$+0.37$ $[0.26, 0.48]$	$+0.36$ $[0.25, 0.48]$
	Wilcoxon	$< .001$	$< .001$	–	–
Ideal vs GPGOMEA	Paired t-test	0.042	0.003	$+0.02$ $[0.3, -0.52]$	$+0.04$ $[0.033, 0.047]$
	Wilcoxon	< 0.037	$< .001$	–	–
Ideal vs GPGOMEA (noisy data)	Paired t-test	$< .001$	$< .001$	$+0.4$, $[0.28, 0.51]$	$+0.42$ $[0.30, 0.53]$
	Wilcoxon	$< .001$	$< .001$	–	–

Table 4.2: Statistical Comparison of Model Performances

4.4 Discussion

This experiment aimed to determine if GP-GOMEA-ST can overcome the XOR problem and compare its performance to a greedy survival tree. Additionally, this experiment tried to evaluate the impact noise has on GP-GOMEA-ST.

4.4.1 Performance of GP-GOMEA-ST

The results of this experiment demonstrated that the GP-GOMEA-ST consistently outperformed the SST on both the noisy and non-noisy XOR problem, and it is more robust to noise. As shown in Table 4.1, GP-GOMEA-ST outperforms the SST as it achieves lower IBS values across both datasets. GP-GOMEA-ST reduces the IBS by more than 31% compared to SST on the XOR problem and by more than 42% on the noisy XOR problem. This highlights a substantial gain in predictive accuracy, indicating that GP-GOMEA-ST predicts the survival function better than the SST. Not only does GP-GOMEA outperform the greedy survival tree, but it is also capable of finding the optimal solution. The near-zero score on the Δ train for the XOR problem shows that GP-GOMEA-ST can find the optimal solution for the XOR problem. Figure 4.2 shows that even in the presence of noise, GP-GOMEA successfully identified the optimal solution, whereas the SST did not approach the optimal solution. The low IBS reflects better survival probability estimation, hence the ability to model the complex interactions in XOR problems. On the other hand, the high IBS for the SST is consistent with the known limitation of standard survival trees, which typically struggle with highly non-linear, non-separable problems like XOR. Table 4.2 further proves that GP-GOMEA-ST outperforms the SST by as much as on both XOR problems, the comparison yields a p-value of $p < 0.001$ on both the paired-t test and the Wilcoxon test. This shows that GP-GOMEA-ST is statistically significantly better than the SST. This means that GP-GOMEA-ST is better equipped to handle complex datasets where potential non-linear interactions may happen and noisy features are present.

Unlike the SST, GP-GOMEA-ST can also optimize its structure. This can be seen in Figure 4.4, where GP-GOMEA-ST managed to plot the XOR problem using only one decision node. The other two decision nodes are void of elements. The reason why the structure was also optimized is that GP-GOMEA-ST optimizes the IBS as well as the size, due to the algorithm using multi-objective optimization. This ability is essential in practice as it makes the subsequent survival trees more interpretable for humans.

While GP-GOMEA-ST outperforms the SST, the SST exhibits lower variance between training and testing performance than GP-GOMEA-ST. As shown in Table 4.1, GP-GOMEA-ST has low training and high testing variance, especially in the case of the noisy XOR problem. This trend may indicate overfitting in GP-GOMEA-ST. If this is indeed the case, it may imply that GP-GOMEA-ST is likely to get stuck in a local optimum during the search process and be unable to grasp the structure of the data fully. This trend can also be visualized in Figure 4.2, where both GP-GOMEA-ST plots show greater variance than the SST counterpart. The reason behind the potential overfitting is likely the EA aspect of GP-GOMEA, which may favor more complex models that fit the training data closely. A consequence is that GP-GOMEA-ST may not be able to generalize properly.

4.4.2 The effect of Noise on GP-GOMEA-ST

Table 4.2 indicates that noise significantly negatively impacts the performance of GP-GOMEA-ST. This can be seen from the p-value of $p < 0.001$ on both the paired-t test and the Wilcoxon test when comparing GP-GOMEA-ST on both XOR problems. While the effect on the IBS is small, an increase of approximately 0.37. This increase in IBS corresponds to less accurate survival probability estimates when noisy features are present. A potential explanation is that the addition of noisy features complicates the identification of correct building blocks and thus linkage sets within GP-GOMEA. Noise also deteriorates the quality of the expressions in the decision nodes. The outliers from the regular XOR problem displayed in Figure 4.3 both find the two hidden features: the circular feature and the ”plus” feature. The reason seed 12 performed poorly is that the threshold constant is the circular feature is not optimal. On the other hand, Figure 4.5 shows that the best performing GP-GOMEA-ST in seeds 7, 9, and 11 did not find the circular feature. Hence, this explains the poor performance of these seeds on the noisy XOR.

Furthermore, noise appears to increase the variability in GP-GOMEA-ST's performance across different runs, potentially reflecting a higher susceptibility to convergence on local optima or overfitting to noise patterns. This variability emphasizes the challenge noise poses to evolutionary algorithms that rely on linkage learning.

Another notable effect of noise is observed in the structure of the survival trees generated by GP-GOMEA. In the noiseless XOR problem, the root node consistently corresponds to the circular feature. GP-GOMEA-ST employs a multi-objective optimization approach that simultaneously minimizes the IBS and the complexity of the survival tree. As the circular feature contains more nodes than the ”plus” feature, GP-GOMEA should always prioritize the circular feature over the other feature. However, in the noisy XOR problem, the root node shifts to the alternate feature, suggesting that noise can influence feature selection and tree topology, potentially reflecting a compensatory strategy by the model in response to noise.

Chapter 5

Hyperparameter optimization of the GP-GOMEA-ST

The overarching goal of the 2nd experiment is to analyze the impact of the population size and dataset size. More specifically, this experiment aims to observe how the dataset size and population size impact the performance, time, size of the survival trees, and number of generations. Similar to the previous experiment 4, the performance of a survival tree is given by the IBS defined in Section 3.1.3, where a lower score indicates a better performance, and the size of the survival tree. GP-GOMEA-ST will be compared to the Ideal Survival Tree, which serves as an upper bound for the algorithm.

5.1 Test Problem

In this experiment, a single dataset was employed: the noisy XOR dataset (see Section 4.1). The noisy version is preferred over the non-noisy version, as it better reflects real data sets where not all variables are useful. This dataset was chosen because previous experiments demonstrated that the GP-GOMEA-ST was capable of discovering near-optimal decision boundaries for this problem, making it a suitable benchmark for further analysis. Additionally, the XOR problem requires the algorithm to optimize globally in order to have a performant solution. to simultaneously optimize the nodes and structure of the

To study the effect of data set size on model performance, the original noisy XOR data set containing 10,000 samples was used to generate ten subsets of increasing size, ranging from 1,000 to 10,000 data points in increments of 1,000. Stratified shuffle splitting is applied to each subset to maintain the same proportion of censored to uncensored samples found in the original dataset. This procedure ensured that each subset remained representative of the original distribution while allowing for a controlled analysis of the impact of dataset size on the GP-GOMEA-ST's predictive performance and computational efficiency.

5.2 Experimental Setup

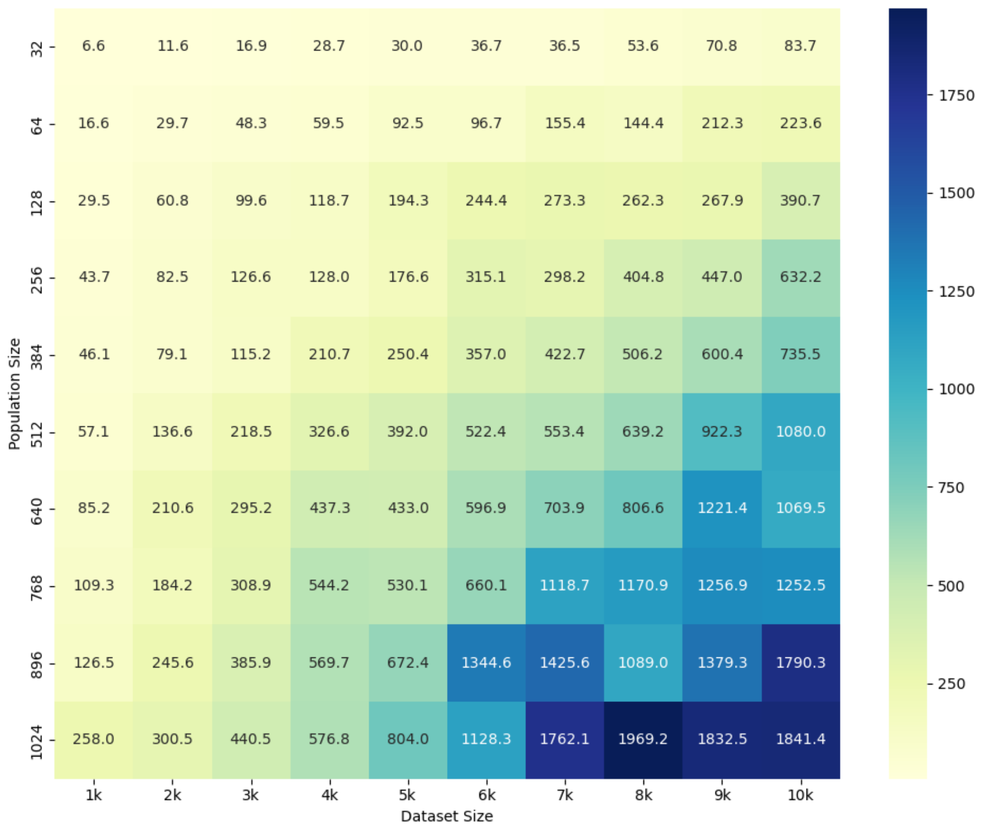
The GP-GOMEA-ST was executed on ten datasets of different sizes and ten different population sizes. A detailed description of the datasets used is provided in Section 5.1. The population sizes tested are 32, 64, 128, 256, 383, 512, 640, 786, 896, and 1024. These sizes span a broad range with gradual increments, allowing the detection of potential performance plateaus. The upper bound of 1024 was chosen because this population size successfully found the ideal decision boundary in the previous experiment (Section 4). Consequently, this experiment aims to determine whether smaller population and dataset sizes can also discover optimal or near-optimal solutions. Additionally, this experiment examines the influence of both dataset size and population size on the algorithm's performance metrics.

To prevent the impact of outlier solutions, each combination of dataset size and population size was evaluated across five different random seeds. The same set of seeds was used for all combinations to ensure fair comparison. For all runs, the approximation fronts were recorded. These fronts contain sets of solutions characterized by training and testing IBS values, tree size, and the corresponding expressions of the survival trees. Furthermore, additional parameters were recorded for each seed, including the number of generations executed and computation time. This information enables a detailed analysis of how population and dataset size affect computational efficiency and predictive performance. For each configuration, the Ideal Survival Tree (IST) was also computed to serve as an upper bound. The IBS value varies with seed, dataset size, and population size as sub-sampling changes the data composition. The IST indicates how close the GP-GOMEA-ST solutions are to the optimal solution.

Afterwards, statistical analysis is done to evaluate differences in performance across population sizes and dataset sizes. While multiple statistical models were considered, such as the Kruskal–Wallis test [25] and the Brown–Forsythe test [3], due to the predictor variables being categorical and the outcome variable being quantitative, the most suitable statistical test is the ANOVA [10, 11]. A two-way ANOVA was employed to examine the main and interaction effects of the p on the various factors. Additionally, when no interaction is recorded, post hoc pairwise comparisons are performed using Tukey's HSD test [43] to identify specific group differences.

5.3 Results

In the upcoming section, all the IBS values are multiplied by 100 for clarity. For all heatmaps, the X-axis depicts the varying dataset sizes, ranging from 1,000 to 10,000 data points. The Y-axis corresponds to the population size, varying from 32 to 1024. The value in each cell was calculated by averaging over all seeds with the same combination of population and dataset size. Warmer, whiter colors indicate low values, whereas cooler, colder tones represent higher values.



Population Size	1k	2k	3k	4k	5k	6k	7k	8k	9k	10k
32	6.6	11.6	16.9	28.7	30.0	36.7	36.5	53.6	70.8	83.7
64	16.6	29.7	48.3	59.5	92.5	96.7	155.4	144.4	212.3	223.6
128	29.5	60.8	99.6	118.7	194.3	244.4	273.3	262.3	267.9	390.7
256	43.7	82.5	126.6	128.0	176.6	315.1	298.2	404.8	447.0	632.2
384	46.1	79.1	115.2	210.7	250.4	357.0	422.7	506.2	600.4	735.5
512	57.1	136.6	218.5	326.6	392.0	522.4	553.4	639.2	922.3	1080.0
640	85.2	210.6	295.2	437.3	433.0	596.9	703.9	806.6	1221.4	1069.5
768	109.3	184.2	308.9	544.2	530.1	660.1	1118.7	1170.9	1256.9	1252.5
896	126.5	245.6	385.9	569.7	672.4	1344.6	1425.6	1089.0	1379.3	1790.3
1024	258.0	300.5	440.5	576.8	804.0	1128.3	1762.1	1969.2	1832.5	1841.4

Figure 5.1: Heatmap of the Time taken in minutes until completion of the algorithm, averaged out over 5 seeds based on the dataset and population size

5.3.1 Computational objectives.

Heatmap 5.1 shows the average time taken in minutes to complete the GP-GOMEA-ST. A correlation between the time taken and the dataset and population size can be observed. Higher dataset sizes or population sizes result in longer times taken. The maximum value is 1,969 minutes, which is approximately 33 hours. This is achieved with a population size of 1,024 individuals and a dataset size of 8,000. The minimum value in the heatmap is 6.6 minutes. This is achieved with a population of 32 and a dataset of size 1,000. Some key outliers are the maximum value and the values 1,344, 1,425, 1,089, and 1,762. The increase from a population of 32 to 1024 translates to an average increase of $2,918\%$ in time with a standard deviation of 827. The maximum increase in time happens with the dataset of size 7,000, with an increase of more than $4,721\%$. The minimum increase is observed with the dataset of size 4000, resulting in an increase of $1,911\%$. The impact of increasing the dataset size from 1,000 to 10,000 results in an average increase of $1,241\%$ in time. Over the 10 different population sizes, the standard deviation is 287. The maximum increase in time occurs with a population size of 512, resulting in a $1,792\%$ increase in time. The minimum occurs at a population size of 1024, resulting in a time increase of only 613%.

Table 5.1: Two-Way ANOVA Summary Table for the Effect of Population Size and Dataset Size on Time

	Sum_sq	df	F	p
Dataset size	1.49e+11	9.0	131.31	$< .001$
Population size	1.95e+11	9.0	170.8	$< .001$
Interaction	7.93e+10	81.0	7.73	$< .001$
Residual	5.06e+10	400.0		

A two-way ANOVA Table 5.1 was conducted to investigate the effects of population size and dataset size on the time required to complete the algorithm, as well as their interaction. The table displays a significant main effect of population size and dataset size, as well as an interaction of time. The effect of population size has an F value of 170.8 and a significant p-value ($p < .001$). The effect of the dataset size has an F value of 131.31 and a significant p-value ($p < .001$). Lastly, the interaction had a lower F-value of 7.73 and a significant p-value ($p < 0.01$).

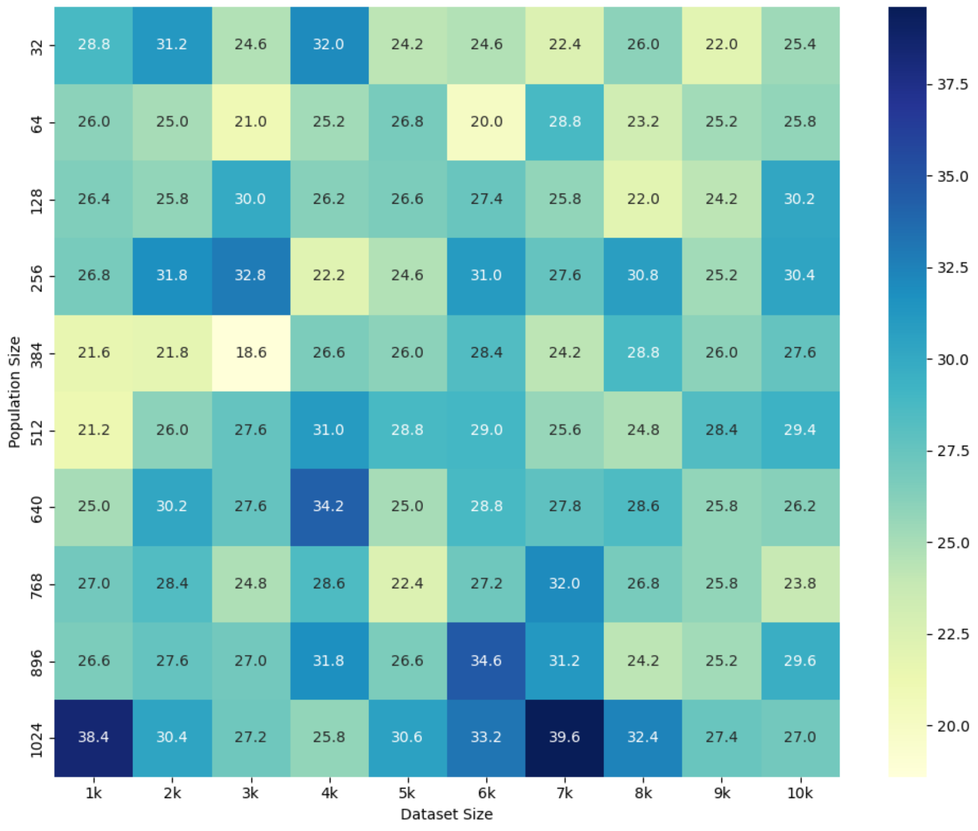


Figure 5.2: Heatmap of the number of Generations until completion of the algorithm, averaged out over 5 seeds, based on the dataset and population size

The following heatmap 5.2 displays the average number of generations required for the algorithm to terminate across five seeds. The heatmap reveals a lack of correlation between the number of generations and the dataset size and population size. Overall, the average number of generations across this heatmap is 27.2. The average maximum number of generations is 39.6. This occurs when the population size is set to 1024 and the dataset size is set to 7,000. The average minimum number of generations is 18.6. This occurs when the population size is set to 384 and the dataset size is 3,000. In addition to the maximum and minimum values, there are two other noticeable outliers: a low value of 20.0 at a population size of 64 and a dataset size of 6,000, and a high value of 38.4 at a population size of 1024 and a dataset size of 1,000.

Table 5.2: Two-Way ANOVA Summary Table for the Effect of Population Size and Dataset Size on Number of Generations

	Sum_sq	df	F	p
Dataset size	5.46e+2	9.0	0.91	0.516
Population size	1.60e+3	9.0	2.67	0.005
Interaction	4.45e+3	81.0	0.82	0.856
Residual	2.67e+4	400.0		

A two-way ANOVA Table 5.2 was conducted to investigate the effects of population size and dataset size on the number of generations required to complete the algorithm, as well as their interaction. The table displays a significant main effect of population size on the number of generations. The effect of population size has an F value of 2.67 and a significant p-value of 0.005. On the other hand, dataset size and the interaction do not appear to have a significant effect on the number of generations.

Table 5.3: Tukey HSD Post-hoc Comparison Between Groups

Comparison	**Mean Diff**	**p(adjusted)**	**Lower CI**
64 - 1,024	6.50	$[1.39, 11.61]$	0.003
384 - 1,024	6.24	$[1.13, 11.35]$	0.005
32 - 1,024	5.08	$[-0.03, 10.19]$	0.053

Table 5.3 is the result of the Tukey's HSD post hoc test to examine pairwise differences between population sizes. The comparison revealed that the population size of 1,024 differs significantly from the other population sizes. Between population sizes 32 and 1,024, there is a mean difference of 6.50, a 95% confidence interval of [1.39, 11.61], and a p-value of .003. Additionally, population size 512 and 1,024 differ with a mean difference of 6.24, 95% confidence interval of [1.13, 11.35], and p-value of .005. While no other pairwise comparisons reached statistical significance, the comparison of population sizes 32 and 1024 has an almost significant p-value.

5.3.2 Performance metrics of GP-GOMEA-ST

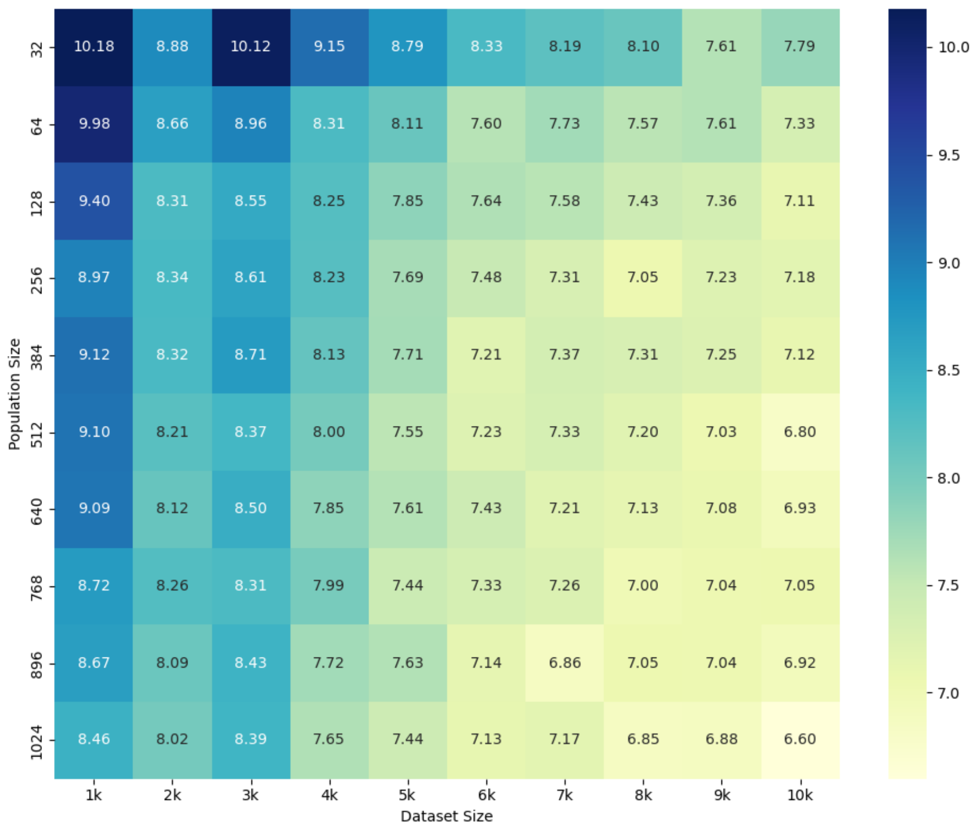


Figure 5.3: Heatmap of the average IBS of the best performing GP-GOMEA-ST in the approximation front over 5 seeds

The heatmap 5.3 depicts the average over 5 different seeds of the GP-GOMEA-ST IBS. In general, the heatmap suggests that a higher population size and a larger dataset yield a better IBS score. Overall, the increase from a population of 32 to 1024 yields an average improvement of 14.27% for the IBS. The maximum improvement is 16.91% for the dataset of size 1,000, and the minimum improvement is 9.58% for the dataset of size 9,000. The increase in the number of data points in the data sets produces an average improvement of 22.65% for the IBS. The maximum improvement for the ten different datasets is 26.49% for an identical population of 64, and the minimum improvement is 19.15% for a population of 768. Lastly, for equivalent population sizes, the dataset with 3,000 samples consistently exhibited lower IBS values than the one with 2,000 samples.

A two-way ANOVA, Table 5.4, was done to examine the effect of population size and dataset size on the IBS of GP-GOMEA-ST. The table displays a significant main effect of population size and dataset size, as well as an interaction on the IBS. The effect of population size has an F value of 80.67 and a significant p-value ($p < .001$). The effect of the dataset size has an F value of 294.31 and a significant p-value ($p < .001$). Lastly, the

Table 5.4: Two-Way ANOVA Summary Table for the Effect of Population Size and Dataset Size on the IBS

	Sum_sq	df	F	p
Dataset size	2.21e-2	9.0	294.31	$< .001$
Population size	6.07e-3	9.0	80.67	$< .001$
Interaction	9.84e-4	81.0	1.45	0.01
Residual	3.34e-3	400.0		

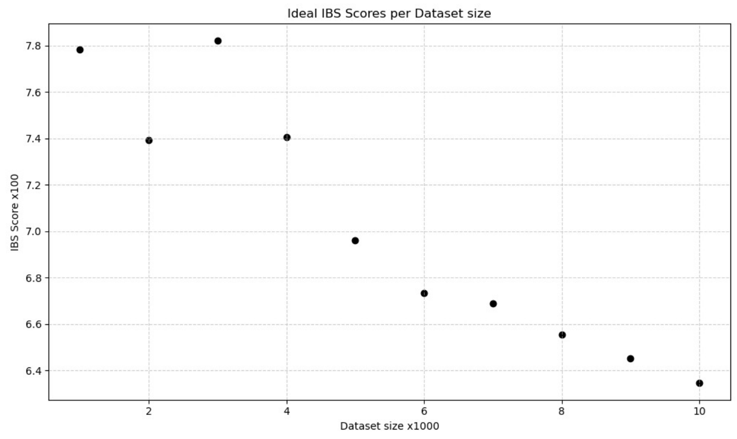


Figure 5.4

interaction had a lower F-value of 1.45 and a lower, but still significant, p-value $(p < 0.01)$.

Figure 5.4 shows the sole impact of the dataset size on the IBS of the IST. When examining the ideal IBS given by the IST, it is apparent that the higher the number of data points in the set, the better the IBS score. There is an 18.5% improvement in IBS score, going from 7.783 to 6.347, as the number of data points increases from 1,000 to 10,000. However, this is not strictly an improvement, as can be seen for 3000 data points, the IST reaches a peak IBS of 7.819 and then steadily decreases.

Figure 5.5 is a heat map showing the difference in IBS values based on the dataset size and population size. This is done by subtracting the IBS given by the IST from the GP-GOMEA-ST IBS values. The resulting heatmap reveals a correlation between the difference in IBS scores and both population size and dataset size. Lower dataset sizes and population sizes result in bigger differences. In contrast, higher dataset sizes and population sizes result in lower differences. The maximum difference in this heat map is 2.39. This is achieved with a population size of 32 and a dataset size of 1,000. The minimum value is 0.17, achieved when the population size is set to 896 and the dataset size is 7,000. The increase in dataset size from 1,000 to 10,000 yields an average improvement of 46.31% with a standard deviation of 13.16. The maximum is recoded at 65.17% at a population size of 512. The minimum is 34.83% when the population size is 896. Regarding the increase in population size from 32 to 1024, the result is an average improvement of 73.47% with a standard

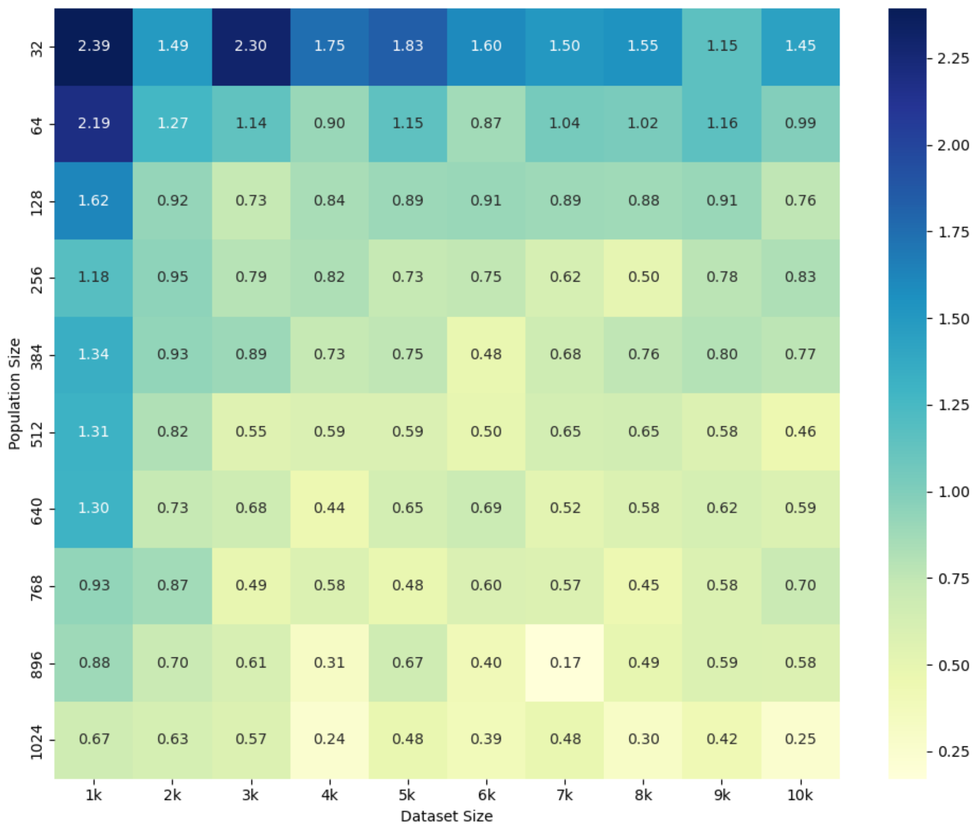


Figure 5.5: Heatmap of the Time taken in minutes until completion of the algorithm for the best performing GP-GOMEA-ST in the approximation averaged out over 5 seeds

deviation of 8.2. The maximum improvement is 82.61%. The minimum improvement is 63.24%.

The heatmap 5.6 depicts the average over 5 different seeds of solution size for the best performing GP-GOMEA-ST. The heatmap reveals a lack of correlation between the solution size and the dataset size and population size. Overall, the average number of generations across this heatmap is 44.3. The average maximum number of generations is 57.6. This occurs when the population size is set to 640 and the dataset size is set to 4,000. The average minimum number of generations is 31.8. This occurs when the population size is set to 128 and the dataset size is 1,000. In addition to the maximum and minimum values, there are four other noticeable outliers: two low values of 34.4 and 35.8, both with a population size of 384 and a dataset size of 2,000 and 4,000, respectively. The other values are 53.0, with a population size of 32 and a dataset size of 9,000, and 52.0, with a population size of 1,024 and a dataset size of 2,000.

A two-way ANOVA, Table 5.5, was done to examine the effect of population size and dataset size on the solution size of GP-GOMEA-ST. The table displays a significant main effect of population size on the solution size. The effect of population size has an F value of

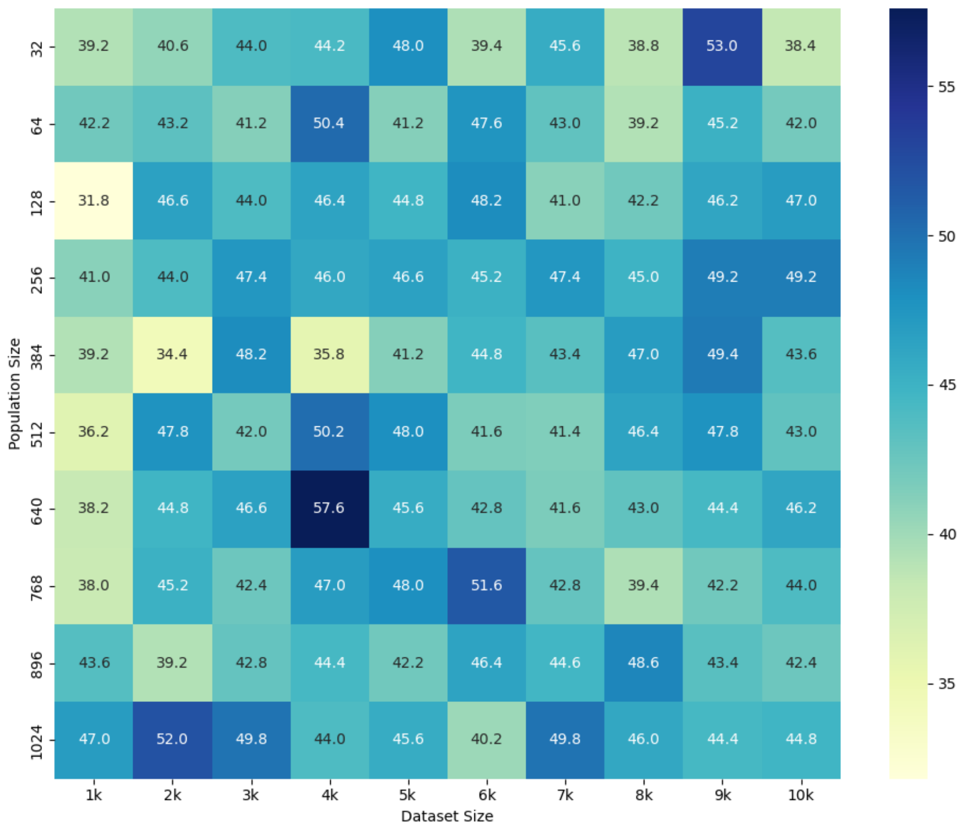


Figure 5.6: Heatmap of the Solution Size for the best performing GP-GOMEA-ST averaged out over 5 seeds

Table 5.5: Two-Way ANOVA Summary Table for the Effect of Population Size and Dataset Size on the Size of GP-GOMEA Solutions

	Sum_sq	df	F	p
Dataset size	7.19e+2	9.0	1.29	0.24
Population size	1.72e+3	9.0	3.08	0.001
Interaction	5.93e+3	81.0	1.18	0.15
Residual	2.48e+4	400.0		

3.08 and a significant p-value of 0.001. On the other hand, dataset size and the interaction do not appear to have a significant effect on the solution size.

Table 5.6 is the result of the Tukey's HSD post hoc test to examine pairwise differences between population sizes. The comparison revealed that the population size of 32 differs significantly from those of the other populations. Between population sizes 32 and 128, there is a mean difference of 5.46, a 95% confidence interval of [0.36, 10.56], and a significant p-value of .025. The population sizes of 32 and 256 differ by a mean difference

Table 5.6: Significant pairwise comparisons using Tukey's HSD ($\alpha = 0.05$)

Comparison	Mean Diff	95% CI	p (adjusted)
32 – 128	5.46	[0.36, 10.56]	.025
32 – 256	6.96	[1.86, 12.06]	.001
32 – 384	5.48	[0.38, 10.58]	.024
32 – 512	5.14	[0.04, 10.24]	.046
32 – 896	6.88	[1.78, 11.98]	.001

of 6.69, with a 95% confidence interval of [1.86, 12.06] and a significant p-value of .001. The population sizes of 32 and 384 differ with a mean difference of 5.48, a 95% confidence interval of [0.38, 10.58], and a significant p-value of .024. The population sizes of 32 and 512 differ with a mean difference of 5.14, a 95% confidence interval of [0.04, 10.24], and a significant p-value of .046. Lastly, the population sizes of 32 and 896 differ with a mean difference of 6.88, a 95% confidence interval of [1.78, 11.98], and a significant p-value of .001.

While no other pairwise comparisons reached statistical significance, the comparison of population sizes 32 and 1024 has an almost significant p-value.

5.3.3 Boundaries of outliers

The first outlier is when the population size is set to 1024 and the dataset size is 1,000. In Figure 5.7, the red boundary depicts the first expression Z_1, the blue decision boundary represents Z_2 and lastly, the brown dotted line represents Z_3. For seed 1, Figure 5.7a, the best performing GP-GOMEA-ST has an IBS of 8.58. Its expressions are as follows:

$$
\begin{aligned}
Z_1 &= \frac{2^2}{(-6.97) - \frac{x_4}{-26.87}} \leq \left[(x_1^2 - \mathbf{1}_{x_1 \leq x_1}) + (x_0^2)^2\right], \\
Z_2 &= 1 \leq (x_0 + x_1)^4, \\
Z_3 &= x_2 \leq \left[(-22.06)^2 \cdot \frac{x_0}{x_1} \cdot (x_4 \cdot -18.34)^2\right].
\end{aligned}
$$

For seed 2, Figure 5.7b, the best performing GP-GOMEA-ST has an IBS of 8.59. Its expressions are as follows:

$$
\begin{aligned}
Z_1 &= (x_1 - x_0)^4 \leq \frac{11.02x_1 - \frac{6.85}{x_1}}{2x_0 + x_3x_0}, \\
Z_2 &= (x_0 - x_1)^2 - x_1^2 \leq 1, \\
Z_3 &= (x_1^2 + 1)^2 \leq (x_0 + x_1)^4.
\end{aligned}
$$

For seed 3, Figure 5.7c, the best performing GP-GOMEA-ST has an IBS of 8.91. Its expressions are as follows:

$$\begin{aligned}
&Z_1 = (7.85x_0)^4 \leq \left[\left(11.02x_1 - \frac{6.85}{x_1}\right) \cdot \frac{-38.29}{x_0}\right],\\
&Z_2 = 0,\\
&Z_3 = (-0.69)^8 \leq 1 - (x_1x_0) - (x_1 - x_0)^2.
\end{aligned}$$

For seed 4, Figure 5.7d, the best performing GP-GOMEA-ST has an IBS of 8.27. Its expressions are as follows:

$$\begin{aligned}
&Z_1 = \left(\frac{x_0}{x_1} + x_0 \leq 0\right) = \left[(x_1^2 \leq 0.794^2)^2\right],\\
&Z_2 = (-0.798^2 - x_0^2 + 1) \leq (x_0 - x_1)^2,\\
&Z_3 = (x_0 + x_1)^4 \leq (x_1^2 + 1)^2.
\end{aligned}$$

For seed 5, Figure 5.7e, the best performing GP-GOMEA-ST has an IBS of 7.92. Its expressions are as follows:

$$\begin{aligned}
&Z_1 = [(x_0 - 31.20)(x_0x_1)(x_2 + 5.99 + x_3)] \leq \left[(11.02x_1 - \frac{6.854}{x_1})(x_0 \cdot -31.38 + \frac{19.58}{x_0})\right],\\
&Z_2 = 1,\\
&Z_3 = 0.
\end{aligned}$$

For seed 1, Figure 5.8a, the best performing GP-GOMEA-ST has an IBS of 7.11. Its expressions are as follows:

$$\begin{aligned}
&Z_1 = \left[(2x_1)(x_0x_3) \leq \left(\frac{x_3}{x_0} \cdot x_1\right)\right] \leq \left[x_1^2 + (x_1^2)^2\right],\\
&Z_2 = \left[(-6.961)^2 - 46.946 - (x_1 - 0)^2\right] \leq \left[(x_0 - x_1)^2 - \mathbf{1}_{(x_2 - x_2) \leq \frac{x_1}{x_0}}\right],\\
&Z_3 = \left[\left(\frac{x_1 + x_0}{1^2}\right)^2 \leq (-1.030^2)^4\right].
\end{aligned}$$

For seed 2, Figure 5.8b, the best performing GP-GOMEA-ST has an IBS of 6.85. Its expressions are as follows:

$$\begin{aligned}
&Z_1 = x_3^2 \leq \left[-33.254 \cdot \frac{x_1}{x_0} \cdot (x_0 - 14.705)^2\right],\\
&Z_2 = \left[(-0.927)^2 - x_0^2 - (x_1^2)^2\right] \leq \left[(x_0^2 + x_1^2)^2\right],\\
&Z_3 = \frac{(-17.858)^4}{(20.024)^4} \leq (x_0^2 + x_1^2).
\end{aligned}$$

For seed 3, Figure 5.8c, the best performing GP-GOMEA-ST has an IBS of 6.69. Its expressions are as follows:

$$\begin{aligned}
&Z_1 = (-1.030^2)^4 \leq \left[(x_0 - x_1)^2 + (x_1 + x_0)^2\right],\\
&Z_2 = (x_0 + x_1)^4 \leq (x_1 - x_0)^4,\\
&Z_3 = [x_1x_0 - \mathbf{1}_{1=0}] \leq (0.320^2)^4.
\end{aligned}$$

For seed 4, Figure 5.8d, the best performing GP-GOMEA-ST has an IBS of 6.69. Its expressions are as follows:

$$
\begin{aligned}
Z_1 &= (-1.030^2)^4 \leq \left[(x_1 + x_0)^2 + (x_0 - x_1)^2\right], \\
Z_2 &= (x_1 - x_0)^4 \leq (x_1 + x_0)^4, \\
Z_3 &= \left[(x_1^2 \leq x_1)^2 = ((0 \leq x_0)^2)^2\right].
\end{aligned}
$$

For seed 1, Figure 5.8a, the best performing GP-GOMEA-ST has an IBS of 6.95. Its expressions are as follows:

$$
\begin{aligned}
Z_1 &= \left[(x_0^2)^2 + x_1^2\right] \leq \left[1 - x_0^2 - (x_1^2)^2\right], \\
Z_2 &= (x_0^4 \cdot x_1^2) \leq \frac{x_0/36.840}{x_1}, \\
Z_3 &= \left[\left(\frac{x_1}{x_0} \leq x_1^2\right)^2 \leq x_1^2\right].
\end{aligned}
$$

5.4 Discussion

The aim of this second experiment was to determine how the population size and dataset size impact GP-GOMEA-ST's performance across different metrics and solutions found.

5.4.1 Performance metric

Two performance metrics were analyzed in this experiment: the solution size of the GP-GOMEA-ST and the Integrated Brier Score (IBS). The results indicate that solution size is not influenced by dataset size, but rather by population size. The ANOVA (Table 5.5) and subsequent Tukey HSD post-hoc analysis (Table 5.6) show that the solution size of GP-GOMEA-ST differs significantly across several population sizes. Specifically, a population size of 32 yields significantly smaller solutions than population sizes of 128, 256, 384, 512, and 896 ($p < .05$). Smaller solution sizes correspond to simpler and more interpretable decision boundaries; however, they may also indicate limited exploration and, consequently, suboptimal boundaries due to underfitting. When combined with the information from the heatmaps (Figures 5.3 and Figure 5.5), these results suggest that smaller populations indeed lead to underfitting.

For the XOR problem, the optimal solution has a size of 17, whereas the average size of the solutions found was 42—more than twice as large. Examination of the evolved expressions revealed that many of these solutions could be algebraically simplified. This suggests that while GP-GOMEA-ST can discover highly predictive structures, it does so at the cost of interpretability. Furthermore, among the solutions in the solution front, larger trees tended to perform better than smaller ones. However, the model that performs best on the training data is often not the best on the testing data, indicating that it has overfit to the training set. Overall, to achieve better solution sizes, it is essential to maintain a high population size to prevent underfitting. Perhaps a better alternative to solution size would be the ”effective size”. This metric would compute the size of a given survival tree after simplifying the mathematical expressions.

In contrast to the size of solutions generated by GP-GOMEA-ST, the IBS appears to be correlated with both the dataset size and population size, as well as their interaction. This is supported by examining Table 5.4. The increase in population and dataset size leads to an overall improvement of the IBS. The reason for such performance increases is that an increase in population size enables more exploration of the solution space, resulting in a greater variety of solutions. Additionally, a larger population reduces the chance of getting stuck in local optimas.

In regards to the dataset size, this outcome aligns with expectations. A greater number of data points provides a more representative sample of the underlying data distribution; hence, GP-GOMEA-ST can learn decision boundaries that generalize better, reducing the risk of overfitting to noise present in smaller datasets. Additionally, larger datasets enable more reliable estimation of survival probabilities. Figure 5.4 shows that larger dataset sizes generally lead to improved predictive performance, as reflected by the decreasing IBS values of the IST. However, this relationship is not strictly monotonic. The temporary increase in IBS observed at 3,000 data points suggests that performance may depend on the specific data samples used, potentially due to sampling variability or noise sensitivity at intermediate dataset sizes. This observation highlights that smaller datasets may not provide sufficient information to reliably estimate survival probabilities, resulting in unstable model behavior. Furthermore, in real-world applications, the absence of a known optimal solution complicates the evaluation of model quality, making it challenging to determine whether observed performance fluctuations stem from data limitations or inherent model variability.

The decision boundaries generated by GP-GOMEA-ST provide additional insight into the model’s behavior. As illustrated in Figure 5.7c and Figure 5.7e, some models accurately capture the circular decision boundary, while others only approximate it. Interestingly, even with 1,000 data points, GP-GOMEA-ST was able to generate a 1-expression solution to solve the XOR problem. This shows that even in smaller datasets, GP-GOMEA-ST is capable of finding a near-optimal solution.

From the observed results, the circular boundary was found in two out of five seeds for the population size of 1,024 with a dataset of 1,000 points (Figure 5.7), and in four out of five seeds for the dataset shown in Figure 5.8. In both cases, when the circular boundary was identified, the resulting surfaces closely approximated the ideal threshold for that boundary. Importantly, all boundaries produced by GP-GOMEA-ST were multivariate, independent of the population and dataset size, demonstrating its ability to construct complex, nonlinear decision surfaces that traditional univariate survival trees cannot reproduce.

Lastly, the accuracy of the estimated constants improved with larger datasets, suggesting that the availability of more data enhances the numerical stability of the evolved expressions. Conversely, smaller datasets may lead to inaccurate or unstable constant estimation, ultimately limiting the precision of the decision boundaries. This behavior further supports the notion that data quantity plays a critical role in the structural robustness of GP-GOMEA-ST.

Overall, when examining the IBS results, a clear trade-off emerges between dataset size and population size. Smaller datasets require larger population sizes to achieve IBS values approaching the ideal boundaries. However, even with high population sizes, GP-GOMEA-ST can become trapped in a local optimum, resulting in suboptimal decision boundaries. Furthermore, the limited data available in smaller datasets negatively affects the quality of constant estimation, which may further hinder model performance. These findings suggest that future work should investigate whether tuning additional hyperparameters, such as tree depth, can help GP-GOMEA-ST generate more robust and accurate solutions across varying dataset sizes.

5.4.2 Computational metrics

Two computational metrics were analyzed: the execution time of the algorithm and the number of generations required for convergence. The results indicate that the number of generations is primarily influenced by population size rather than dataset size. The ANOVA (Table **??**) and subsequent Tukey HSD post-hoc analysis (Table 5.3) show significant differences in the number of generations across several population sizes. Specifically, a population size of 1,024 required significantly more generations than population sizes of 64 and 384 ($p < .05$). This suggests that larger populations enable broader exploration of the search space, which can delay convergence but helps the algorithm escape local optima.

Across all 500 runs, only 15 (3%) reached the cap of 50 generations, indicating that the algorithm generally converges before the maximum number of iterations is reached. For cases in which the circular decision boundary was not found, early termination likely reflects premature convergence rather than successful optimization. While increasing the early-stopping threshold could mitigate this issue, it is already set to 10 generations (20% of the total). Alternatively, adjusting the mutation rate may promote more effective exploration and reduce the likelihood of premature convergence.

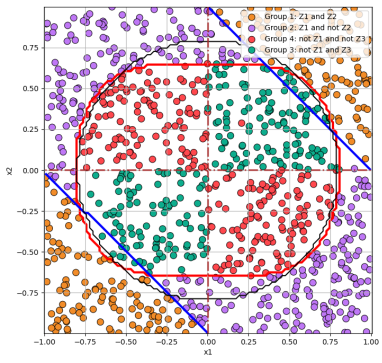
These findings highlight a key computational trade-off in GP-GOMEA-ST: larger populations improve exploration and solution quality but come at the cost of higher computational effort and longer convergence times. Balancing this trade-off is essential for achieving both efficiency and accuracy.

Figure 5.1 illustrates the relationship between dataset size, population size, and computational time for GP-GOMEA-ST. The results show a clear upward trend: as either the dataset size or the population size increases, the computational time rises substantially. This observation is supported by the two-way ANOVA 5.1, which reveals significant main effects for both dataset size and population size ($p < .001$), as well as a significant interaction effect ($p < .001$).

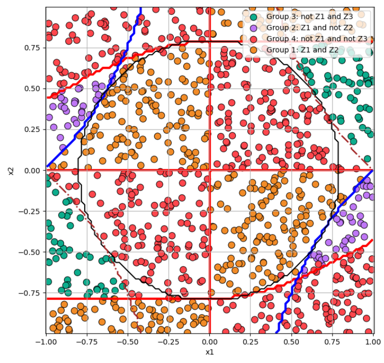
The main effect of dataset size indicates that larger datasets substantially increase computation time, primarily because evaluating individuals becomes more computationally expensive when more samples are used to compute the survival functions and fitness. Similarly, the significant effect of population size suggests that larger populations prolong the optimization process by increasing the number of individuals evaluated per generation. Together, these two factors lead to an increase in computational cost, as reflected in the darker regions of the heatmap.

The significant interaction effect implies that the influence of population size on computational time depends on the dataset size, and vice versa. Specifically, for small datasets, increasing population size leads to modest time increases. However, for larger datasets, computational time escalates sharply with population size, reaching over 1,800 minutes for the largest configurations. This suggests that the joint scaling of both parameters disproportionately amplifies time, possibly due to the combined effects of increased evaluations and longer convergence times.

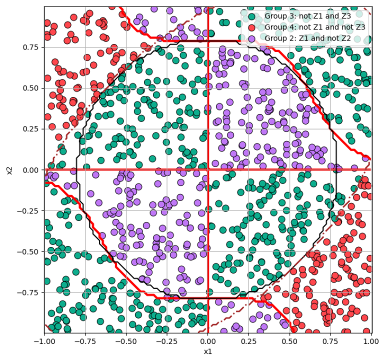
Despite these increases, it is worth noting that the scaling behavior appears consistent and predictable, allowing for the planning of computational resources. These results highlight an important trade-off in GP-GOMEA-ST: while larger populations and datasets improve model robustness and generalization, they do so at the expense of higher computational cost. Future work could explore techniques to mitigate some computational burden without compromising model quality.



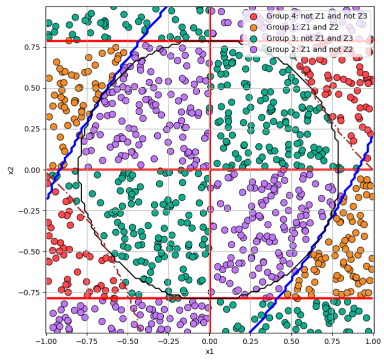
(a) Boundaries of the best performing GP-GOMEA-ST in the approximation front population size 1,024 and dataset size 1,000 on seed 1



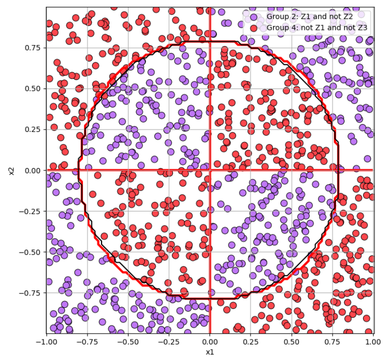
(b) Boundaries of the best performing GP-GOMEA-ST in the approximation front population size 1,024 and dataset size 1,000 on seed 2



(c) Boundaries of the best performing GP-GOMEA-ST in the approximation front population size 1,024 and dataset size 1,000 on seed 3

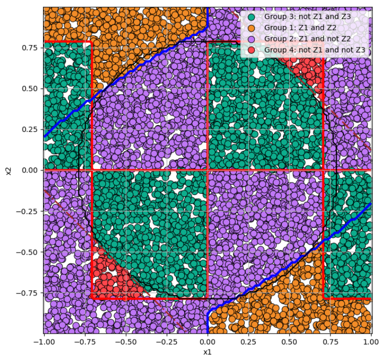


(d) Boundaries of the best performing GP-GOMEA-ST in the approximation front population size 1,024 and dataset size 1,000 on seed 4

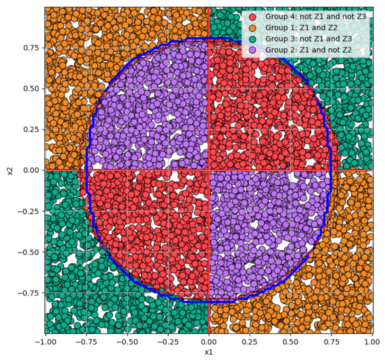


(e) Boundaries of the best performing GP-GOMEA-ST in the approximation front population size 1,024 and dataset size 1,000 on seed 5

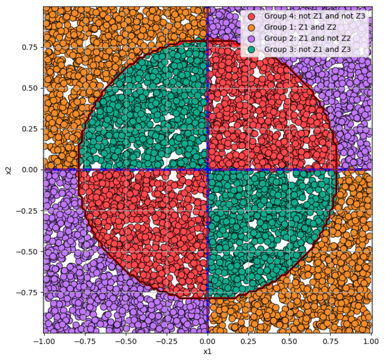
Figure 5.7: Outliers on noisy XOR problem



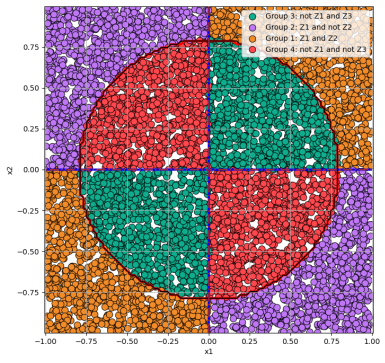
(a) Boundaries of the best performing GP-GOMEA-ST in the approximation front population size 896 and dataset size 7,000 on seed 1



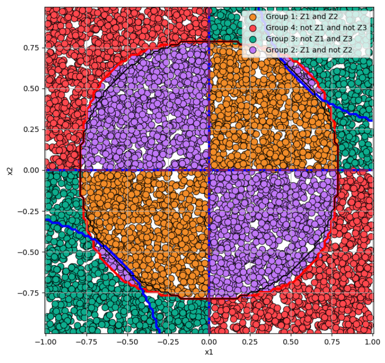
(b) Boundaries of the best performing GP-GOMEA-ST in the approximation front population size 896 and dataset size 7,000 on seed 2



(c) Boundaries of the best performing GP-GOMEA-ST in the approximation front population size 896 and dataset size 7,000 on seed 3



(d) Boundaries of the best performing GP-GOMEA-ST in the approximation front population size 896 and dataset size 7,000 on seed 4



(e) Boundaries of the best performing GP-GOMEA-ST in the approximation front population size 896 and dataset size 7,000 on seed 5

Figure 5.8: Outliers on noisy XOR problem

Chapter 6

Conclusions and Future Work

This thesis aimed to develop a survival tree optimized using GP-GOMEA, enabling the simultaneous optimization of both the structure of the survival tree and the content of decision nodes. The proposed GP-GOMEA-ST was evaluated on a single synthetic benchmark problem and compared to two different survival trees. The results demonstrated that this new approach achieves promising performance in some evaluation criteria. This final chapter summarizes the key findings of the experiments and outlines potential directions for future research on GP-GOMEA-ST.

6.1 Conclusions

Survival trees are interpretable models that allow users to understand the interactions between predictor variables. However, a major limitation of traditional survival trees is their inability to effectively capture nonlinear relationships coupled with their univariate splitting strategy. The proposed survival tree GP-GOMEA-ST addresses these limitations by simultaneously optimizing both the structure and content of the survival tree and its decision nodes using the state-of-the-art evolutionary algorithm GP-GOMEA. As a result, GP-GOMEA-ST is capable of discovering more effective and globally optimized splitting strategies compared to greedy, locally optimized survival trees.

Furthermore, GP-GOMEA was able to find optimal or near-optimal splitting strategies on the synthetic problems. These problems require non-linear splits in order to perform well on them. This suggests that GP-GOMEA-ST is capable of capturing non-linear relationships. Additionally, GP-GOMEA-ST outperformed the SST by over 35% across the different problems. The solution front generated from the multi-objective optimization allows GP-GOMEA-ST to avoid overfitting to the problem compared to single-objective optimization and offers a variety of solutions with different trade-off benefits between interpretability and accuracy.

These findings suggest that incorporating global, multivariate optimization through GP-GOMEA enables survival trees to overcome the inherent limitations of greedy splitting strategies. GP-GOMEA-ST offers a viable pathway toward more accurate, interpretable, and robust survival models. Making them a potentially viable tool for professionals to use to better understand how variables interact to lead to a given outcome.

While GP-GOMEA-ST appears to be a promising approach to survival trees, this thesis overlooks some key limitations. Notably, the algorithm was tested only on synthetic data, and hyperparameter tuning was limited. Hence, the upcoming section discusses potential future work that would expand on GP-GOMEA-ST to solve some of these limitations.

6.2 Future work

In this section, several possible directions for future work are outlined.

Apply GP-GOMEA-ST to real-world survival datasets. While this thesis evaluated GP-GOMEA-ST on synthetic datasets to control complexity and interpret the learned decision boundaries, real-world validation is essential to assess generalizability. Real datasets typically have higher censoring rates, more features, and limited sample sizes (often below 1,000 samples) [26, 8, 29, 9]. Applying GP-GOMEA-ST to such datasets would test its robustness to noise, scalability, and practical predictive performance.

Compare GP-GOMEA-ST against state-of-the-art survival models. In this thesis, GP-GOMEA-ST was primarily compared to a greedy survival tree implementation. Future research should include systematic benchmarking against advanced survival tree methods such as the Optimal Sparse Survival Tree[18], Globally Induced Survival Tree [24], and ensemble approaches like the Random Survival Forest. These comparisons would clarify where GP-GOMEA-ST excels—particularly in terms of interpretability, robustness to noise, and global optimization—and where further algorithmic refinements are necessary.

Extend the hyperparameter tuning of GP-GOMEA-St. Two noticeable factors that were not tuned in this thesis are the depth of the decision nodes and the number of decision nodes. Tuning these hyperparameters will aid in comprehending the implications of these hyperparameters and their impact on the IBS, computation costs, and model complexity associated with scaling these metrics. This would provide a more comprehensive picture of GP-GOMEA-ST's scalability, generalization capacity, and computational efficiency.

Redefine the complexity objective beyond tree size. In this thesis, GP-GOMEA-ST utilized multi-objective optimization, balancing predictive accuracy (IBS) and solution size to create a solution front. However, the size metric did not consistently promote interpretability, as larger models often achieved superior IBS scores. Future work could explore alternative complexity metrics. An alternative to the size metric could be the ”effective” size: the size of a tree after all expressions have been simplified. This metric would have a more better proxy to interpretbility. The advantage of such metric would be that allows the tree to still be big and hence diverse. This alternative metric may yield more meaningful trade-offs between interpretability and accuracy.

Bibliography

[1] Odd Aalen. Nonparametric inference for a family of counting processes. *The Annals of Statistics*, 6(4):701–726, 1978. doi: 10.1214/aos/1176344247.

[2] Paul Blanche, Michael W Kattan, and Thomas A Gerds. The c-index is not proper for the evaluation of t-year predicted risks. *Biostatistics*, 20(2):347–357, 02 2018. ISSN 1465-4644. doi: 10.1093/biostatistics/kxy006. URL `https://doi.org/10.1093/biostatistics/kxy006`.

[3] Morton B. Brown and Alan B. Forsythe. Robust tests for the equality of variances. *Journal of the American Statistical Association*, 69(346):364–367, 1974. doi: 10.1080/01621459.1974.10482955.

[4] Taane G Clark, Michael J Bradburn, Sharon B Love, and Douglas G Altman. Survival analysis part i: basic concepts and first analyses. *British journal of cancer*, 89(2): 232–238, 2003.

[5] Jorge Couchet, Daniel Manrique, Juan Ríos, and Alfonso Rodríguez-Patón. Crossover and mutation operators for grammar-guided genetic programming. *Soft Computing*, 11(10):943–955, 2007.

[6] D.R. Cox. Regression models and life-tables. *Journal of the Royal Statistical Society. Series B (Methodological)*, 34(2):187–220, 1972. doi: 10.1111/j.2517-6161.1972.tb00899.x.

[7] Charles Darwin. *On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life*. John Murray, London, 1859. URL `https://www.gutenberg.org/ebooks/2009`. First edition.

[8] Angela Dispenzieri, Jerry A. Katzmann, Robert A. Kyle, David R. Larson, L. Joseph Melton, Clifton L. Colby, Terry M. Therneau, and S. Vincent Rajkumar. Use of nonclonal serum immunoglobulin free light chains to predict overall survival in the general population. *Mayo Clinic Proceedings*, 83(8):879–885, 2008. doi: 10.4065/83.8.879.

[9] Bradley Efron. Logistic regression, survival analysis, and the kaplan–meier curve. *Journal of the American Statistical Association*, 83(402):414–425, 1988. doi: 10.1080/01621459.1988.10478612.

[10] Ronald A. Fisher. *Statistical Methods for Research Workers*. Oliver and Boyd, Edinburgh, 1925. Foundational text introducing ANOVA and popularizing the ANOVA table.

[11] Ronald A. Fisher. *The Design of Experiments*. Oliver and Boyd, Edinburgh, 1935. Further development and standardization of ANOVA and its tabular presentation.

[12] Alex A Freitas. *Data mining and knowledge discovery with evolutionary algorithms*. Springer Science & Business Media, 2002.

[13] Louis Gordon and Richard A Olshen. Tree-structured survival analysis. *Cancer Treatment Reports*, 69(10):1065–1069, 1985.

[14] James A. Hanley and Barbara J. McNeil. Censoring in survival analysis: Potential for bias. *Medical Decision Making*, 31(6):744–753, 2011. doi: 10.1177/0272989X11410053.

[15] Jr Harrell, Frank E., Robert M. Califf, David B. Pryor, Kerry L. Lee, and Robert A. Rosati. Evaluating the yield of medical tests. *JAMA*, 247(18):2543–2546, 05 1982. ISSN 0098-7484. doi: 10.1001/jama.1982.03320430047030. URL `https://doi.org/10.1001/jama.1982.03320430047030`.

[16] Nicholas Hartman, Sehee Kim, Kevin He, and John D. Kalbfleisch. Pitfalls of the concordance index for survival outcomes. *Statistics in Medicine*, 42(13):2179–2190, 2023. doi: 10.1002/sim.9717.

[17] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, New York, 2nd edition, 2009. doi: 10.1007/978-0-387-84858-7.

[18] Zhanglong Hu, Senne Huisman, Andrada E. Ivanescu, and Jelle J. Goeman. Optimal sparse survival trees. *Bioinformatics*, 40(6):btae321, 2024. doi: 10.1093/bioinformatics/btae321.

[19] 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. doi: 10.1214/08-AOAS169.

[20] Li Jiang, Ming Zhao, Wei Chen, Hui Zhang, and Peng Li. Limitations of using cox proportional hazards model in cardiovascular research. *Cardiovascular Diabetology*, 23(1):59, 2024. doi: 10.1186/s12933-024-02302-2. URL `https://cardiab.biomedcentral.com/articles/10.1186/s12933-024-02302-2`.

[21] John D Kalbfleisch and Ross L Prentice. *The statistical analysis of failure time data*. John Wiley & Sons, 2002.

[22] John P. Klein and Melvin L. Moeschberger. Censoring issues in survival analysis. *Lifetime Data Analysis*, 3(3):255–277, 1997. doi: 10.1023/A:1009675427161.

[23] Malgorzata Kretowska and Marek Kretowski. Global induction of oblique survival trees. In *International Conference on Computational Science*, pages 379–386. Springer, 2024.

[24] Malgorzata Kretowska and Marek Kretowski. Evolutionary induced survival trees for medical prognosis assessment. *Applied Soft Computing*, 170:112674, 2025. ISSN 1568-4946. doi: https://doi.org/10.1016/j.asoc.2024.112674. URL `https://www.sciencedirect.com/science/article/pii/S1568494624014480`.

[25] William H. Kruskal and W. Allen Wallis. Use of ranks in one-criterion variance analysis. *Journal of the American Statistical Association*, 47(260):583–621, 1952. doi: 10.1080/01621459.1952.10483441.

[26] Robert A. Kyle, Terry M. Therneau, S. Vincent Rajkumar, James R. Offord, David R. Larson, Michael F. Plevak, and L. Joseph Melton. A long-term study of prognosis in monoclonal gammopathy of undetermined significance. *New England Journal of Medicine*, 346(8):564–569, 2002. doi: 10.1056/NEJMoa011332.

[27] Michael LeBlanc and John Crowley. Survival trees by goodness of split. *Journal of the American Statistical Association*, 88(422):457–467, 1993.

[28] Hoang N. Luong and Peter A. N. Bosman. Elitist archiving for multi-objective evolutionary algorithms: To adapt or not to adapt. In *Parallel Problem Solving from Nature – PPSN XII*, volume 7492 of *Lecture Notes in Computer Science*, pages 72–81. Springer, 2012. doi: 10.1007/978-3-642-32964-7_8. URL `https://scispace.com/pdf/elitist-archiving-for-multi-objective-evolutionary-3odwrn5ncd.pdf`.

[29] C. A. McGilchrist and C. W. Aisbett. Regression with frailty in survival analysis. *Biometrics*, 47(2):461–466, 1991. doi: 10.2307/2532138.

[30] Wayne Nelson. Theory and applications of hazard plotting for censored failure data. *Technometrics*, 14(4):945–966, 1972. doi: 10.1080/00401706.1972.10488991.

[31] Michael O'Neill. Riccardo poli, william b. langdon, nicholas f. mcphee: A field guide to genetic programming: Lulu. com, 2008, 250 pp, isbn 978-1-4092-0073-4, 2009.

[32] Richard Peto and Julian Peto. Asymptotically efficient rank invariant test procedures. *Journal of the Royal Statistical Society. Series A (General)*, 135(2):185–207, 1972. doi: 10.2307/2344317. URL `https://academic.oup.com/jrsssa/article/135/2/185/7104381`.

[33] Sebastian Pölsterl. scikit-survival: A library for time-to-event analysis built on top of scikit-learn. *Journal of Machine Learning Research*, 21(212):1–6, 2020. URL `http://jmlr.org/papers/v21/20-729.html`.

[34] Shankar Prinja, Nidhi Gupta, and Ramesh Verma. Censoring in clinical trials: review of survival analysis techniques. *Indian Journal of Community Medicine*, 35(2):217–221, 2010.

[35] M Radespiel-Tröger, M Schmid, and G Heinze. Association between split selection instability and predictive error in survival trees. *Computational Statistics & Data Analysis*, 50(12):3401–3420, 2006. doi: 10.1016/j.csda.2005.07.015.

[36] Alain Ratle and Michele Sebag. Avoiding the bloat with stochastic grammar-based genetic programming. In *International Conference on Artificial Evolution (Evolution Artificielle)*, pages 255–266. Springer, 2001.

[37] Peter Rockett. Constant optimization and feature standardization in multiobjective genetic programming. *Genetic Programming and Evolvable Machines*, 23(1):37–69, 2022.

[38] Thalea Schlender, Mafalda Malafaia, Tanja Alderliesten, and Peter A. N. Bosman. Improving the efficiency of gp-gomea for higher-arity operators. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '24)*, pages 971–979. ACM, 2024. doi: 10.1145/3638529.3654118. URL `https://ir.cwi.nl/pub/34530/34530.pdf"`.

[39] E.M.C. Sijben, Tanja Alderliesten, and Peter A. N. Bosman. Multi-modal multi-objective model-based genetic programming to find multiple diverse high-quality models. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '22)*, pages 440–448. ACM, 2022. doi: 10.1145/3512290.3528850. URL `https://ir.cwi.nl/pub/31974/31974.pdf"`.

[40] William Sealy Student. The probable error of a mean. *Biometrika*, 6(1):1–25, 1908. URL `https://seismo.berkeley.edu/~kirchner/eps_120/Odds_n_ends/Students_original_paper.pdf`. Published under pseudonym “Student”.

[41] Dirk Thierens and Peter A. N. Bosman. Optimal mixing evolutionary algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '11)*, pages 975–982. ACM, 2011. URL `https://homepages.cwi.nl/~bosman/publications/2011_optimalmixingevolutionary.pdf`.

[42] Dirk Thierens and Peter AN Bosman. Optimal mixing evolutionary algorithms. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pages 617–624, 2011.

[43] John W. Tukey. Comparing individual means in the analysis of variance. *Biometrics*, 5(2):99–114, 1949. doi: 10.2307/3001913.

[44] Marco Virgolin, Tanja Alderliesten, Cees Witteveen, and Peter AN Bosman. Improving model-based genetic programming for symbolic regression of small expressions. *Evolutionary computation*, 29(2):211–237, 2021.

[45] Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1 (6):80–83, 1945. URL https://sci2s.ugr.es/keel/pdf/algorithm/articulo/wilcoxon1945.pdf. Introduces signed-rank and rank-sum tests.

[46] Samuel S. Wilks. The large-sample distribution of the likelihood ratio for testing composite hypotheses. *The Annals of Mathematical Statistics*, 9(1):60–62, 1938. doi: 10.1214/aoms/1177732360. URL https://projecteuclid.org/journals/annals-of-mathematical-statistics/volume-9/issue-1/The-Large-Sample-Distribution-of-the-Likelihood-Ratio-for-Testing/10.1214/aoms/1177732360.full.

[47] World Health Organization. Cancer. WHO Fact Sheets, February 2025. URL https://www.who.int/news-room/fact-sheets/detail/cancer. [Online; accessed 11-August-2025].