A Tutorial on Interval-Censored Time-to-Event Data Analysis Using R and Python

Rui Alves[1], Aurélio Sidumo[1], Carla Moreira[2][0000−0002−0570−0650], and Luís Meira-Machado[2][0000−0002−8577−7665]

[1] University of Minho, Braga,
ruimiguelalves03@gmail.com
[2] Centre of Mathematics, University of Minho, Braga,
lmachado@math.uminho.pt

Abstract. Interval-censored time-to-event data arise frequently in biomedical and epidemiological studies when the exact time of an event is not observed but is only known to lie within an inspection interval. The analysis of such data requires specialised methods that go beyond classical survival techniques developed for right-censored observations. This tutorial provides a practical and comparative introduction to the analysis of interval-censored survival data using R and Python. Using a synthetic breast cancer dataset, the paper illustrates the estimation of survival functions, group comparisons, and regression modelling under interval censoring, including non-parametric, parametric, semi-parametric, and selected machine-learning approaches. The strengths of the R ecosystem are highlighted, reflecting its mature support for interval-censored data through dedicated packages for non-parametric estimation, hypothesis testing, fully interval-censored regression models, and tree-based methods. In contrast, the Python ecosystem currently offers more limited but increasingly flexible tools, notably through packages such as surpyval, complemented by lifelines and xgboost for midpoint-based regression and boosting approaches. Overall, the tutorial provides a concise and accessible guide for researchers and practitioners interested in the practical analysis of interval-censored survival data.

Keywords: Interval censoring · Survival analysis · Turnbull estimator · Cox and AFT models · Machine learning

1 Introduction

Interval-censored survival data arise when the exact time to an event is not observed but is only known to lie within an inspection interval. This situation is common in medical follow-up studies, screening programmes, and longitudinal cohort designs, where individuals are assessed at discrete time points rather than continuously. In such settings, the event of interest, such as disease recurrence or progression, is known to occur between two successive visits, leading to uncertainty in the true event time. Unlike standard right-censored

data, interval-censored observations cannot be adequately analysed using classical survival methods without additional assumptions. In particular, both the estimation of the survival function and the assessment of covariate effects require methodological adaptations. While midpoint imputation is sometimes used as a pragmatic approximation to convert interval-censored data into a right-censored framework, this approach ignores the width of the censoring intervals and the associated uncertainty, potentially leading to biased inference. These limitations motivate the use of specialised non-parametric and model-based methods that explicitly account for interval censoring.

This paper presents a comparative tutorial on the analysis of interval-censored survival data using R and Python, with an emphasis on practical implementation and methodological understanding. The R ecosystem offers mature and comprehensive support for interval-censored analysis, including Turnbull's non-parametric estimator [1], likelihood-based hypothesis tests, parametric and semi-parametric regression models [2], and tree-based machine-learning methods adapted to interval-censored outcomes. In contrast, Python currently provides a more limited but rapidly evolving toolkit. The `surpyval` package [3] supports Turnbull estimation and parametric modelling under interval censoring, while libraries such as `lifelines`, `scikit-survival` [4], and `xgboost` [5] enable regression and machine-learning approaches based on midpoint imputation.

Using a synthetic breast cancer dataset inspired by classical survival analysis studies, the tutorial illustrates key components of interval-censored data analysis, including data representation, non-parametric survival estimation, group comparison, regression modelling, and selected machine-learning extensions. Throughout the paper, parallel workflows in R and Python are presented to highlight similarities and differences in modelling strategies, software design, and inferential capabilities. The overarching goal is to provide researchers and practitioners with a clear and practical guide for selecting and implementing appropriate methods for interval-censored survival data.

The remainder of the paper is organised as follows. Section 2 introduces the synthetic breast cancer dataset and presents non-parametric methods for survival estimation under interval censoring, including midpoint-based Kaplan–Meier estimation and Turnbull's estimator. Section 3 focuses on regression models for interval-censored survival data, covering midpoint-based Cox and accelerated failure time models as well as fully parametric and semi-parametric approaches with native support for interval censoring. Section 4 discusses selected machine-learning methods for interval-censored data, highlighting differences in availability and implementation between R and Python. Finally, Section 5 summarises the main findings and provides concluding remarks.

2 A Synthetic Case Study in Interval-Censored Survival Analysis

To illustrate the practical implementation of survival analysis methods for interval-censored data, we consider a synthetic dataset inspired by the German Breast

Cancer Study Group (GBSG2) data, which is widely used in the survival analysis literature. The synthetic data are designed to emulate typical clinical follow-up scenarios observed in breast cancer patients, where the exact time to recurrence is generally unknown and can only be inferred to lie between routine diagnostic assessments, such as scheduled clinical visits or mammographic screenings.

In this synthetic dataset, all recurrence times are treated as interval-censored. For each individual, the event of interest is assumed to occur between the last clinical visit at which no recurrence was detected and the first visit at which recurrence is confirmed. Individuals who do not experience recurrence during the follow-up period are represented as right-censored observations through open intervals, either at the time of their last clinical evaluation or at the administrative end of the study. This data structure reflects the inherent uncertainty in the true event time, a defining feature of interval-censored survival data.

To replicate realistic follow-up patterns, diagnostic assessments were scheduled at increasing time intervals after surgery, specifically at 3, 6, 12, 24, 48, and 72 months, mirroring common clinical protocols. To reflect heterogeneity in clinical practice and observation processes, stochastic variability was introduced in the observation times, allowing for irregular follow-up and uncertainty in event detection. This design yields interval-censored outcomes of varying widths, closely resembling those encountered in real-world longitudinal studies.

The explanatory variables included in the analysis comprise age at diagnosis, tumor size (`size`), number of positive lymph nodes (`nodes`), tumor grade (`grade`), menopausal status (`menopause`), hormonal therapy (`hormone`), and levels of progesterone (`prog_recp`) and estrogen receptors (`estrg_recp`). These covariates are well established prognostic factors in breast cancer research and provide a meaningful framework for illustrating regression and machine-learning methods under interval censoring. Prior to the analysis, categorical variables in the synthetic dataset (`bcint`) were appropriately recoded as factors to ensure correct interpretation in the survival models.

2.1 Non-Parametric Survival Estimation

Standard survival methods developed for exact or right-censored event times, such as the Kaplan–Meier estimator [6], which are implemented in widely used survival analysis libraries (`survival` and `lifelines`) in both R and Python, are not directly applicable in the presence of interval censoring. Although midpoint imputation is sometimes adopted as a pragmatic approximation and an initial exploratory step, it ignores the width of the censoring intervals and may lead to biased survival estimates. Consequently, specialised methods are required to properly analyse interval-censored survival data.

A principled non-parametric approach for interval-censored data is provided by Turnbull's estimator [1], which extends the Kaplan–Meier framework to settings where event times are only known to lie within intervals. The estimator is defined through a likelihood-based formulation and is computed using an Expectation–Maximization (EM) algorithm that allocates probability mass to a set of disjoint intervals compatible with the observed censoring structure. This

results in a stepwise estimate of the survival function that explicitly accounts for uncertainty in event timing.

In R, Turnbull's estimator and related interval-censored methods are implemented in dedicated packages `survival` and `interval` [7], which provide native support for interval-censored data. In Python, non-parametric estimation under interval censoring can be performed using the `surpyval` package, which explicitly implements Turnbull's estimator. Exploratory midpoint-based Kaplan–Meier estimation in Python is typically carried out using general-purpose survival analysis libraries such as `lifelines`. Although differences exist in software design and data handling, the underlying statistical methodology remains equivalent across both environments.

Comparing survival curves obtained using midpoint-based Kaplan–Meier estimation with those from Turnbull's estimator illustrates the effect of interval censoring on survival inference. Kaplan–Meier curves based on midpoint imputation treat interval-censored observations as right-censored at a fixed event time, whereas Turnbull's estimator explicitly accounts for uncertainty by distributing probability mass across the censoring intervals.

Figure 1 displays the estimated survival functions for the full dataset obtained using both approaches. The Kaplan–Meier curve typically exhibits earlier and more abrupt declines in survival, reflecting the artificial concentration of events at midpoint times. In contrast, the Turnbull-based curve shows a more gradual decrease, with step changes aligned with the interval structure of the data, thereby providing a representation that is more consistent with the underlying censoring mechanism.

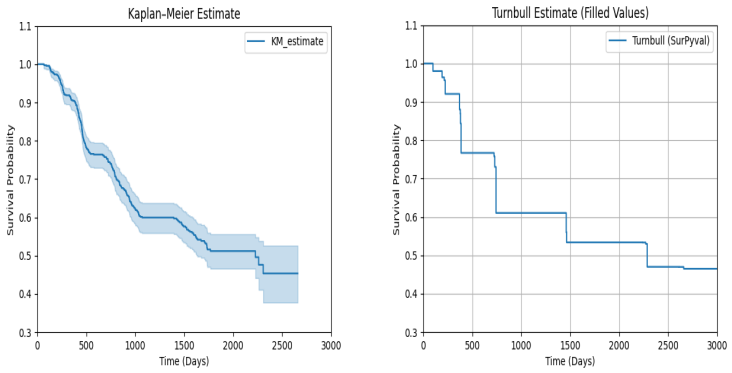

Fig. 1. Estimated survival functions using Kaplan–Meier with midpoint imputation (left) and Turnbull's estimator (right).

When survival curves are stratified by qualitative covariates, such as tumor grade, similar patterns are observed. Midpoint-based Kaplan–Meier curves may exaggerate differences between groups by failing to account for interval width.

In contrast, Turnbull-based curves provide a more stable and conservative representation of group-specific survival patterns by respecting the interval-censored structure within each stratum.

Figure 2 presents survival curves stratified by tumor grade using both estimators. While the relative ordering of the groups is generally preserved, Turnbull-based estimates tend to display smoother separation over time, particularly in later follow-up periods, where censoring intervals are wider.

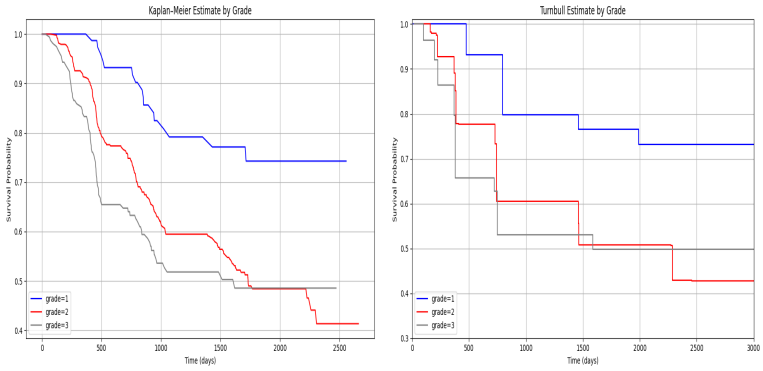

Fig. 2. Kaplan–Meier survival curves with midpoint imputation (left) and Turnbull's estimator with filled right bounds (right), stratified by tumor grade.

In R, these curves are obtained using the `survival` and `interval` packages, while in Python they are produced using `lifelines` for Kaplan–Meier estimation and `surpyval` for Turnbull's estimator. Despite minor differences in implementation and data handling, the qualitative interpretation of the results is consistent across both environments.

When comparing survival curves across groups, the log-rank test is the standard approach for testing the null hypothesis of identical survival functions under right censoring. This test can be performed in R using the `survdiff` function from the `survival` package and in Python using the `multivariate_logrank_test` function from the `lifelines` library. However, when interval censoring is present, the classical log-rank test is no longer valid.

For interval-censored data, the `interval` package in R provides exact and asymptotic likelihood-based tests specifically designed for this setting [8]. These tests offer an appropriate alternative for assessing group differences while properly accounting for censoring intervals.

To further investigate differences between specific pairs of tumor grades, pairwise comparisons of survival curves can be conducted. In the context of right-censored data, this can be achieved using the `pairwise_survdiff` function from the `survminer` package in R, which automatically adjusts for multiple comparisons and reports individual p-values for each group contrast. Under midpoint

imputation, these pairwise comparisons confirm the differences observed in the overall analysis, with statistically significant differences between grade 1 and grade 2 (p = 0.00023) and between grade 1 and grade 3 (p = 7.8e-5). The comparison between grade 2 and grade 3 (p = 0.10144) is not statistically significant at the conventional 5% level, suggesting more similar survival experiences between these two groups.

At present, there is no dedicated function in R for performing automated multiple pairwise comparisons specifically tailored to interval-censored data. Although two-by-two comparisons can be carried out manually by subsetting the data and applying the `ictest` function, this approach becomes impractical when the number of groups is large and does not provide built-in adjustments for multiple testing. Similarly, there is currently no native support in Python for pairwise comparison procedures under interval censoring.

2.2 Regression Models

Regression models play a central role in survival analysis by quantifying the effect of covariates on time-to-event outcomes. Among the most widely used approaches are the Cox proportional hazards model [9] and parametric accelerated failure time (AFT) models. When event times are interval-censored, however, these models cannot be applied directly without additional assumptions.

A common pragmatic solution is midpoint imputation, whereby each censoring interval is replaced by its midpoint, effectively transforming the data into a right-censored format. This strategy enables the use of standard Cox and AFT models as an exploratory baseline, but it ignores the uncertainty associated with the censoring intervals and may lead to biased inference. Alternatively, parametric and semi-parametric regression models can be formulated directly under interval censoring by incorporating interval bounds into the likelihood, thereby preserving the full information contained in the data. In what follows, midpoint-based regression is used as a reference approach, while primary emphasis is placed on regression models that explicitly account for interval censoring and are natively supported in R and Python.

Proportional Hazards Regression Model A straightforward approach to applying Cox regression in the presence of interval censoring consists of replacing each censoring interval by its midpoint, thereby reducing the problem to a right-censored setting. Although this approximation disregards the uncertainty inherent in interval-censored observations, it facilitates the use of standard survival regression tools and provides a useful benchmark for comparison with models that explicitly incorporate interval censoring.

In Python, a Cox proportional hazards model can be fitted to midpoint-imputed event times using the `CoxPHFitter` class from the `lifelines` package. Prior to model fitting, categorical covariates are encoded as dummy variables to ensure compatibility with the regression framework. This approach yields hazard ratio estimates and allows assessment of covariate effects using familiar diagnostic tools. The fitted model suggests that tumour grade, lymph node involvement,

progesterone receptor expression, and hormone therapy are statistically significant predictors of survival under midpoint imputation (Table 1).

In R, an equivalent Cox model can be fitted using the `coxph` function from the `survival` package, again relying on midpoint-imputed times to approximate the interval-censored response. The resulting coefficient estimates and hazard ratios are directly comparable to those obtained in Python and convey similar qualitative conclusions regarding covariate effects.

Table 1. Cox proportional hazards model fitted to midpoint-imputed event times

Covariate	β	$HR = exp(\beta)$	SE(β)	z	p
age	-0.0156	0.9845	0.0097	-1.613	0.1066
size	0.0067	1.0067	0.0041	1.610	0.1074
nodes	0.0471	1.0482	0.0077	6.123	< 0.001
prog_recp	-0.0020	0.9980	0.0006	-3.528	0.0004
estrg_recp	0.0003	1.0003	0.0005	0.560	0.5755
menopause_2	0.3122	1.3664	0.1895	1.647	0.0995
hormone_2	-0.3406	0.7113	0.1348	-2.527	0.0115
grade_2	0.7174	2.0492	0.2635	2.723	0.0065
grade_3	0.8049	2.2366	0.2840	2.835	0.0046

To assess the proportional hazards assumption, diagnostic tests based on Schoenfeld residuals can be applied in both environments. In Python, this assessment is performed using the `proportional_hazard_test` function from `lifelines`, while in R the corresponding test is implemented through the `cox.zph` function in the `survival` package. Evidence of non-proportionality, particularly for tumour grade, suggests that hazard ratio estimates obtained under midpoint imputation should be interpreted with caution and highlights the limitations of Cox regression in this setting.

Accelerated Failure Time (AFT) Model Parametric regression models provide a flexible and principled framework for analysing interval-censored survival data by directly modelling the distribution of event times. Within this class, accelerated failure time (AFT) models offer an appealing alternative to proportional hazards models, allowing covariate effects to be interpreted in terms of acceleration or deceleration of survival time [10]. This interpretation is often more intuitive, particularly when the proportional hazards assumption is questionable.

In both Python and R, parametric AFT models can be fitted under interval censoring using commonly adopted baseline distributions, including Weibull, log-normal, and log-logistic. Model selection in both environments is guided by information criteria; throughout this analysis, competing models are compared using the Akaike Information Criterion (AIC) [11]. Across distributions, the log-normal specification consistently provides the best fit to the data.

In Python, interval-censored AFT models are implemented in the `lifelines` package through parametric AFT fitters with interval-censoring support, such as `LogNormalAFTFitter`. In R, equivalent models can be fitted using the `ic_par` function from the `icenReg` package, which supports both proportional hazards and AFT formulations for interval-censored responses. Across both implementations, regression coefficients are naturally interpreted as time ratios (acceleration factors), providing a direct measure of the multiplicative effect of covariates on survival time (Table 2).

Table 2. Log-normal AFT regression model fitted to interval-censored data. Regression coefficients are reported on the log-time scale; time ratios $\exp(\beta)$ greater than one indicate longer survival times, while values below one indicate shorter times to recurrence.

Covariate	β	SE(β)	z	p	Time Ratio ($\exp(\beta)$)
age	0.0183	0.0082	2.24	0.025	1.018
menopause_2	-0.3321	0.1704	-1.95	0.051	0.717
hormone_2	0.3640	0.1157	3.15	0.0017	1.439
size	-0.0073	0.0037	-1.93	0.053	0.993
grade_2	-0.5935	0.1978	-3.00	0.0027	0.552
grade_3	-0.7100	0.2192	-3.24	0.0012	0.492
nodes	-0.0568	0.0095	-5.97	< 0.001	0.945
prog_recp	0.0015	0.0004	3.58	0.0003	1.002
estrg_recp	-0.0001	0.0004	-0.35	0.724	1.000

The results obtained in R and Python are broadly consistent and identify tumour grade, lymph node involvement, hormone therapy, and progesterone receptor expression as key prognostic factors. Higher tumour grade and nodal involvement are associated with shorter times to recurrence, whereas hormone therapy and higher progesterone receptor levels exhibit protective effects. Variables such as age, tumour size, and menopausal status tend to show weaker or borderline associations across both implementations.

Within the R ecosystem, additional modelling options are available for interval-censored data. Beyond parametric AFT models, the `icenReg` package also provides Cox-type semi-parametric regression models through the `ic_sp` function, which directly accommodate interval censoring without midpoint imputation. Inference for these models is typically obtained using bootstrap procedures, either externally or via the built-in `bs_samples` option. Furthermore, the `flexsurv` package offers a complementary and highly flexible framework for parametric survival modelling, supporting a broader range of distributions as well as advanced tools for prediction and visualisation under interval censoring.

Overall, parametric AFT models form a robust and widely applicable methodological core for regression analysis of interval-censored survival data. While Python currently focuses primarily on fully parametric AFT regression through `lifelines`, the R ecosystem provides a broader array of parametric and semi-

parametric tools via `icenReg` and `flexsurv`, enabling more extensive modelling and inference when required.

2.3 Machine Learning Approaches

In recent years, machine learning methods have become increasingly important in survival analysis, providing flexible alternatives to classical regression models. In particular, tree-based approaches are well suited to capturing complex interactions and nonlinear effects among covariates, while requiring fewer parametric assumptions about the underlying survival distribution. These methods extend naturally to time-to-event data and have been adapted to settings involving censoring, including interval-censored outcomes.

Within this class of methods, survival trees, random survival forests ([12], [13]), and gradient-boosted models [14] have emerged as powerful tools for modelling survival processes from a predictive perspective. In this section, machine learning approaches for interval-censored survival data are explored with an emphasis on tree-based models. We first consider single-tree models, which provide transparent and interpretable representations, and then move to more complex ensemble-based methods.

Survival Trees Survival trees provide an interpretable and flexible approach for modelling survival data, including interval-censored outcomes. By recursively partitioning the data according to covariates that best separate survival times, these trees define groups of individuals with similar risk profiles and naturally capture interactions and nonlinear effects without requiring strong parametric assumptions.

In R, survival trees for interval-censored data can be fitted using the `LTRCtrees` package through the `ICtree()` function. This implementation directly accommodates interval censoring by using the lower and upper bounds of the event time, with right-censored observations handled by setting the upper bound to infinity. The splitting process is guided by statistical significance, leading to a transparent decision structure that highlights variables most strongly associated with survival differences. In the present analysis, the fitted tree identified the number of positive lymph nodes as the most influential variable, followed by tumour grade and progesterone receptor expression. Terminal nodes correspond to distinct risk groups, each associated with its own survival curve, providing an intuitive clinical interpretation of how combinations of covariates affect prognosis.

Figure 3 displays the resulting survival tree, illustrating how recursive partitioning based on lymph node involvement, tumour grade and progesterone receptor expression leads to clinically interpretable risk groups.

In Python, an analogous single-tree model can be obtained using the `xgboost` library under the accelerated failure time (AFT) framework by fitting a gradient-boosted model (more detail in subsection 2.3) with a single tree. Interval censoring is handled natively through the `survival:aft` objective by specifying lower

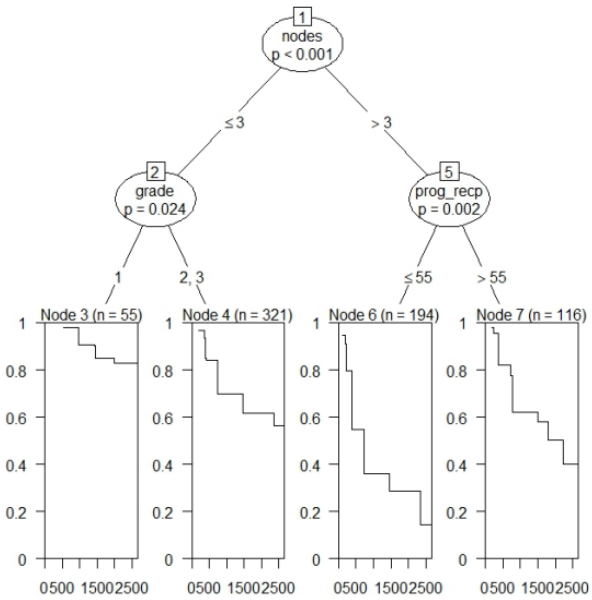

Fig. 3. Visualization of the survival tree for interval-censored survival data using `LTRCtrees`.

and upper bounds for the event time. Unlike the R implementation, splits in `xgboost` are selected by optimizing a predictive criterion based on the AFT log-likelihood rather than formal hypothesis testing. The resulting tree highlights covariates such as hormone receptor expression, tumour size, and nodal involvement as important drivers of survival differences. For interpretability, nonparametric Turnbull survival curves can be estimated within each terminal node, enabling direct comparison with the survival curves obtained from `ICtree()` in R.

Figure 4 shows the corresponding single-tree XGBoost AFT model, highlighting the main splitting variables and the resulting terminal nodes used for risk stratification.

Overall, survival trees in both environments provide an interpretable representation of interval-censored survival data. The R implementation emphasizes inferential transparency through significance-based splitting, whereas the Python implementation prioritizes predictive performance through likelihood-based optimization. Despite these differences, both approaches identify similar prognostic variables and offer a useful foundation for exploring complex covariate interactions, serving as a natural precursor to ensemble methods such as random survival forests and gradient-boosted survival models.

Random Survival Forests Random survival forests extend survival trees by aggregating multiple trees into an ensemble, improving predictive stability and

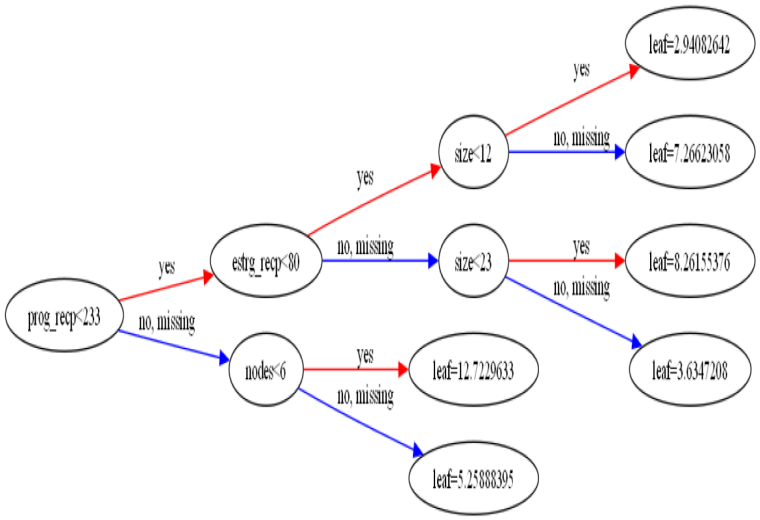

Fig. 4. Single-tree XGBoost AFT model for interval-censored survival data.

accuracy while retaining the flexibility of tree-based models [15]. This approach is particularly useful in survival analysis, where complex nonlinear effects and interactions among covariates are often present. For interval-censored survival data, the `ICcforest` package provides a dedicated implementation of conditional inference forests that explicitly accounts for the interval-censoring structure.

Before fitting the model, right-censored observations, identified by missing values in the upper bound of the censoring interval, are handled by replacing the missing upper endpoint with a large finite value. This step is required because the `ICcforest` implementation does not accept infinite bounds. The random survival forest is then fitted using a large number of trees, allowing the ensemble to reduce the instability typically associated with single survival trees. Key tuning parameters, such as the number of variables considered at each split, are automatically selected based on the data, which simplifies model specification and reduces the need for extensive manual tuning.

An important aspect of random survival forests is the assessment of variable importance. In general, variable importance in random forests can be evaluated using impurity-based measures or permutation-based approaches. Impurity-based importance, commonly used in classical implementations, quantifies the total decrease in node impurity attributed to each variable across all trees in the forest. However, such measures are not fully implemented for `ICcforest` objects in the interval-censored setting. To address this limitation, variable importance is assessed using a permutation-based strategy, whereby the importance of a covariate is quantified by the increase in the Integrated Brier Score (IBS) after randomly permuting its values, as implemented in the `sbrier_IC()` function. This approach provides an interpretable measure of each variable's contribution

to predictive performance. The IBS summarizes prediction error over the follow-up period, with lower values indicating better predictive accuracy. In the present analysis, the random survival forest achieves an IBS of approximately 0.14, representing a clear improvement over the single survival tree and highlighting the benefits of ensemble learning.

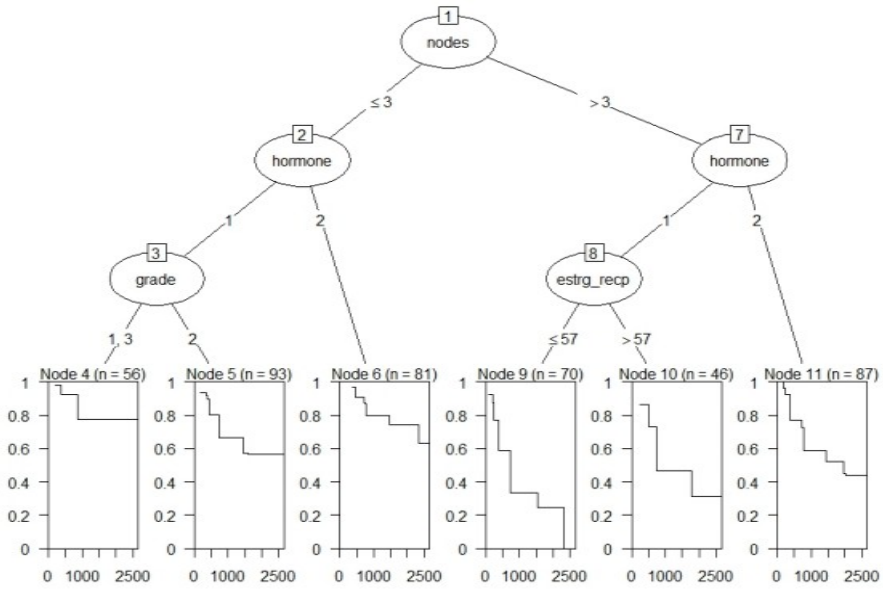

Fig. 5. Visualization of the 50th tree from the fitted random survival forest model (`ICcforest`).

Beyond global performance measures, inspection of individual trees within the ensemble provides insight into the structure of the fitted model. Figure 5 displays the 50th tree of the forest. The variable `nodes` (number of positive lymph nodes) appears most frequently at the top of the tree and often defines the root split, indicating its dominant role in stratifying patients according to survival risk. Other covariates, such as tumour grade, progesterone receptor expression, and hormone therapy, tend to appear in subsequent splits, refining risk stratification within clinically meaningful subgroups.

To further support this finding, the variable selected at the root node is examined across all trees in the ensemble. This analysis confirms that `nodes` is chosen as the root split in the vast majority of trees, reinforcing its central importance in the predictive structure of the model. Survival curves associated with terminal nodes corresponding to different levels of lymph node involvement exhibit clear separation, with higher node counts generally associated with poorer survival outcomes.

Overall, the random survival forest fitted using `ICcforest` provides a robust and flexible framework for analysing interval-censored survival data. Although some interpretability is sacrificed relative to a single survival tree, the ensemble offers improved predictive performance and a more stable representation of covariate effects. The combination of permutation-based importance measures and inspection of individual trees yields meaningful insight into the role of key prognostic variables.

Gradient Boosted Survival Models Gradient boosting is a powerful ensemble learning technique that builds predictive models by sequentially combining weak learners, typically decision trees. In the context of survival analysis, recent developments have extended gradient boosting to accommodate censored data, including interval-censored outcomes. Both R and Python provide implementations of gradient boosted survival models under the Accelerated Failure Time (AFT) framework [10], allowing the censoring intervals to be incorporated directly into the model without resorting to midpoint imputation.

In R, gradient boosted survival models for interval-censored data are implemented through the `xgboost` package using the `survival:aft` objective. Interval censoring is specified by providing lower and upper bounds for the event time, with right-censored observations handled by replacing the upper bound with a large finite value. The model yields individual-level predictions of the expected time to event conditional on the covariates, thereby preserving the uncertainty associated with the censoring intervals and offering a principled alternative to imputation-based approaches.

In Python, the same modelling framework is available through the `xgboost` library, which also supports the AFT objective for interval-censored data. Model fitting is typically combined with validation-based monitoring and early stopping to select the optimal number of boosting iterations. The resulting model emphasizes predictive accuracy through likelihood-based optimization, while still allowing interpretation through tree visualizations, variable importance measures, and survival curve estimation within terminal nodes.

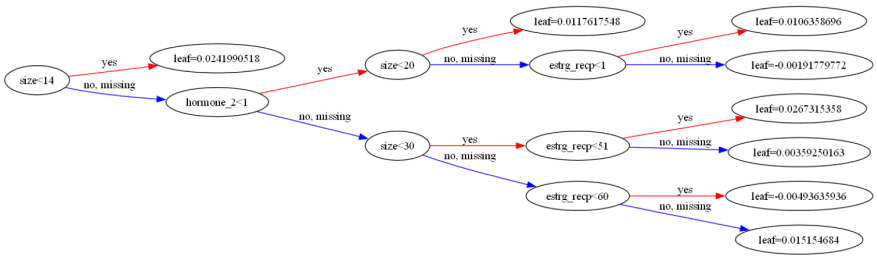

Fig. 6. Decision tree corresponding to the best iteration of the XGBoost AFT model for interval-censored survival data.

Figure 6 displays the decision tree corresponding to the best boosting iteration selected by early stopping in the Python implementation. Unlike classical survival trees, splits are determined by minimizing the AFT negative log-likelihood rather than by formal hypothesis testing. Consequently, the selected splitting variables may differ from those highlighted by inference-oriented approaches, reflecting the predictive focus of gradient boosting.

To facilitate interpretation of survival patterns within each terminal node, nonparametric Turnbull estimators are computed for the observations assigned to each leaf. This enables a direct comparison of empirical survival curves across the subgroups identified by the model. Figure 7 presents the Turnbull survival curves for the terminal leaves of the best-iteration tree. Leaves associated with higher AFT location values correspond to slower declines in survival probability, indicating longer expected times to recurrence, whereas leaves with lower values exhibit poorer survival outcomes.

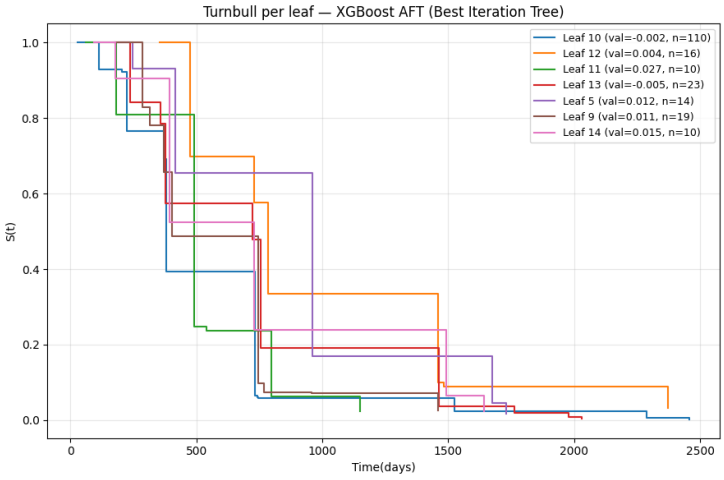

Fig. 7. Turnbull survival curves for terminal leaves of the best iteration tree from the XGBoost AFT model.

Across both R and Python implementations, gradient boosted survival models provide a flexible and powerful approach for modelling interval-censored survival data. Relative to single survival trees and random survival forests, boosting offers enhanced predictive performance and automated variable selection, at the cost of reduced transparency. Nevertheless, the combination of AFT-based predictions, tree visualizations, and Turnbull survival curves enables meaningful interpretation while fully respecting the interval-censored structure of the data.

3 Conclusion

This tutorial has provided a comparative and practical overview of methods for analysing interval-censored time-to-event data using both R and Python. Interval censoring arises naturally in many biomedical and epidemiological studies, and its proper treatment requires specialised statistical tools that go beyond classical survival methods developed for right-censored data. Using a synthetic breast cancer dataset inspired by the GBSG2 study, we illustrated how different modelling strategies address the uncertainty inherent in interval-censored observations.

Non-parametric estimation was first considered, with midpoint-based Kaplan–Meier curves serving as an exploratory baseline and Turnbull's estimator representing a principled approach that fully respects the interval-censored structure. The comparison showed that midpoint imputation may distort survival patterns, whereas Turnbull's estimator yields more conservative and realistic survival estimates. These differences were consistently observed across both software ecosystems.

Regression modelling under interval censoring was then examined using both pragmatic and fully likelihood-based approaches. Although midpoint-based Cox and AFT models are easy to implement and interpret, they rely on simplifying assumptions and may violate key model conditions. In contrast, parametric and semi-parametric models with native support for interval censoring provide a more rigorous inferential framework. Results obtained in R and Python were broadly consistent, identifying tumour grade, lymph node involvement, hormone therapy, and progesterone receptor expression as important prognostic factors, while also highlighting differences in the scope and maturity of the available software tools.

Finally, machine learning methods were explored as flexible alternatives for capturing complex nonlinear effects and interactions. Survival trees, random survival forests, and gradient-boosted survival models exhibited increasing predictive performance at the expense of reduced interpretability. The R ecosystem currently offers more specialised implementations for interval-censored data, particularly for tree-based and inference-oriented methods. Python, although more limited in this respect, has evolved rapidly through libraries such as `xgboost`, enabling native modelling of interval-censored data under the AFT framework and providing competitive predictive performance.

Overall, this tutorial emphasises the importance of selecting methods that are consistent with the censoring mechanism and aligned with the objectives of the analysis. R remains the most comprehensive environment for interval-censored survival analysis, especially when formal inference and specialised models are required. Python, in turn, offers increasing flexibility and strong integration with modern machine learning tools, making it an attractive alternative for predictive modelling. By presenting parallel workflows and stressing methodological understanding, this work aims to support researchers and practitioners in making informed choices when analysing interval-censored survival data.

Code availability

All R and Python code used to reproduce the analyses and figures is available at https://github.com/ruialves-30/data-analytics-projects/blob/main/CodeAppendix_A%20Tutorial%20on%20Interval-Censored%20Time-to-Event/cint_code.txt.

Acknowledgements

This work is funded by national funds through FCT – Fundação para a Ciência e a Tecnologia, I.P., under the UID/00013: Centro de Matemática da Universidade do Minho (CMAT/UM) Program Contract, and the project reference 2023.14897.PEX (DOI: 10.54499/2023.14897.PEX).

References

1. Turnbull, B.W.: The empirical distribution function with arbitrarily grouped, censored and truncated data. Journal of the Royal Statistical Society: Series B (Methodological) **38**(3), 290–295 (1976). DOI 10.1111/j.2517-6161.1976.tb01501.x
2. Sun, J., Zhao, X.: Interval Censoring: Statistical Methods for Censored and Truncated Data. Springer, New York (2006)
3. McNicholas, P.D., McNicholas, T.J., ElSherbiny, A.: surpyval: Survival analysis in python. Journal of Open Source Software **6**(67), 3519 (2021)
4. Pölsterl, S.: 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)
5. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794. ACM (2016)
6. Kaplan, E.L., Meier, P.: Nonparametric estimation from incomplete observations. Journal of the American statistical association **53**(282), 457–481 (1958)
7. Fay, M.P., Shaw, P.A.: interval: Analysis of Interval-Censored Data (2023). URL https://CRAN.R-project.org/package=interval. R package version 2.1.1
8. Fay, M.P., Shaw, P.A.: Exact and asymptotic weighted logrank tests for interval-censored data: The interval r package. Journal of Statistical Software **36**(2), 1–34 (2010). DOI 10.18637/jss.v036.i02
9. Cox, D.R.: Regression models and life-tables. Journal of the Royal Statistical Society: Series B (Methodological) **34**(2), 187–202 (1972)
10. Keiding, N., Andersen, P.K., Klein, J.P.: The role of frailty models and accelerated failure time models in describing heterogeneity due to omitted covariates. Statistics in medicine **16**(2), 215–224 (1997)
11. Akaike, H.: A new look at the statistical model identification. IEEE Transactions on Automatic Control **19**(6), 716–723 (1974). DOI 10.1109/TAC.1974.1100705
12. Hothorn, T., Hornik, K., Zeileis, A.: Unbiased recursive partitioning: A conditional inference framework. Journal of Computational and Graphical Statistics **15**(3), 651–674 (2006)
13. Ishwaran, H., Kogalur, U.B., Blackstone, E.H., Lauer, M.S.: Random survival forests. The Annals of Applied Statistics **2**(3), 841–860 (2008)

14. Zhao, T., McAuley, J., White, R.W.: Deepsurv: Personalized treatment recommender system using a cox proportional hazards deep neural network. In: Proceedings of the Machine Learning for Healthcare Conference (MLHC) (2020)
15. Breiman, L.: Random forests. Machine learning **45**(1), 5–32 (2001)