NEW VARIANTS OF RANDOM FOREST-BASED METHODS FOR SURVIVAL ANALYSIS AND APPLICATIONS TO BIOMEDICAL DATASETS

A THESIS SUBMITTED TO

THE UNIVERSITY OF KENT

IN THE SUBJECT OF COMPUTER SCIENCE

FOR THE DEGREE

OF PHD.

By

Tossapol Pomsuwan

December 2022

Abstract

Survival analysis problems involve predicting the time passed until the occurrence of an event of interest (the target variable), based on the values of some predictive features. Survival analysis is a specific type of supervised machine learning problem where the value of the target variable can be censored — i.e., for some individuals, it may be known only that they "survived" (did not experience the event of interest) until a certain date, whilst it is unknown if the event of interest occurred after that date. Traditional supervised learning methods cannot directly cope with censored data, and so they need to be modified to properly address survival analysis problems.

In this context, this thesis focuses on the random forest algorithm (a popular and powerful supervised learning algorithm), and proposes new variants of random forest (RF) or RF-based algorithms for survival analysis.

The proposed RF or RF-based variants are evaluated on 11 survival analysis datasets created for this research, where the target variable is the time passed until an individual is diagnosed with a certain age-related disease. Most of these datasets were created by extracting relevant data from databases of longitudinal studies of ageing, so that the target variable denotes in general the time passed until an individual is diagnosed with some age-related disease.

This thesis has three main contributions, which involve proposing three new types of variants of RF or RF-based algorithms to cope with censored data in survival analysis problems, as follows.

The first contribution is to propose new RF variants with a modified procedure for creating subsets of training data to be used for learning the decision trees in a RF. This involves replacing the censored value of a target variable by another value which is then treated as an uncensored target value, therefore allowing the other parts of a traditional RF algorithm to be applied without modification. Experiments with the aforementioned 11 survival analysis datasets have shown that the proposed RF variants improved predictive accuracy in general when compared with the standard RF and some standard statistical methods for survival analysis, with statistical significance in some cases. However, the proposed RF variants were outperformed by a standard random survival forest (RSF) algorithm, which is a powerful RF-based algorithm developed specifically for survival analysis.

Motivated by the good performance of the RSF algorithm in the previously mentioned experiments, the second contribution of this thesis is to propose several new variants of the RSF algorithm. The proposed RSF variants focus on modifying two major components of the standard RSF algorithm: (a) the criterion used for feature selection at each node of each tree in the forest, and (b) the procedure used for computing the target variable value predicted by each leaf node of each tree. Experiments with the aforementioned 11 survival analysis datasets have shown that, although the variations in the feature-selection criterion did not lead to significant differences in predictive accuracy, one of the variations in the procedure for computing the values predicted at leaf nodes achieved in general significantly higher accuracies than the standard RSF algorithm and the popular Cox Proportional Hazard (PH) algorithm.

The third contribution is to propose several new variants of the Deep Survival Forest (DSF) algorithm, which essentially learns a more complex survival analysis model by stacking several learned RSF models into layers, inspired by deep learning principles. The proposed DSF variants focus on the base RSF algorithm used

to learn the RSF models at each layer. More precisely, the proposed DSF variants replace the standard RSF algorithm with one of the RSF variants proposed earlier in this thesis, as base learners in each layer. Experiments with the aforementioned 11 survival analysis datasets have shown that one of the proposed DSF variants achieved significantly higher predictive accuracy than the popular Cox PH algorithm and somewhat higher accuracy than the standard DSF in general.

In summary, this research has proposed new variants of RF or RF-based algorithms for coping with censored data in survival analysis problems; and in general the proposed algorithm variants have been shown to be competitive with (sometimes significantly more accurate than) standard methods for survival analysis.

Acknowledgements

In the Thailand, there is a saying, “putting gold leaf on the unseen back of a Buddha statue” meaning performing acts of kindness without seeking recognition or reward. I’d like to offer my heartfelt thanks to everyone who has ‘put gold leaf’ on this thesis.

I would like to honour Prof. Alex Freitas, my supervisor, for his unwavering support and dedication throughout the long gestation of this thesis, and indeed, throughout my whole university life. Alex’s whole attitude towards life has been one that consistently fueled my ambitions to flourish as a researcher. To me, he is a living example of professionalism. He inspired me. Through being under his supervision for so long, I discovered how gifted he is at doing research. Beyond the scope of academic requirements, his consistent willingness to extend his help has truely impacted both my academic growth and personal development.

I also extend my gratitude to the Development and Promotion of Science and Technology Talents Project (DPST), Royal Thai Government, for funding and the effort that its members, Khun Jaruwan and Nantharat most of all, have put into overseeing me for ten years. I cannot imagine living here without money.

I am thankful to my supervisory panel, Dr Marek Grzes and Dr Fernando Otero, and Dr Rogerio de Lemos as the panel Chair, for their constructive feedback that pushed me forward at the progress review every year.

I was touched by the supportive environment provided by School of Computing, particularly the computing equipment that enabled my experiments to

run intensive algorithms. I thank Tim Bishop and Darren Lissenden for their assistance in troubleshooting all the technical issues.

Many thanks to all members of the Computational Intelligence group for all the advice and for holding the weekly seminar series. The presentations from both internal and external speakers and my colleagues broadened my knowledge in the area of Artificial Intelligence.

Among my fellow PhD students, I owe my big thanks to Jakub, Kathryn, Huy and Mahan for their unconditional friendships. Even though covid took us apart, those two years we spent together meant a lot to me and made my PhD experience more enjoyable than I thought it would be.

I would like to thank Prof. Christopher Farmer and Steve Childs, for their collaboration with the Haemodialysis studies. They not only provided the Haemodialysis dataset used in the experiments reported in this thesis, but also kindly offered help with data cleansing.

Besides my research project, I am sincerely thankful to the students who were attending my classes as well as the module convenors for allowing me to teach. Those experiences were a great start for my future career in academia.

To all friends that I met here, specially Bow, Chai, Dawid, Golf, Toei, Tangtang and P'Por, you were literally a bleeding heart of generosity. Always lent me a helping hand in times of need, for which I am very grateful.

Last but not least, my gratitude is owed to my family and my girlfriend, Saowalak. Side by side, we chased our shared dream of earning PhDs — the ideal life in England I could ever wish for. Mum and Dad and Sun not only supported me all the way on this long journey, but also set me up on this path when I was too naive to understand how important this study would be to my future life.

Abbreviations

C-index:: Concordance Index

CHF: Cumulative Hazard Function

Cox PH: Cox Proportional Hazard

DF: Deep Forest

DSF: Deep Survival Forest

DSF-KNN: Deep Survival Forest with K-Nearest Neighbours

ELSA: English Longitudinal Study of Ageing

HD: Haemodialysis

IPC: Inverse Probability of Censoring

K-NN: K-Nearest Neighbours

kNN-RF: k-Nearest-Neighbour-Imputation Random Forest

LB: Lower-bound

ML: Machine Learning

RF: Random Forest

RMSE: Root-Mean-Square Error

RSF: Random Survival Forest

RSF-KNN-mean: Random Survival Forest with K-Nearest Neighbours mean prediction

RTIF: Random Target-Imputation Forest

SHARE: Survey of Health, Ageing and Retirement in Europe

UB: Upper-bound

Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1

Introduction

Supervised Machine Learning (ML) is a very active area of research where algorithms essentially employ training data to build a model capable of making predictions about the value of a pre-defined target variable, based on the values of other variables, called features or attributes (Singh, Thakur and Sharma, 2016). The two main types of machine learning tasks addressed by supervised ML algorithms are classification and regression, where the target variable takes categorical (nominal) or real-valued numeric, respectively.

Among these two types of tasks, regression is more relevant for this thesis. Hence, it is worth first briefly reviewing the basic principles of the regression task.

Regression methods learn a predictive regression model from instances (e.g. patients in medical data) whose values of the target variable are known. This is called the training phase. After a regression model has been created, it is expected to be able to predict well the unknown values of the target variable of instances previously unseen in the training phase, i.e. to show a good generalisation ability for new instances. For example, medical researchers could use a regression method for modelling patients' cancer risk or predicting cancer patients' survival time.

This thesis addresses the task of survival analysis, which consists of a set of statistical or machine learning methods concerned with analysing the time

until the occurrence of an event of interest (Kleinbaum and Klein, 2012). The occurrence of such event is often referred to as a "failure" in the literature, but the event of interest can be a "failure" or a "success", or any other type of event of interest.

To some extent, survival analysis is similar to the regression task, since the target variable to be predicted in survival analysis, the time to an event occurrence, is a numeric variable. A key difference, however, is that survival analysis involves censored data, representing uncertainty about whether an event has occurred for some individuals (instances), which cannot be effectively handled by standard regression methods. In other words, censored data are data where the value of the target variable is just partly known - i.e. it is known that the event did not occur until a certain time point, but it is now known precisely when the event will occur.

Note that censoring is very common in biomedical data. For example, in a dataset of patients who have undergone surgery, say, one year ago, typically there will be a mixture of uncensored and censored patients (instances), as follows. Some individuals are known to have died, so that the value of their "survival time" (after surgery) is precisely known (uncensored). However, for the individuals who are still alive, their survival time is only partially known (censored), i.e., it is only known to be at least one year.

Survival analysis has been extensively studied in the statistical literature (Kaplan and Meier, 1958; Nelson, 1972; Cox, 1972; Aalen, 1978; Simon et al., 2011), but it has been relatively less studied (much less studied than classification and regression) in the supervised ML literature.

Hence, this thesis focus on developing supervised ML algorithms for survival analysis (with censored data). More specifically, regarding the type of supervised ML algorithm, this thesis focuses on variants of the random forest algorithm (Breiman, 2001; Touw, Bayjanov and al, 2013), including mainly the random

survival forest algorithm (Ishwaran and Kogalur, 2007). In essence, random forests are a powerful type of supervised ML method for regression tasks, due to the algorithm's good performance of achieving high predictive accuracy in general, using the power of an ensemble of decision trees to make more robust predictions. Random forest algorithms also have the advantage of being non-parametric (i.e. they do not require any assumption about the data distribution), naturally coping with both numerical and nominal features, and being computationally efficient (relatively fast, particularly for an ensemble method).

Note that, since random forest algorithms were developed for standard regression, they cannot directly cope with censored data. However, random forest algorithms have been adapted to the context of censored data, with the proposal of the random survival forest algorithm (Ishwaran and Kogalur, 2007), which is currently a powerful and popular method for survival analysis in the machine learning literature (Li et al., 2022; Zhang et al., 2022; Snider and McBean, 2022; Miao et al., 2018a; Weeraddana et al., 2020; Gul et al., 2020).

In this context, broadly speaking, the main contribution of this thesis is to propose a number of variants of random forests (mainly random survival forest) algorithms for survival analysis, coping with censored data. The thesis' contributions will be described in more detail in Section 1.2.

The proposed algorithm variants have been evaluated on a number of real-world biomedical datasets, which generally contain data about age-related diseases, and were mainly created by extracting data from longitudinal databases of ageing. More specifically, the datasets were created from three data sources, namely: the English Longitudinal Study of Ageing (ELSA) (Clemens et al., 2019), the Survey of Health, Ageing and Retirement in Europe (SHARE) (Börsch-Supan et al., 2013; Gruber, Hunkler and Stuck, 2014), and the Haemodialysis survey

(UK Renal Registry (UKRR), 2022)[1]. These data sources and the dataset creation process will be discussed in detail in Chapter 3.

The motivation for focusing on datasets of age-related diseases is as follows. According to a World Health Organization's report on World Population Ageing (WHO, 2022), one in six human beings on earth will be 60 or older by 2030. By this point, there will be 1.4 billion people over the age of 60, going up from 1 billion in 2020. The number of persons in the globe who are 60 years or older will double by 2050 (2.1 billion). Furthermore, between 2020 and 2050 the number of people 80 or older is projected to triple, reaching 426 million. Such a large increase in the proportion of elderly people will lead to strong pressure on healthcare systems, considering that the elderly tend to suffer from multiple age-related diseases (George, Elliott and Stewart, 2008).

Hence, there has been a growing need for statistical or supervised ML methods to analyse ageing-related data, including data on age-related diseases (Fabris, de Magalhães and Freitas, 2017; Bha, 2006), in order to better understand and potentially improve the diagnosis of such diseases. Previous studies have shown that supervised ML-based survival analysis methods such as random survival forests can effectively analyse biomedical data with censoring, including data on age-related diseases (Rahman et al., 2021; Leary et al., 2020; Spooner et al., 2020; Akai et al., 2018a; Nakatsu et al., 2018) Hence, this thesis further contributes to this area by applying the proposed variants of random forest-based algorithms to data about age-related diseases.

[1]The data reported here have been supplied by the UKRR of the UK Kidney Association. The interpretation and reporting of these data are the responsibility of the authors and in no way should be seen as an official policy or interpretation of the UKRR or the UK Kidney Association.

1.1 Aims and Objectives

The broad aims of this research consist of developing new variants of random-forest-based algorithms for survival analysis and evaluating those algorithm variants in biomedical datasets with censored data. Most of the proposed algorithm variations involve random survival forests, including deep survival forests that use random survival forests as classifiers. The motivation for focusing on these two types of algorithms is that, among the types of algorithms specifically designed for survival analysis, those two types of algorithms tend to yield state-of-the-art predictive performance results in survival analysis problems, in general. In addition, the thesis also proposes some variations in standard random forest algorithms, in order to adapt those algorithms (which were not designed for survival analysis) to the survival analysis task. The main motivation for the latter algorithm variations is that they are conceptually simpler than random survival forests, and so they can act as a simple baseline method, against which the more sophisticated random survival forest algorithms can be compared.

More precisely, this thesis addresses four objectives, one involving dataset creation and three involving the development of three new types of variants of random forest-based algorithms for survival analysis, as follows:

1. The first objective of this research is to create biomedical datasets with censored data for evaluating machine learning algorithms designed for survival analysis. This research focuses particularly on datasets of age-related diseases, given their strategic importance in biomedical sciences, as discussed earlier. This objective is addressed in Chapter 3.
2. The second objective is to develop new variants of the random forest algorithm based on the imputation of censored target variables. This objective is addressed in Chapter 4. The technical approach followed to achieve this objective was, in essence, to modify only an early step of the standard random

forest algorithm for regression, transforming censored data into uncensored data in that early step, so that the other steps of the standard random forest algorithm can be used without change.

3. The third objective is to develop new variants of the random survival forest algorithm, which is in itself a version of the random forest algorithm that was designed specifically for survival analysis (with censored data). This objective is addressed in Chapter 5. The technical approach followed to achieve this objective was, in essence, to modify the criterion used to select a feature at each tree node and the criterion used to compute the value predicted by each leaf node of the trees in the forest.
4. The fourth objective is to develop new variants of the deep survival forest algorithm, which is in itself an expanded version of the random survival forest algorithm, consisting of stacking multiple random survival forest models into layers, based on some deep learning principles. This objective is addressed in Chapter 6. The technical approach followed to achieve this objective was to replace the base algorithm within the deep survival forest algorithm, which was originally a standard random survival forest algorithm, by a new variant of the random survival forest algorithm proposed to achieve the previous (third) objective.

All these types of random forest-based variant for survival analysis are evaluated by using the same set of 11 real-world datasets of age-related diseases (whose creation is described in Chapter 3).

1.2 Contributions of This Research

At a high level of abstraction, the main contributions of this research are essentially new variants of random forest-based algorithms for survival analysis coping with censored data. The new proposed random forest algorithm variants can be

divided into three groups, leading to three types of contributions, where each contribution type involves modifying a certain type of random forest-based algorithm, as described next.

First, this thesis proposes variants of the standard random forest (RF) algorithms which modify the procedure for creating subsets of training data to be used for learning the decision trees in a RF – which is basically the first step in the execution of a RF algorithm. This modification involves replacing the censored value of a target variable by another target value which is then treated as an uncensored target value, so that all other components of the RF algorithm can be used without modification. The thesis proposes two variants of RF algorithms for this data transformation. The first one is based on randomly generating uncensored target values within certain lower and upper bounds, and the second one is based on using the well-known K-Nearest Neighbour (K-NN) algorithm to estimate the uncensored target value of each censored instance. The proposed variations of the RF algorithm in general outperformed baseline versions of random forest as well as the popular Cox Proportional Hazards (PH) method. The proposed RF algorithm variants are described in Chapter 4.

Second, this thesis proposes a number of variants of the random survival forest (RSF) algorithm (Ishwaran et al., 2008), (Wang and Li, 2017), which is the most popular type of RF algorithm for survival analysis in the area of machine learning. The RSF algorithm learns an ensemble of survival trees (decision trees adapted to survival analysis problems, considering censored data), and has been shown to outperform several methods in survival analysis (Weeraddana et al., 2020). Unlike the classical RF algorithm (Breiman, 2001), the RSF algorithm employs some statistical techniques which enable it to cope with censored data. The proposed RSF variants focus on modifying two major components of the RSF algorithm: the procedure used for selecting the feature to be used for splitting the data at each tree node (the node-splitting criterion), and the procedure used for computing

the value predicted by each leaf node (the leaf-node-prediction criterion). One of the proposed variants of the RSF algorithm outperformed not only the other RSF variants (including the standard one), but also the popular Cox PH method. The proposed RSF algorithm variants are described in Chapter 5.

Third, this thesis proposes several variants of the deep survival forest (DSF) algorithm, which is an extension of the RSF algorithm based on principles of deep learning. In essence, a DSF model consists of multiple RSF models stacked into layers, where each base RSF can be learned using a standard RSF algorithm. The proposed DSF algorithm variants use the same overall process for stacking RSF models into layers as used by a standard DSF algorithm. What is modified in the proposed DSF algorithm variants is the type of RSF algorithm used to learn the base RSF models. More precisely, the proposed DSF algorithm variants use the RSF algorithm variants proposed in Chapter 5 to learn the base RSF models, rather than using the standard RSF algorithm. The best proposed DSF variant outperformed both the standard DSF algorithm and the popular Cox PH method.

As an additional, secondary type of contribution, this research involved the creation of 11 real-world datasets of age-related diseases with censored data, where the target variable represents the time passed until an individual is diagnosed with a given age-related disease. These datasets were created by extracting and processing data mainly from two freely available databases for studying human ageing, namely the ELSA (Clemens et al., 2019) and SHARE (Börsch-Supan et al., 2013) databases, with one dataset being based on haemodialysis data kindly provided by a clinical professor at the University of Kent. In all these sources of data, the original, raw data was far from being directly suitable for machine learning, and had to be extensively processed for the creation of the age-related survival datasets used in the experiments reported in this thesis. The creation of these 11 biomedical datasets is described in Chapter 3.

1.3 Thesis Structure

Chapter 2 contains a background review on supervised machine learning and statistical methods for survival analysis. It will start with a review of two classical types of supervised machine learning methods, namely decision trees and random forests for regression tasks. Next, it will review the main concepts and methods for survival analysis in the area of statistics. This will be followed by a review of the machine learning methods for survival analysis most relevant to this thesis; consisting mainly of the standard random survival forest and the deep survival forest algorithms. Finally, the last part of that chapter will discuss related work on variants of random survival forests proposed in the literature.

Chapter 3 describes in detail the creation, preparation and pre-processing of 11 real-world survival datasets, using data extracted mainly from the freely available ELSA and SHARE databases, and also data from a Haemodialysis database provided by a clinical professor at the University of Kent. Specifically, this chapter focuses on data cleansing and specifying the predictive features and target variables for each dataset, where the target variables represent the time passed until the diagnosis of some age-related diseases (i.e., the "time-to-event" which is the core characteristic of survival analysis problems). These 11 datasets are used in the computational experiments reported in other chapters of this thesis, in order to evaluate the performance the proposed methods for survival analysis.

The next three chapters describe the main contributions of this thesis, which are new variants of random forest-based algorithms for survival analysis, as follows.

Chapter 4 describes new variants of Random Forests which involve the basic idea of imputing the value of the target variable for censored individuals (instances) when generating the data used for learning each tree in a random forest. This chapter describes two methods for this imputation, named the Random Target Imputation Forest (RTIF) and the k-Nearest-Neighbour-Imputation Random

Forest (kNN-RF) algorithms. The final part of this chapter reports results comparing these two new variants of random forest against standard statistical or machine learning methods for survival analysis.

Chapter 5 describes new variants of the Random Survival Forest (RSF) algorithm, which is a state-of-art, very popular technique for survival analysis in the area of machine learning. More specifically, this chapter proposes two novel types of modification of RSFs and experiments with 8 different RSF variants. The first type involves modifications of the node-split criterion of the algorithm, leading to 5 RSF variants; and the second type involves modifications of the leaf-node-prediction criterion, leading to 3 RSF variants.

Chapter 6 describes some modifications of the Deep Survival Forest (DSF) algorithm, which is a very recent machine learning method proposed for survival analysis. The original DSF algorithm essentially consists of stacking random survival forests into a predictive model, partially based on some principles of deep learning. This chapter proposes two new variations of DSF, based on the RSF variants proposed in the previous chapter; and reports the results of experiments comparing those variants against the standard RSF and DSF algorithms. This Chapter also provides an interpretation of the best predictive models created in the experiments, as a contribution to the study of human age-related diseases.

Finally, Chapter 7 summarises the contributions of this research, and presents some conclusions and suggestions for future research.

1.4 Publications Derived from This Research

This research has led to the publication of two papers in peer-reviewed conference proceedings, as follows.

T. Pomsuwan and A.A. Freitas. Adapting random forests to cope with heavily

censored datasets in survival analysis. In: *Proceedings of the 28th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN 2020)*, 697-702. i6doc.com publ. ISBN: 978-2-87587-074-2. URL: https://www.esann.org/proceedings/2020. Conference held in Bruges, Belgium, Oct. 2020.

T. Pomsuwan and A.A. Freitas. New variations of random survival forests and applications to age-related disease data. In: *Proceedings of the 2022 10th IEEE International Conference on Healthcare Informatics (ICHI-2022)*, 1-10. IEEE Computer Society - Conference Publishing Services, 2022. Conference held in Rochester, Minnesota, USA, June 2022.

Chapter 2

Background

This chapter reviews the background on survival analysis and machine learning algorithms relevant to this research. This chapter is organised as follows. Section 2.1 reviews the types of classical machine learning methods that are the focus of this work, namely decision trees and random forests — focusing mainly on the regression task. Section 2.2 reviews concepts and methods for survival analysis, focusing on right-censored data. Section 2.3 reviews traditional statistical approaches widely used to cope with data censorship. Section 2.4 reviews machine learning approaches for survival analysis. Section 5.2 discusses related work, focusing on random survival forests.

2.1 Decision Tree and Random Forest Algorithms for the Regression Task

This section reviews the background on decision tree and random forest algorithms for the regression task of machine learning – with each of these types of algorithms discussed in a separate subsection. We focus on the components of the algorithms that are most relevant to this thesis.

2.1.1 Decision trees for regression

Decision tree induction algorithms are machine learning algorithms for learning a decision tree from data, typically in a top-down fashion (Quinlan, 1993). They are non-parametric supervised learning methods in the statistical sense — i.e., they do not assume that the data has any predefined distribution. They aim to learn a model to predict the value of a nominal class variable (in classification tasks) or a numerical (typically real-valued) target variable (in regression tasks).

Essentially, decision tree algorithms use a recursive learning process which repeatedly performs three main steps. To begin, the algorithm considers all training instances, which are allocated to the root node of the decision tree. Secondly, a feature (f), which best separates the classes based on a given feature selection criterion, is selected to label the current (root) node. Lastly, the set of instances (I) in the current node is partitioned into k mutually exclusive subsets of instances ($I_1, \ldots, I_k$) according to the values of the selected feature f; where k, the number of instance subsets, is determined based on the type of selected feature f, as described below. Then, each of the instance subsets is allocated to a new child node, and the process is recursively repeated for those nodes until a stopping criterion is satisfied (e.g., all instances in the current node belong to the same class).

The next two subsections discuss in more detail two components of decision tree algorithms for regression: the criteria used for selecting the splitting variable in the internal nodes of decision trees and the approach to compute the predicted value of the target variable at leaf nodes of trees. The reason for this focus is that these are the two algorithmic components that are modified to produce some variants of decision tree-based methods that will be proposed later in this thesis.

Split Criteria for Regression Trees

To partition the current set of instances into k subsets, if feature f is nominal (categorical), k is typically the number of values taken by the feature, so that an

instance subset is created for each of the feature values. If feature f is numerical (continuous), typically k is set to 2, so that the algorithm creates two instance subsets, one with the instances satisfying the condition $f \leq thr$ and the other with the instances satisfying the condition $f > thr$, where thr is a threshold automatically determined to maximise separation (minimising the impurity) among the two instance subsets based on their values of the target or class variable.

Many decision tree induction algorithms were developed to cope with classification tasks (Rokach and Maimon, 2014), where the goal is to allocate instances of different classes to different subsets of instances, so that ideally all instances allocated to each new child node belong to the same class.

Regression tasks, however, require different criteria in order to select the best feature to partition the current set of instances, and then allocate instances with similar values of a target variable to the child nodes, since in regression the target variable takes continuous values, rather than nominal class labels like in classification. A common feature selection criterion for regression is the Root-Mean-Square Error (RMSE) (Wang and Witten, 1996), which minimises the variance of the target variable within each child node. The RMSE is defined in Equation 1:

$$\text{RMSE}(I) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\bar{y}_I - y_i)^2} \tag{1}$$

where I is the set of instances in the current node, $\bar{y}$ is the average value of the target variable y over all instances in I, y_i is the value of the target variable in the i-th instance, and n is the number of instances in I. More precisely, the feature selection criterion will be a weighted mean of the variance calculated by Equation 1 over the two child nodes, where the weight for each node is given by the proportion of the set of instances which is allocated to each child node. Hence, more emphasis is given to the minimization of the variance in the child node containing more instances.

This is shown in Equation 2, where $E(I|f)$ denotes the evaluation function for

the current node I, when the data in that node is split based on the values of feature f. In this equation, I_L and I_R represent the numbers of instances allocated to the left and right child nodes, respectively. In other words, $E(I|f)$ computes the weighted mean of the variance across the two child nodes, where the weights are the proportion of instances in each created child node. For example, if one child node has 75% of the instances and the other child node has 25% of the instances of the current tree node, then minimizing the variance in the former child node has a weight of 75%. Note that the expression VAR can be any variance measure and is not limited to the RMSE measure.

$$E(I \mid f) = \frac{I_L}{I} \operatorname{VAR}(I_L) + \frac{I_R}{I} \operatorname{VAR}(I_R) \tag{2}$$

Prediction at Leaf Nodes of Regression Trees

Similarly to classification decision trees, regression decision trees make a prediction at every leaf node. There are two types of decision trees for regression, each using a different type of leaf node, as follows.

In classical regression trees, the predicted value for each leaf node is the mean of the target variable over all instances in that leaf node, as shown in Equation 3, where n is the number of instances in the current leaf node.

$$\hat{\mu_I} = \frac{\sum_{i=1}^{n} y_i}{n} \tag{3}$$

By contrast, model regression trees build an internal (local) linear regression model in each leaf node (Wang and Witten, 1996; Torgo, 1997).

In this thesis, we focus on the first variant, classical regression trees.

2.1.2 Random Forests

Although decision tree induction algorithms' effectiveness in terms of predictive accuracy is generally acceptable, they are not considered the state of the art in terms of predictive accuracy. However, more modern decision tree-based algorithms like random forests (Breiman, 2001; Touw, Bayjanov and al, 2013), tend to obtain higher predictive accuracy in general (Fernández-Delgado et al., 2014), using the power of an ensemble of decision trees to make more robust predictions.

On the other hand, the fact that random forests use an ensemble of (with a large number of) decision trees makes the model much harder to be interpreted by the user than a single decision tree.

Suppose a dataset has M features, and T represents the training set containing n instances, defined as follows:

$$\mathrm{T} = \{(X_l, y_l), (X_2, y_2), \ldots, (X_n, y_n)\}$$

where X_i is a set (or vector) of features $f_{i1}, f_{i2}, \ldots, f_{iM}$ and y_i is the target variable value. Essentially, a random forest algorithm constructs a set of S classification or regression trees, and then aggregates them into a single prediction model.

The learning process is based primarily on two types of ensemble methods – bagging and random subspace (Breiman, 2001). Hence, in order to create S trees in the forest, the algorithm starts by creating S data samples, each of them created by randomly sampling instances from T with replacement, as shown in Figure 1. The random sampling process is repeated n times for each created dataset, so that their sizes are the same as the original dataset. This results in $T_1, T_2, \ldots, T_S$ training sets, where each T_i is called a bootstrap (training) dataset. Note that due to using a "with-replacement" sampling approach, T_i can have duplicate instances or be missing a subset of instances from the original training set T. This process

is known as bootstrapping. In short, the bagging method is the process of taking S bootstrap training sets and then aggregating the models (trees) learned from each T_i.

In order to build each tree, at each node, the random forest algorithm first randomly samples m features from the original feature set X, where m is usually calculated as either $\sqrt{M}$ or $\ln(M+1)$ (a user-made decision), where the resulting fraction (if any) is rounded (Breiman, 2001). Technically, the m features are randomly sampled from the feature set X without replacement. This is called the random subspace method. Then, the m features are considered candidate features for selection, and the algorithm selects the best of those m features (based on a given feature-selection criterion like minimizing the RMSE) to label the current tree node and to split the set of instances in that node, as usual in regression trees.

The last step of the bagging method is making a prediction. Since the random forest algorithm constructs S prediction models (trees), $K_1, K_2, \ldots, K_S$, feeding an instance into these models will result in S prediction values (outputs), as shown in Figure 1. In order to report a single prediction value, a random forest model combines the predictions of all trees by using an aggregation procedure such as average (for regression) or voting (for classification).

2.2 Concepts for Survival Analysis, Focusing on Right-censored Data

Survival analysis consists of a set of statistical methods concerned with analysing the time until the occurrence of an event of interest (Kleinbaum and Klein, 2012). The occurrence of such event is often referred to as a "failure" in the literature, but the event of interest can be a "failure" or a "success", or any other type of event of interest. To some extent, survival analysis is similar to linear regression analysis

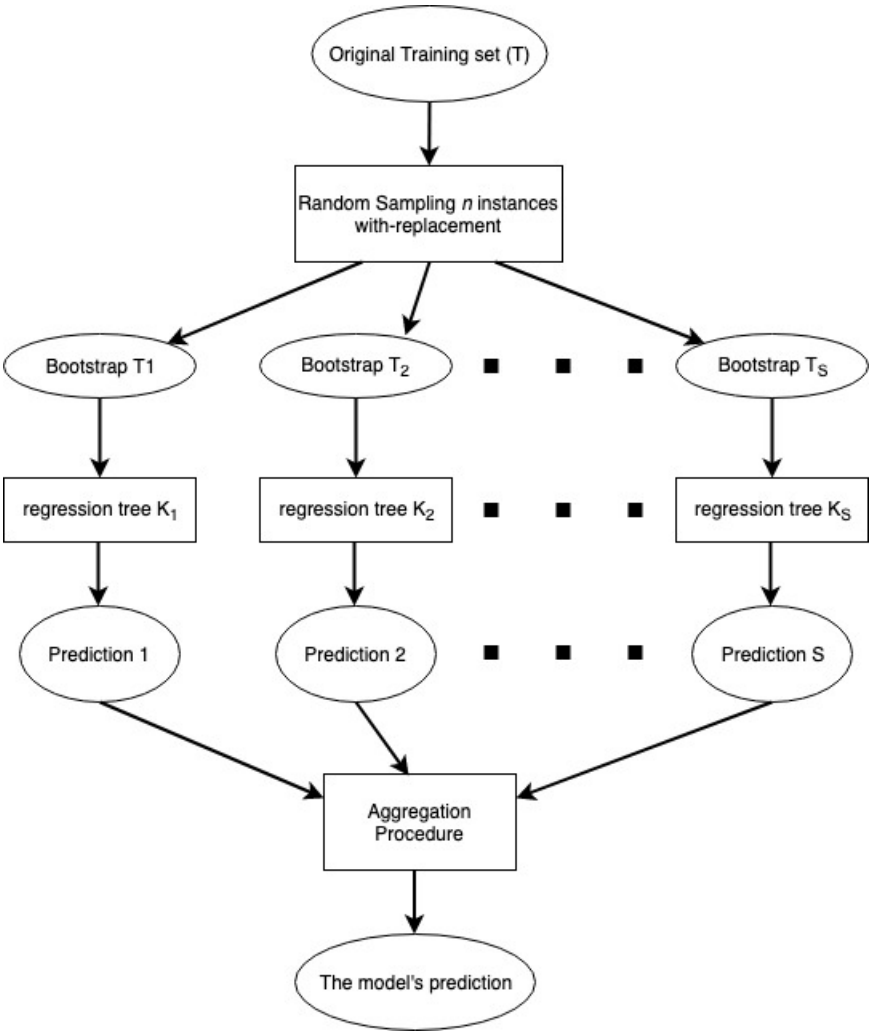


Figure 1: : The basic idea of Random Forest algorithm, combining the predictions of many decision trees, each learner from a bootstrap sample of the data

where the prediction of the value of a target (response) variable is computed from a set of features, since the time to an event occurrence can be considered a numerical target variable to be predicted. The important difference, however, is the presence of data censoring, which cannot be effectively handled by traditional linear regression methods.

2.2.1 Censoring

Censoring occurs when observed instances have some information available for estimating the survival time but the information is incomplete. Figure 2 shows different situations where censoring occurs in survival analysis. In this Figure, the occurrence of an event of interest is marked with a red cross (X). Looking first at instance (subject) A, it is labelled as "uncensored" since its survival time is precisely known - i.e., subject A was followed since the start of the study, and her/his failure time was observed. Apart from that, the other instances are censored.

There are two main types of censoring, namely right censoring and left censoring (Kleinbaum and Klein, 2012). The first and most common one is right censoring, where no event of interest occurred for a subject during the period in which he/she was observed in the study. There are essentially two reasons for the occurrence of right censoring. First, the patient was observed until the end of the study, and no event of interest occurred until that time (instance B in Fig.2). Second, the subject dropped out of the study or was lost to follow up before its end and no event of interest occurred before the dropout (instance C in Fig.2). Note that, in both cases of right censoring, the last observed time for a subject is a lower bound for the unknown event occurrence time. Left censoring occurs when a subject experiences the event of interest before the start of the study, resulting in their entry into the study after the occurrence of the event (referred to as instance D in Fig.2). Therefore, any information available at the beginning

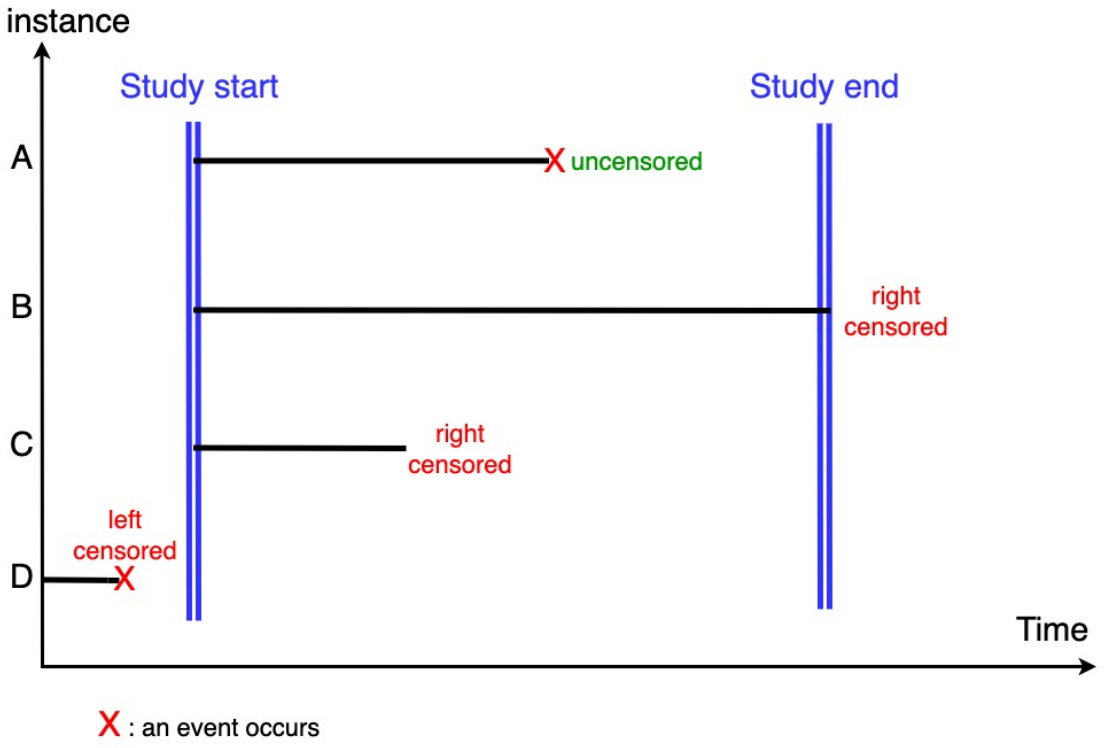


Figure 2: Diagrammatic representation of uncensored and censored instances in survival analysis. "X" denotes an occurrence of the event of interest.

of the study holds no predictive power for the left-censored individuals.

We focus on right-censoring (as opposed to left-censoring), which is generally encountered in medical research (Kleinbaum and Klein, 2012), such as the prediction of cancer survival (Molinaro, Dudoit and Van Der Laan, 2004). More specifically, it is also a common type of censoring in the datasets used in our experiments, described later; since a patient is often either lost to follow up before the end of the study or does not experience the event during the study.

2.2.2 Kaplan-Meier estimates of survival time

Regarding nonparametric methods, the Kaplan-Meier method (or estimator) is a popular one for estimating the survival function from lifetime data (Kaplan and Meier, 1958). An important advantage of the Kaplan-Meier method is that it

Table 1: An example of the Kaplan Meier estimates for a survival function

Time (t)	Risk set (n_t)	Failed (m_t)	Censored (c_t)	$S(t)$
0	1000	0	0	1
2	1000	90	10	$1 \times \left(1 - \frac{90}{1000}\right) = 0.910$
3	900	300	100	$0.910 \times \left(1 - \frac{300}{900}\right) = 0.607$
7	500	250	50	$0.607 \times \left(1 - \frac{250}{500}\right) = 0.303$
9	200	50	50	$0.303 \times \left(1 - \frac{50}{200}\right) = 0.2275$
10	100	10	90	$0.227 \times \left(1 - \frac{10}{100}\right) = 0.2048$

can take into account some types of censored data, particularly right-censoring (Vock, Wolfson et al., 2016). In other words, the method considers both types of information about a subject's survival time, namely the time to the event and the event status, when estimating the survival function $S(t)$, given by Equation 4, where n_j is the number of subjects in the risk set at the failure time j — i.e., the set containing subjects who have survived at least to time j; and m_j is the number of failures at time j. Essentially, this equation measures the product of fractions representing the proportions of subjects alive across the time points.

$$S(t) = \prod_{j=0}^{t} \left(1 - \frac{m_j}{n_j}\right) \tag{4}$$

Table 1 shows an example (adapted from (Kleinbaum and Klein 2012)) of how the Kaplan-Meier method estimates several values of a survival function from a dataset of 1000 subjects, with censored data included. There are five columns, where t is the observed failure time; n_t is the number of subjects in the risk set at time t, i.e., the set containing subjects who have survived at least to time t; m_t is the number of failures at time t; c_t is the number of subjects who were censored at time t and $S(t)$ is the value of the survival function estimated at time t. Note

that there are five unique failure times by which the table is ordered. To make the example more concrete, the time (t) is assumed to be measured in years.

To begin, the probability of surviving past zero is unity (all 1000 subjects in the dataset), as it will always be for any dataset. Next, the probability of surviving past the first ordered failure time of 2 years is given by 910/1000 (or 0.910) because 90 subjects failed at the 2-year mark. In addition, 10 subjects were censored at the 2-year mark. Hence, 900 subjects from the original 1000 remained as survivors past 2 years. Subsequently, the rest of the survival estimates is calculated by multiplying the estimate of $S(t)$ for the preceding failure time by a fraction. For example, the fraction is 600/900 for surviving beyond 3 years since 900 subjects remained up to this year and 300 of these failed to survive past it. Hence, the probability of surviving past 3 years is 0.607. Similarly, the next probability concerns subjects surviving past 7 years, which is 0.303 where the fraction is 250/500: 500 remaining up and 250 of them surviving past this time point; and so on.

2.2.3 Nelson-Aalen Estimates of Cumulative Hazard Function

The Cumulative Hazard Function (CHF) defines the ratio of occurrence of the event of interest given that subjects survive past a certain amount of time. We review the CHF here since it is an important component of Random Survival Forests, one of the target types of algorithms in this work, as discussed later. The CHF is another measure of the population's survival distribution against time. Whilst the well-known Kaplan-Meier estimator analyses the survival distribution of the population through their survival function, the Nelson-Aalen estimator is its counterpart, analysing the survival distribution of the population through their CHF (Nelson, 1972).

Note that if a survival analysis focuses on frequencies of occurrences of the

Table 2: An example of the Nelson-Aalen estimates for a CHF

Time (t)	Risk set (n_t)	Failed (m_t)	Censored (c_t)	$H(t)$
0	1000	0	0	0
2	1000	90	10	$\frac{90}{1000} = 0.09$
3	900	300	100	$0.09 + \frac{300}{900} = 0.42$
7	500	250	50	$0.42 + \frac{250}{500} = 0.92$
9	200	50	50	$0.92 + \frac{50}{200} = 1.17$
10	100	10	90	$1.17 + \frac{10}{100} = 1.27$

event, then the Nelson-Aalen estimator should be preferred, since Cumulative Hazard estimates can be used to compute the expected number of events (Ishwaran et al., 2008). By contrast, if the study is interested in the survival time of the subjects, survival probability estimates produced by the Kaplan-Meier method are more appropriate, since survival functions have a natural interpretation regarding survival times - e.g., calculating the mean or median survival times.

The Nelson-Aalen estimate of the Cumulative Hazard Function for the event of interest at a time point t, denoted by $H(t)$, is given by Equation (23) (Nelson, 1972).

$$H(t) = \sum_{j=0}^{t} \left(\frac{m_j}{n_j} \right) \tag{5}$$

Where m_j is the number of failures at time j and n_j is the number of subjects in the risk set at time j, i.e., the set containing subjects who have survived at least to time j.

Table 2 shows an example of how the Nelson-Aalen method estimates several values of a Cumulative Hazard rate from an example hypothetical dataset of 1000 subjects, with censored data included. Table 2 has five columns, where

- t is the observed failure time;
- n_t is the number of subjects in the risk set at time t, i.e., the set containing subjects who have survived at least to time t; for each row i where i in $[2 \ldots 6]$, corresponding to t in {2,3,7,9,10}, $n_i = n_{i-1} - m_{i-1} - c_{i-1}$;
- m_t is the number of subjects who "failed" at time t;
- c_t is the number of subjects who were censored in the time interval starting with time t up to but excluding the next failure time.
- $H(t)$ is the ratio estimated by the Cumulative Hazard Function at time t. Note that there are five unique failure times by which the table is ordered.

Hence, in the example of Table 2, subjects who survived until at least time $t = 10$ have a CHF of 1.27.

2.2.4 The Log-Rank Statistic

In order to compare two (or more) groups of subjects through their estimated survival functions (curves), several statistical tests are available. If the probability distribution of the function fits a certain known distribution, then the associated parametric test method is a natural choice. Otherwise, a non-parametric test is a natural choice.

The Log-rank test, which is basically a type of chi-square test, is widely used in practice since it is a nonparametric test designed for comparing the survival distributions between two groups — i.e., it evaluates the difference in survival times between two groups of subjects. In particular, it compares the hazard functions at each observed event time. The Log-rank test is commonly used together with the Kaplan-Meier method since both are nonparametric methods. The Log-rank statistic is given by Equation (6):

$$\text{Log–rank statistics } = \frac{(O_i - E_i)^2}{\text{Var}\,(O_i - E_i)} \tag{6}$$

$$O_i - E_i = \sum_{j=1}^{k} (m_{ij} - e_{ij}) \tag{7}$$

$$e_{ij} = \left(\frac{n_{ij}}{n_{1j} + n_{2j}} \right) \times (m_{1j} + m_{2j}) \tag{8}$$

$$\mathrm{Var}\,(O_i - E_i) = \sum_{j=1}^{k} \frac{n_{1j} n_{2j} \,(m_{1j} + m_{2j})\,(n_{1j} + n_{2j} - m_{1j} - m_{2j})}{(n_{1j} + n_{2j})^2\,(n_{1j} + n_{2j} - 1)} \tag{9}$$

In Equation (6), O_i is the sum of the number of observed failures in group i across all failure times and E_i is the expected value of the sum of the number of failures in group i across all failure times. To compute the Log-rank statistic, we need to calculate the term $O_i - E_i$, which is a measure of the overall differences of the survival or hazard function (curve) over all k failure times and is given by Equation (7), where e_{ij} is the expected number of failures for group i at the failure time j, as shown in Equation (8). *Var($O_i - E_i$)* is the estimated variance, which involves the number of subjects in the risk set in each group (n_{ij}) and the number of failures in each group (m_{ij}) at time j. k is the number of distinct times of observed failures. The summation is over all distinct failure times. Note that when comparing any pair of survival functions, this calculation will be done for just one of the two groups since the absolute difference is the same for the two groups.

2.2.5 Metrics for Survival Analysis

In survival analysis, several evaluation metrics have been developed to assess the predictive accuracy of a predictive model. Among these metrics, Harrell's C-index, Uno C-index, and the integrated Brier score are widely used to measure the performance of survival models. This subsection will elaborate these three evaluation metrics, especially the Harrell's C-index as being focused in this thesis.

Harrell's C-index

Harrell's C-index, a.k.a. Concordance index (C-index), is a predictive accuracy measure of the learned model's performance. The C-index can be interpreted as the probability of correctly ordering the predicted survival values for a randomly chosen pair of subjects whose actual survival times are different. As described in (Harrell, Lee and Mark, 1996), the C-index can be adapted for censored data by considering the concordance of actual survival times versus predicted survival times among pairs of subjects whose survival outcomes can be ordered with respect to their survival times, i.e., among pairs where both subjects are observed to experience an event, or one subject is observed to experience an event before the other subject is censored. Note that in this latter case we know that the censored subject survived longer than the subject whose event was observed, so this pair of subjects can be ordered, even though we do now know the precise survival time for the censored subject.

Specifically, Table 3 enumerates four different types of subject pairs and determines the "usable" pairs for computing the C-index. Let T_1 and T_2 be the value of the target variable (survival time) for subjects 1 and 2. Note that for simplicity the second and third rows of this table refer only to the case where subject 1 is censored and subject 2 is uncensored, but the subject order is arbitrary. Hence these two rows also cover the dual cases where subject 1 is uncensored and subject 2 is censored, which would involve reversing the inequality operators in the second column of the table.

As shown in Equation 10, C-index is a ratio of the count of useful and concordant pairs of subjects (the numerator) over the number of usable pairs of subjects (the denominator). The term *Agreed_order* is defined in Equation 11 where $\hat{T}_i$, $\hat{T}_j$ are the estimated (predicted) survival time for subject i and j. Note that when the predicted survival times are identical for a subject pair, $\frac{1}{2}$ rather than 1 is added to the count of concordant pairs in the numerator of Equation 10. In this

Table 3: determining "usable" subject pairs for computing the C-index (concordance index)

Subject 1	**Subject 2**	**Usable?**	**Remark**
Uncensored	Uncensored	Yes, unless $T_1 = T_2$	Usable pair, unless the survival times for Subjects 1 and 2 are equal
Censored at time T_1	Uncensored, with survival time $T_2 \leq T_1$	Yes	Subject 1 is known to have survived at least as long as subject 2
Censored at time T_1	Uncensored, with survival time $T_2 > T_1$	No	Subject 1 was not followed long enough to know whether or not he/she will survive longer than Subject 2
Censored at time T_1	Censored at time T_2	No	Unknown which subject survived longer, since both subjects have unknown survival times

case, one is still added to the denominator of Equation 10 (such subject pair is still considered usable). A subject pair is unusable if both subjects are uncensored and experienced the event at the same (known) time.

$$\text{C-index} = \frac{|\{(i,j)|\ \text{Usable}\ (i,j)\ \text{AND Agreed\_order}\ (i,j)\}|}{|\{(i,j)|\text{Usable}(i,j)\}|} \tag{10}$$

$$\text{Agreed\_order}(i,j) = \begin{cases} \textbf{true}, \text{ if } \widehat{T}_i > \widehat{T}_j \text{ and } T_i > T_j \\ \textbf{true}, \text{ if } \widehat{T}_j > \widehat{T}_i \text{ and } T_j > T_i \\ \textbf{false}, \text{ otherwise} \end{cases} \tag{11}$$

This thesis uses Harrell's C-index as the measure of predictive performance for all methods evaluated in the experiments.

Uno's C-index

The Uno's C-index, introduced by Uno et al. (2011), addresses a limitation of the Harrell's C-index, which does not directly address the uncertainty introduced by censoring, i.e., it does not use instance weights to account for censored data. That is, Uno's method incorporates Inverse Probability of Censoring (IPC) weights to assign positive weights to uncensored instances rather than just the rank ordering as Harrell's C-index. This can be meaningful to upweight the contributions of censored subjects based on their predicted risks — i.e., addressing the uncertainty

introduced by censoring. Equation 12 is the Uno C-index:

$$\text{Uno C-index } = \frac{\sum_i [w_i \cdot I(T_i > T_j) \cdot I(\widehat{T}_i > \widehat{T}_j)]}{\sum_i [w_i \cdot I(T_i > T_j)]} \tag{12}$$

where w_i denotes the IPC weight for the i-th subject (instance). The weights can be calculated using Equation =18, which will be discussed in Subsection 2.3.2, which is specifically about IPC weights. I is an indicator function that equals 1 if the condition is true.

Therefore, both Harrell's C-index and Uno's C-index account for censoring in survival analysis, but there is a difference in their specific calculations and weighting schemes. While Uno's C-index incorporates IPC weights, Harrell's C-index considers the orderings of pairs involving censored observations without explicitly using weights.

Brier Score and Integrated Brier Score

The Brier score at time t is defined as the mean squared difference between the observed status and the predicted survival probability. Mathematically, for a set of n individuals, it is defined as:

$$BS(t) = \frac{1}{n} \sum_{i=1}^{n} \left(\widehat{S}(t|\mathbf{x}_i) - \delta_i(t) \right)^2, \tag{13}$$

where $\widehat{S}(t|\mathbf{x}_i)$ is the estimated survival probability of individual i at time t given covariates $\mathbf{x}_i$, and $\delta_i(t)$ is the observed status of individual i at time t (1 if the individual has died, 0 otherwise).

The Integrated Brier Score (IBS) is a metric commonly used to assess the accuracy of survival predictions. It measures the mean squared difference between predicted survival probabilities and observed survival time over a specified time interval. The IBS ranges from 0 (perfect calibration) to 1 (worst calibration).

The Integrated Brier Score is the time-average of these Brier scores up to a

certain time point T, defined as:

$$IBS(T) = \frac{1}{T}\int_0^T BS(t)dt. \tag{14}$$

The Brier score measures the mean squared difference between the predicted probability of survival and the actual outcome, so a lower Brier score corresponds to more accurate predictions. The IBS extends this idea by averaging these scores over time, providing a single metric that summarises the model's predictive performance up to a certain time point.

It should be noted that the Brier score and by extension the IBS only assess the calibration of the model, or the accuracy of the predicted probabilities, and not the discrimination, which is the model's ability to distinguish between individuals who experience an event and those who do not.

Moreover, the IBS can be used if and only if the the machine learning/statistical model is able to prediction survival functions, which is not a built-in feature for all models. Specifically, methods such as the Cox Proportional Hazards (Cox PH) model, commonly used in survival analysis, do not naturally estimate survival function. The Cox PH model is essentially a regression model which, rather than estimating the survival function directly, estimates the effect of explanatory variables on survival time.

2.3 Traditional Statistical methods for survival analysis

This section provides an overview of traditional statistical approaches to analyse survival data involving censored data.

The information about the survival time of a subject is composed of two parts: one is the time to the occurrence of an event (when this is known) and the other

is the event status, which records if the event of interest occurred or not. In order to analyse survival data based on the target variable, although one could use standard regression methods, such methods are in general inadequate for several reasons. First of all, standard regression methods were not designed to cope with censored data, which are very common in survival analysis. This is because usually we are not able to observe an occurrence of the event of interest for every individual, e.g., some individuals were lost to follow-up. Second, survival data in general follow a skewed distribution, whereas standard regression methods assume a normal distribution. Data violating this assumption may lead to a drop in the performance of standard regression methods. Thirdly, most regression methods ignore the fact that survival times cannot take negative values. Last but not least, although regression methods can be used to estimate the time to event, some survival studies are interested in the risk of experiencing the event conditional on surviving until a certain point (Hazard function) given some attributes (factors) of interest, such as age, gender, treatment, etc (Kleinbaum and Klein, 2012).

Two types of probability functions are used to describe survival data: the survival function and the hazard function. They are key and opposite concepts in survival analysis for describing the distribution of event times. The survival function is the probability of surviving (or not experiencing the event of interest) from the time at the start of the study to a specified future time, while the hazard function is the failure rate for the occurrence of an event at a certain time given that an individual has survived up to that time. In other words, in contrast to the survivor function, which focuses on not having an event, the hazard function focuses on the event occurring.

Therefore, in order to analyse survival data, statistical methods typically estimate one of these two functions. There are several statistical methods available to estimate the survival function and the hazard function by using different types of approaches, namely parametric, nonparametric and semiparametric methods.

Parametric methods make an assumption about the distribution of the survival times following a certain known probability distribution. For example, the exponential distribution is one of the common assumptions for survival functions, whereas the Weibull distribution fits well into hazard functions (Kleinbaum and Klein, 2012).

In the next two subsections, we briefly review two traditional statistical methods for survival analysis which have been used as baseline methods in some experiments reported later in this thesis: first, the very popular Cox's Proportional Hazard Regression method; and second, the Inverse Probability of Censoring method.

2.3.1 Cox's Proportional Hazard Regression

Cox's Proportional Hazard Regression (Cox, 1972) is a semiparametric method – making fewer assumptions than parametric methods such as the Weibull and the Exponential hazard models. Specifically, while the Cox method does not assume any particular distribution for the estimated survival time function, it assumes a type of distribution for the features (attributes, or covariates) – i.e., the requirement of the proportional hazard assumption must be met. This refers to a hazard ratio (between two subjects or two groups of subjects) being constant over time or, ideally, the feature values must not change once measured. Note that a few common health-related features such as a person's age and weight can be allowed even though their values would not stabilize (Kleinbaum and Klein, 2012). This type of exception is made due to the very little change in their effect on the hazard function over time.

There are two terms involved in Cox's Proportional Hazard Regression formula. Each term takes one of the two parameters, time and features, into the calculation separately, as shown in Equation 15:

$$h(t, X) = h_0(t)e^{\Sigma\beta X} \tag{15}$$

This hazard function (or hazard model) expresses the hazard rate at time t for an individual with a given specification of a set of features (covariates) denoted by X. As shown in Equation 15, the Cox model is the product of two parts. The first one, $h_0(t)$, is called the baseline hazard function, which involves the time t. The second part, $e^{\Sigma\beta X}$, is the exponential expression which involves the set of features X, where β is the set of coefficients measuring the impact (weight) of the features. Note that β and X represent a set (or vector) of coefficients and features, so that βX is a shorthand notation for $\beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_d X_d$ where d is the number of features.

Therefore, the Cox regression model estimates hazard functions considering predictive features, unlike the Kaplan-Meier method. Hence, it could be used directly to predict the hazard function and/or survival function of a subject with the feature set X. However, this would require some assumptions about the type of baseline hazard function. To avoid this assumption in practice, the Cox model is generally used to compute a *Hazard Ratio* for two subjects or groups of subjects with different sets of features.

From Equation 16, the *Hazard Ratio* equation can be derived as follows:

$$\text{Hazard Ratio } = \frac{h\left(t, X_1\right)}{h\left(t, X_2\right)} = \frac{h_0(t) e^{\sum \beta X_1}}{h_0(t) e^{\sum \beta X_2}} = e^{\sum \beta (X_1 - X_2)} \tag{16}$$

Since the baseline hazard part is cancelled out, the Cox regression model is a proportional hazard model where the hazard curves are proportional and cannot cross. As an example adapted from (Kleinbaum and Klein, 2012), Figure 3 illustrates two different scenarios for the hazard rates of two different groups of patients, G_1 and G_2, and the Hazard Ratio (HR) between groups (the green line). Part (a) of this figure illustrates that despite the hazard rates of G_1 and G_2 changing over time, both of them rise proportionally and the *Hazard Ratio* (the green line) almost levels off (or changes very little): this meets the proportional hazard assumption. Accordingly, a *Hazard Ratio* of 2.0 indicates the risk of patients in

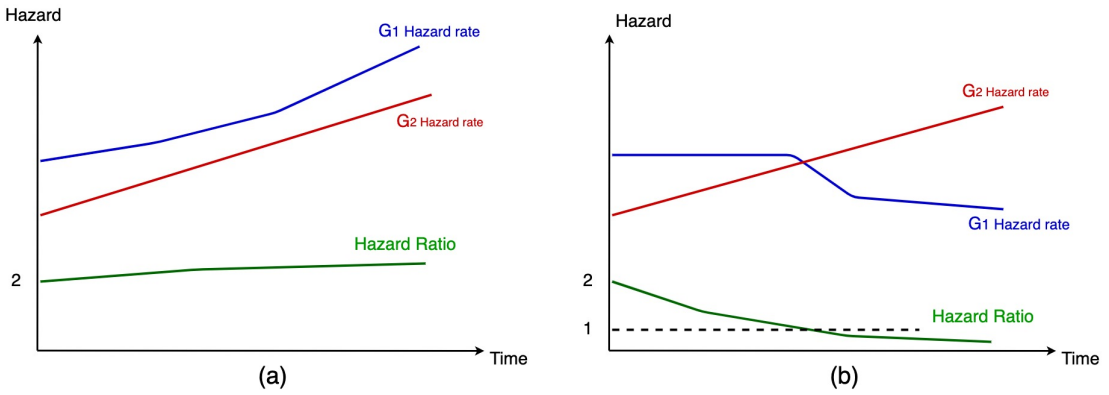


Figure 3: An example of how the *Hazard Ratio* between two groups may remain constant (a) or change over time (b), depending on the stratified feature

G_1 is twice of those in G_2. However, in actual fact said assumption is unrealistic to be satisfied.

Part (b) of the figure shows how the hazard rates change over time for two different groups of individuals. Unlike part (a), for part (b) we suppose G_1 denotes the group with a substantial reduction of the hazard rate with time, while the hazard rate of group G_2 monotonically increases with time. G_1 could represent a group of patients who undergoes serious surgery – e.g., removing a cancerous tumour. Such process normally leads to a high risk for a period of time after the surgery but the individuals' condition improves (i.e. the hazard ratio is reduced) once they recover. G_2, however, could represent another group of patients suffering from the same long-term disease but not receiving any kind of treatment. In other words, the hazard rate of G_2 describes the likelihood of how the disease develops over time in a natural way, without medical intervention. Therefore, looking at the *Hazard Ratio* (green line) dropping from 2.0 to lower than 1.0 in part (b) of the figure, the effect is reversed when patients in group G_2 have a much higher risk. This greatly violates the proportional hazard assumption, since the *Hazard Ratio* is greater than 1 at any time point before the cross, whereas being smaller than 1 afterwards. Hence, a HR of 2.0, estimated by the Cox model ignoring the effect of time, is totally misleading.

Furthermore, the method assumes that the effects of the features upon survival are constant over time, due to Equation 16 being independent of the time t (Cox, 1972). This is often an unrealistic assumption, particularly when predicting age-related diseases (the focus of this research) since the effect of several features on a subject's probability of survival often depends on the time the feature is measured along the life of that individual (i.e., their age).

Another strong assumption is that each predictive feature has a linear relationship with the survival time, since the term $\Sigma\beta X$ has a linear regression form.

2.3.2 Inverse Probability of Censoring

Based on sampling techniques, (Robins and Rotnitzky, 1992) introduced the concept of Inverse Probability of Censoring (IPC) weights, where positive weights are assigned to uncensored instances while the weights of the censored ones are set to 0. As the name suggests, a subject who has a long survival time is assigned a large IPC weight value, which is inversely proportional to its probability of being censored.

The probability of censoring is estimated by the Kaplan-Meier method. Note, however, that this is a different use of the Kaplan-Meier method. Instead of estimating a survival function as in Section 2.2.2, the method is used to estimate a censoring function, i.e., the probability that the censored time is greater than t, denoted $G(t)$, as shown in Equation 17, where n_j is the number of subjects in the risk set, i.e., the set containing subjects who have survived at least to time j, and c_j is the number of subjects who were censored at time j.

$$G(t) = \prod_{j=0}^{t} \left(1 - \frac{c_j}{n_j}\right) \tag{17}$$

After $G(t)$ – the probability of censoring function – has been computed, the IPC weight for instance i (w_i) is calculated as shown in Equation 18

$$w_i = \begin{cases} \frac{1}{G(t_i)}, \text{ if } i \text{ is uncensored} \\ 0, \text{ Otherwise} \end{cases} \tag{18}$$

Where t_i is the survival time observed for the i-th subject (recall that this is known because this part of Equation 18 applies to uncensored subjects), and the value of $G(t_i)$ is given by Equation 17.

2.4 Machine Learning Approaches for Survival Analysis

This section reviews the main machine learning concepts and methods for analysing survival data with censored target variables. It consists of four sub-sections. First, it contrasts the two basic approaches of predicting the numerical value of the target survival variable (a regression problem) and binarising the target variable and treating its prediction as a classification problem. Second, it reviews the survival ensemble method, which is based on pre-processing the data in a way that takes into account the censorship of the target variable. Third, it reviews random survival forests, which is a popular machine learning method for survival analysis of censored data, and is used as the basis for several variants of random survival forests proposed in Chapter 5. Fourth, it reviews deep survival forests, which is an extension of random survival forests inspired by deep learning concepts, and is also the basis for some variants of this method proposed in Chapter 6.

2.4.1 Binary Survival Classification vs Prediction of the Numerical Survival Outcome (Regression)

Supervised machine learning studies focus on predicting the survival time of new instances by building a model from the training data. Broadly speaking, there

are two types of survival time prediction problems. The first one is a binary classification problem, where a classification model is built in order to predict whether or not an event of interest will occur within a certain period based on a pre-defined survival time threshold. This approach is used in several studies (Štajduhar, Dalbelo-Bašić and Bogunović, 2009; Delen, Walker and Kadam, 2005; Urquhart et al., 2015; Chang et al., 2013; Chen, Ke and Chiu, 2014; Panahiazar et al., 2015; Roadknight et al., 2015; Zacharaki, Morita et al., 2012) in which the survival time thresholds were either varied or set to create a dataset with a balanced class distribution. Note that normally varying the thresholds means repeating experiments with different values of the survival time threshold, rather than modifying the algorithms to try to find the best threshold. In this approach, each instance (subject) is assigned a binary survival variable as a class variable, taking the value "survival" or "non-survival". Survival means the subject did not experience an event before the time threshold and non-survival means otherwise. The second and more challenging type of problem is the prediction of continuous survival time, which is the focus of this work.

In general, conventional machine learning methods cannot effectively handle censored data because of two problems. First, missing values for features during the part(s) of the study when the subject was not observed (left censoring and one type of right censoring) introduce uncertainty into feature values. Second, right censoring introduces uncertainty into the value of the target variable to be predicted. Recall that, when an instance is right censored, the last observed time for a subject is a lower bound for the event occurrence time for that subject, i.e., the exact event time for that subject is unknown. Note that this is true when the prediction problem is cast as regression, but not necessarily true when it is cast as binary classification, since in the latter case the "lower bound" for a subject may be greater than the time threshold, in which case we know the class for that subject is "survival" anyway.

Although censored data can be a serious problem for regression or classification algorithms, if the number of censored instances is relatively low compared to the dataset's size, we can handle this problem by removing the censored instances, as performed in some studies (Blanco et al., 2005; Delen, Walker and Kadam, 2005; Zacharaki, Morita et al., 2012). Note, however, that the proportion of the instances removed from the dataset (for being censored) was not discussed in these papers. After removal, they applied traditional classification algorithms to the dataset. Alternatively, (Štajduhar, Dalbelo-Bašić and Bogunović, 2009) treated censored instances as event-free, and then applied a conventional classification algorithm. The authors concluded that their simple technique could handle a relatively small level of censoring (up to 20%) without affecting the predictive accuracy.

However, when the number of censored instances grows large, the drawback of the above techniques cannot be ignored. Essentially, not only is the number of instances available for learning substantially reduced when censored instances are removed, but also both techniques would introduce some biases in the training set for machine learning algorithms. The censored instances might contain information relevant to predicting the survival time, so that not including them in the learning process may lead to the model failing to capture the underlying trend in the data. Therefore, several studies have been conducted to overcome this limitation (Kourou et al., 2015).

2.4.2 Survival Ensemble

Instead of completely removing censored instances from experiments, a few studies attempted to oversample the uncensored instances such that they represent the removed (censored) instances. As explained earlier in this chapter, (Robins and Rotnitzky, 1992) introduced Inverse Probability of Censoring (IPC) weights where positive weights are assigned to uncensored instances while the weights of the

censored ones are 0. See Equations 17 and 18 for more details.

One of the studies making use of the IPC weight technique to cope with censorship is (Hothorn et al., 2006). This work predicts the log of the continuous survival time (explained next) using a modified random forest algorithm. A random forest is constructed where each tree was derived from bootstrap data where each instance is sampled with a probability based on its IPC weights (recall that censored instances had IPC weights of 0). More precisely, each instance whose censoring probability can be drawn from the computed distribution is assigned a weight, which is inversely proportional to its censoring probability. Hence, censored instances are not used in the tree-building process.

Another study is (Vock, Wolfson et al., 2016), which assigns IPC weights to the training set, and then applied classical classification algorithms. The applied methods included Bayesian networks, k nearest neighbours, decision trees, and generalized additive models, since these methods have a natural way to cope with weighted instances.

2.4.3 Random Survival Forests

There are a number of studies that attempt to directly modify machine learning algorithms to enable their learning capability from censoring data in survival analysis — e.g., Bayesian Network (Ibrahim et al., 2001), Artificial Neural Networks (Faraggi and Simon, 1995) and Support Vector Machine (Pölsterl, Navab and Katouzian, 2015) to name a few. However, in terms of popularity of use, Random Forests can consistently be seen in many practical uses in both industries and academia.

As discussed earlier, the Random Forest algorithm is an ensemble learning method consisting of decision-tree learning algorithms as base learners. One of the convincing reasons behind its success is the fact that ensemble learning usually reduces prediction errors by reducing the effect of the variance on the final

error (Zhou, 2012; Friedman, 1997). To do so, prediction outputs are aggregated across the base learners, by means of average values, for example. Thus, random prediction errors that would possibly be made by a stand-alone model (without using an ensemble) can be avoided since they are more likely to be overwhelmed by the predictions of the majority of trees in the forest. This is, of course, conditional on the assumptions that not only are the base learners very different from each other (so that their prediction errors are uncorrelated), but also they must perform better than a random prediction procedure (Breiman, 1996; Zhou, 2012).

Hence, in the remainder of this subsection, we review the Random Survival Forest algorithm, which consists of a major adaptation of the Random Forest algorithm for coping with censored data in survival analysis tasks. As for other well-known machine learning techniques modified for survival tasks, the reader is referred to (Wang, Li and Reddy, 2019).

Motivation

As mentioned earlier for (Hothorn et al., 2006), the log transformation approach has also been used to transform the survival time (target variable) in some other studies such as (Wang and Dinse, 2011; Pölsterl, Navab and Katouzian, 2015). This approach transforms highly skewed distributions, which are normally the case for survival time (Clark et al., 2003), into less skewed ones. Therefore, it can be helpful for some statistics and machine learning methods that assume the survival times to be normally distributed when analysing survival data. Nevertheless, as pointed out in (Pölsterl, Navab and Katouzian, 2015), even after log transformation, the distribution of the survival time is often still far from the normal distribution, which violates an assumption of ordinary least squares/classical linear regression.

In order to avoid using the log transformation of the survival time, one approach is to use a survival analysis method which does not make the assumption

of normal distribution for the target variable. In this context, the most popular method for survival analysis in machine learning is known as Random Survival Forest (RSF), proposed by (Ishwaran et al., 2008).

The RSF algorithm has been developed in order to produce a specific type of predicted outcome at the leaf nodes which was designed to cope with censored data. In other words, this replaces the normal prediction of target values at leaf nodes in random forests for regression (which cannot cope with censored data). The authors named said prediction "Ensemble Mortality", which represents the expected rate of deaths or the expected cumulative hazard rate. That is, instead of focusing on the survival times of the instances, a survival tree uses the Nelson-Aalen method for each leaf node to estimate the Cumulative Hazard Function (CHF).

In addition, RSF learns an ensemble of "survival trees", as opposed to standard regression trees. It uses the Log-rank test as the node-splitting criterion; which is a non-parametric test specifically for survival analysis rather than regression, as discussed in more detail below.

Methodology

We assume that the reader is familiar with the well-known standard Random Forest algorithm for regression (Breiman, 2001) – an overview of which was presented in Section 2.1.2. Hence, we focus here on describing mainly the characteristics of the Random Survival Forest (RSF) algorithm that makes it specifically adapted for survival analysis with censored data (Ishwaran et al., 2008), rather than standard regression.

RSF is a powerful technique for learning predictive models from survival data (with censoring) which learns an ensemble of "survival trees", rather than standard regression trees. It uses the Log-rank test as the node-splitting criterion; this is a non-parametric test designed for comparing the survival distributions between

two (or more) groups (in this case, child nodes in a survival tree). It compares the hazard or survival functions at each observed event time. The Log-rank statistics is given by Equation (19):

$$\text{Log–rank statistics } = \frac{(O_i - E_i)^2}{\text{Var}\,(O_i - E_i)} \tag{19}$$

$$O_i - E_i = \sum_{j=1}^{k} (m_{ij} - e_{ij}) \tag{20}$$

$$e_{ij} = \left(\frac{n_{ij}}{n_{1j} + n_{2j}}\right) \times (m_{1j} + m_{2j}) \tag{21}$$

$$\text{Var}\,(O_i - E_i) = \sum_{j=1}^{k} \frac{n_{1j} n_{2j}\,(m_{1j} + m_{2j})\,(n_{1j} + n_{2j} - m_{1j} - m_{2j})}{(n_{1j} + n_{2j})^2\,(n_{1j} + n_{2j} - 1)} \tag{22}$$

In Equation (19), O_i is the sum of the number of observed failures in group i across all failure times and E_i is the expected value of the sum of the number of failures in group i across all failure times. To compute the Log-rank statistics, we need to calculate the term $O_i - E_i$, which is a measure of the overall differences of the survival or hazard function (curve) over all k failure times and is given by Equation (20), where e_{ij} is the expected number of failures for group i at the failure time j, as shown in Equation (21). *Var(O_i - E_i)* is the estimated variance, which involves the number of subjects in the risk set in each group (n_{ij}) and the number of failures in each group (m_{ij}) at time j. k is the number of distinct times of observed failures. The summation is over all distinct failure times. Note that when comparing any pair of survival functions, this calculation will be done for just one of the two groups since the absolute difference is the same for the two groups.

In addition, standard RSF uses a specific type of predicted outcome at their leaf nodes, based on the ensemble Cumulative Hazard Function (Ishwaran et al.,

2008), (Wang and Li, 2017), which was designed to cope with censored data. Hence, this replaces the normal prediction of target values at leaf nodes in random forests for regression (which cannot cope with censored data).

The ensemble CHF for a given subject is calculated as follows. First, for each tree in the RSF, the subject's feature values are used to find the leaf node used to predict the survival time for that subject. In each tree, the CHF for that subject is calculated using Equation 23 (the Nelson-Aalen estimate for CHF), setting t to the last observed failure time (so that all failure times are considered in the summation), and calculating the terms m_j and n_j for the j-th failure time based on all the subjects assigned to the same leaf node as the current subject. Finally, the ensemble CHF for a subject is simply the arithmetic mean of the CHF for that subject over all trees in the RSF.

$$H(t) = \sum_{j=0}^{t} \left(\frac{m_j}{n_j} \right) \tag{23}$$

Mortality can be interpreted as the expected number of deaths for the set of subjects at a leaf node. Specifically, the Mortality for the entire set of subjects assigned to a given leaf node is defined as the expected value for the sum of the Cumulative Hazard Function (CHF) values over all unique survival times $\{t_1, t_2, \ldots, t_m\}$ in the data, given the set of feature values in the path leading from the root until that leaf node. This is shown in Equation (24). Note that all instances that fall into the same leaf node h are predicted with the same mortality (M_h), i.e. the same expected number of deaths.

$$M_h = M_i = E_i \left(\sum_{j=0}^{m} H_h \left(t_j \right) \right) \tag{24}$$

In Equation (24), E_i is the expected value of mortality under the null hypothesis that all instances j are similar to i. A survival tree enforces a null hypothesis of similar survival within its leaf nodes; individuals in a leaf node share a common

estimated hazard function. $H_h(t_j)$ is the Nelson-Aalen estimate of the Cumulative Hazard Function (CHF) at time point t_j for all instances classified at leaf node h. The ensemble mortality e (predicted outcome) for individual i, denoted $M_{e,i}$, is: $M_{e,i} = \Sigma_{j=1...n} H_{e,h}(t_j)$.

2.4.4 Deep Survival Forest

In this subsection, we review the Deep Survival Forest (DSF) algorithm, a relatively new machine learning algorithm for survival analysis. In order to explain how the DSF algorithm functions, it is useful to start by reviewing some basic principles of Deep Learning (LeCun, Bengio and Hinton, 2015; Goodfellow, Bengio and Courville, 2016; Shrestha and Mahmood, 2019), one of the strongest kinds of machine learning techniques currently.

Recently, Deep Learning has obtained great success, especially when processing images, speech, text, etc. The state-of-art approach is known as Deep Neural Networks (LeCun, Bengio and Hinton, 2015). The term "deep" can be characterised by the existence of several hidden layers in neural networks, and the models are typically trained by means of the very popular Backpropagation technique. Although the algorithm has been around for several decades, it has only been implemented into practical usage in a deep learning context over these past years.

One theoretical explanation behind the successes of deep neural networks would be an increase in model complexity resulting in an increase in the learning ability of machine learning models (LeCun, Bengio and Hinton, 2015; Hu et al., 2021). Specifically, this was done by adding more layers in between the input layer and output layer, creating hidden layers which increase the embedding depths of the learned models.

On the other hand, complexity is not always beneficial for model learning; and there are a few potential drawbacks when models contain many layers of neurons, as follows.

First, this could lead to the model overfitting the training set, due to the model having too much capacity of learning — i.e., the model has high variance and its performance is unstable adjusting too much to the training data. A simple yet effective approach to reduce overfitting is by training the model on a large dataset. Even though a neural network is set to have a huge learning capacity (high complexity model), a sufficiently large number of instances can be fed to allow the model to generalise well.

Second, complex neural networks together with a huge dataset will be very expensive in terms of computational time. It is not a trivial task to algorithmise around with the limited amount of hardware resources. Fortunately, with the development of hardware technology, GPU (Graphics Processing Unit) acceleration is available to address the problem, i.e., GPUs which consist of hundreds or thousands of core processors are able to speed up the computational process, model training particularly. This helps to explain why deep learning has gained so much popularity and is considered to be one of the latest and most important advancements in artificial intelligence.

However, deep neural network algorithms also have some disadvantages, as follows. First, the predictive performance of neural network models is sensitive to their hyper-parameter settings (Yamashita et al., 2018), and they have many hyper-parameters, such as the number of hidden layers, the number of neurons within each layer, the batch size, the number of epochs, the choice of activation function, etc. Tuning these hyper-parameters is a non-trivial task since model training is computationally expensive itself. Furthermore, replicating the results of neural network experiments is tricky – i.e., it is difficult to reproduce the computational results reported in the literature for comparison purposes.

Second, the model's structure is usually inflexible, i.e., predetermined by the hyper-parameters' values. Model complexity should be adaptive, so it can be adjusted upon the data, which would vary from application to application.

Third, a neural network is a black box model due to its high complexity, so interpretability of the models and knowledge extraction from it are challenging – although a lot of research has focused on interpreting neural networks (Zhang et al., 2021; Ghorbani, Abid and Zou, 2019; Wang et al., 2018). As a consequence, the deep neural network approach might be impractical for high-stakes decisions, especially when explainability, ethics or knowledge discovery are concerned (Rudin, 2019).

Thus, using the same basic principle of deep learning, some studies proposed new types of deep learning methods which avoid the use of neural networks, as discussed next.

Motivation for the Deep Forest Algorithm

In (Zhou and Feng, 2019) the authors considered the three most important properties of the deep learning framework, in order to propose a Deep Forests algorithm that avoids the use of neural networks.

The first property is layer-by-layer processing, since adding layers to a network is usually more effective than simply adding base units such as neurons or base models. This approach increases not only the number of base units, but also the embedding depths of the learned model.

Second, the in-model feature transformation (or feature construction) is a relatively rare property in classic supervised machine learning algorithms. However, such property enables the model learning process not to be completely dependent on the original features, and it goes along well with the first property of multiple layers.

Third, sufficient model complexity is what makes the algorithms capable of learning complex relationships in the data. Otherwise, the models might underfit the data.

Therefore, (Zhou and Feng, 2019) proposed the Deep Forests (DF) algorithm,

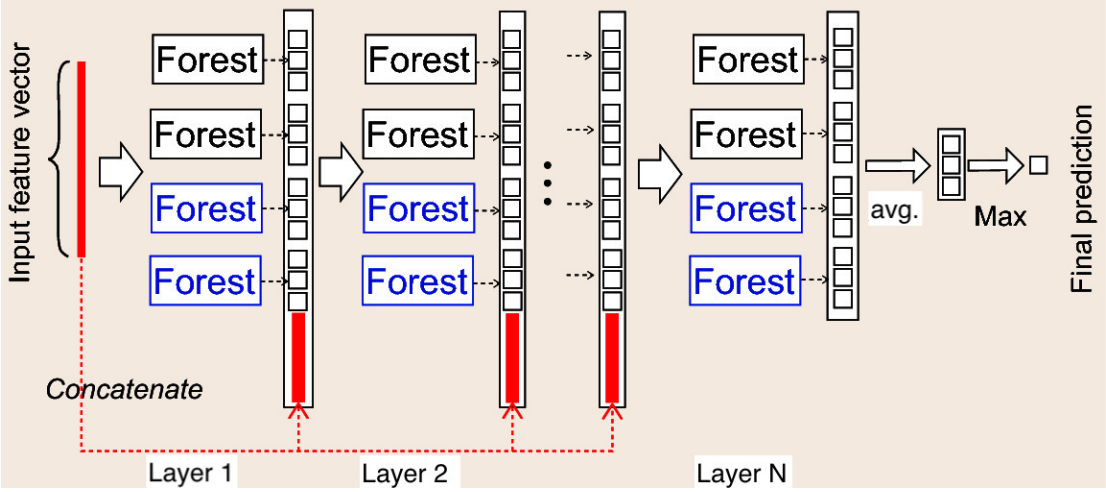


Figure 4: A graphical description of the Deep Forest algorithm, adapted from Zhou and Feng (2019).

as shown in Figure 4 with the following principles.

First, a set of Random Forests (RFs) are organized into a sequence of layers similar to that in deep neural networks, where each layer consists of RFs rather than neurons. Note that, in the same way that a RF is an ensemble of decision trees, a layer of a DF can be considered an ensemble of RFs, or an ensemble of ensembles, as it was called by the algorithm's authors. Hence, the layer-by-layer processing property is available in DF models.

Second, in order to perform in-model feature transformation, the stacking framework is employed. Essentially, the values of the target variable predicted by the RFs in one layer are fed forward to the next layer as additional features. That is, the original feature set is extended by a set of newly created features, and then together they are used as input for all RFs within the next layer. The number of additional features created in this step is equal to the number of RFs in the current layer — i.e., the set of predictions output by each RF will form one set of additional features. Note that the additional features are fed forward only to the next layer. Note also that, for classification tasks, the set of additional features contains $L \times M$ features, where L is the number of class labels and M is

the number of forests in each layer. More specifically, each feature contains the probability of a given class label computed by a given RF. For regression tasks, however, each forest only produces a single numeric value as its prediction, so the number of additional features, created to feed forward to the next layer, is M.

Third, for deep neural network, the number of layers is recognised to be one of the most influential hyper-parameters, since it technically defines model complexity. The more layers there are, the higher the complexity is. In order to make the Deep Forests algorithm more flexible, the number of layers is no longer one of the hyper-parameters whose value needs to be pre-defined; instead, the number of layers is automatically determined by the algorithm. Specifically, the training process is conducted based on the stacking framework where new layers are added one by one into the current DF model. A new layer is only added on the condition that it would improve the predictive accuracy, which is estimated by means of internal cross-validation (using the training set only, not the test set). In other words, the training process terminates when the validation performance on the training set no longer improves, compared against the performance of the current DF model that has been learned so far. In theory, the main benefit of such adaptive model complexity is to adjust the learning capacity of the model, so it would neither underfit nor overfit the training set.

From Deep Forests to Deep Survival Forests

In (Utkin et al., 2020, 2021) the authors proposed Deep Survival Forest (DSF), a new variant of the Deep Forest algorithm aimed to analyse survival data as well as taking into account the presence of censoring. In essence, the DSF algorithm learns a deep (multi-layer) model where each layer consists of multiple Random Survival Forests (RSFs), rather than multiple Random Forests as in the original Deep Forest algorithm.

The DSF algorithm is particularly relevant for this thesis since it is the basis

for the new variants of this algorithm proposed in Chapter 6. Hence, it is described in more detail next.

The pseudocode of the Deep Survival Forest (DSF) Algorithm 1

The pseudocode of the DSF algorithm is shown in Algorithm 1, which uses the following notation:

- N = the number of layers (automatically determined)
- M = the number of Random Survival Forests (RSFs) in each layer
- X_i = the feature set for the i-th layer
- X_1 = the original feature set
- K = the number of folds for internal cross-validation
- $RF_{(i,j,k)}$ = the j-th random forest in the i-th layer in the k-th cross-validation fold
- $Pred_{(i,j)}$ = the prediction of the j-th forest in the i-th layer
- $C\_index\_CV_i$ = c-index value of the ensemble of RFs in the i-th layer

Lines 1-3 in this pseudocode simply initialise some variables. Line 4 is a loop over the layers of the Deep Forest model. For each iteration, there is another loop at Line 6 for an internal cross-validation: the training data are divided into K folds with approximately equal distribution of censored and uncensored instances across the folds.

After that, the third loop at Line 8 goes over the RSFs in the current layer. In this loop, two different types of RSFs are trained per iteration: one standard RSF and one Extra-RSF. An Extra-RSF is a type of tree-based ensemble model that fits a number of extremely randomised survival trees, a.k.a. extra-trees, introduced in (Geurts, Ernst and Wehenkel, 2006). Compared to a classic survival tree in (Segal, 1988), an extremely randomised survival tree is trained with one extra level of randomness in the way splits are computed. As in RSF, a random subset

Algorithm 1: The pseudocode of the Deep Survival Forests algorithm.

$N \leftarrow 0$
$C\_index\_CV_0 \leftarrow 0$
$i \leftarrow 1$
repeat
  $C\_index\_CV_i \leftarrow 0$
  for $k \leftarrow 1$ *to* K **do**
    $valid\_Y_k \leftarrow$ Set fold k as the validation set
    for $j \leftarrow 1$ *to* $\frac{M}{2}$ **do**
      Train a Random Survival Forest model $RF_{(i,j,k)}$ on remaining $K-1$ folds (learning set) with feature set X_i
      Train a completely-Random Survival Forest $RF_{(i,j+\frac{M}{2},k)}$ on remaining $K-1$ folds (learning set) with feature set X_i
    end
    $Ensemble\_RF_{i,k} \leftarrow$ aggregation of all $RF_{(i,1,k)} \dots RF_{(i,M,k)}$
    $C\_index\_CV_i +=$ C index of $(Ensemb\_RF_{i,k}, valid\_Y_k)$
  end
  $C\_index\_CV_i \leftarrow \frac{C\_index\_CV_i}{K}$
  for $j \leftarrow 1$ *to* M **do**
    $Pred_{(i,j)} \leftarrow$ aggregation of the predicted outputs from $RF_{(i,j,1)} \dots RF_{(i,,j,K)}$
    $Add\_x_{ij} \leftarrow Pred_{(i,j)}$
  end
  $X_{i+1} \leftarrow X_1 + \{Add\_x_{i1}, Add\_x_{i1}, \dots, Add\_x_{iM}\}$
  $N++$
until $C\_index\_CV_{i-1} \geq C\_index\_CV_i$
Use as final model the ensemble of RFs learned in the $(N-1)$th layer (previous layer)
$Prediction \leftarrow$ Average of the predicted outputs from $Pred_{(N,1)}, Pred_{(N,2)}, \dots, Pred_{(N,M)}$

of candidate features is used, but instead of looking for the most discriminative thresholds, thresholds are selected at random for each candidate feature and the best of these randomly-generated thresholds is picked as the splitting criterion. Note that in each extremely randomised survival tree, the quality of a split is also measured by the log-rank statistics.

Thus, this step builds M RSFs within the current layer, with half of them being built in the standard way and the other half being built completely at random, to increase the diversity of RSFs within the current layer. All these RSFs are learned from the k-th learning set of the internal cross-validation procedure.

All those M RFs are then aggregated into an ensemble of RFs for the current i-th layer and the current k-th cross-validation fold in line 12. Then, in line 13, the algorithm computes the value of the C-index (as a measure of predictive accuracy) of that ensemble of RFs on the k-th validation set of the internal cross-validation procedure, and the result is added to the total value of the C-index, over all K cross-validation folds, for the current i-th layer of the deep RSF model. Once the internal cross-validation loop from line 6 to line 14 is completed, in line 15 the algorithm computes the final value of the C-index for the i-th layer, as the mean of the C-index values over the K internal cross-validation folds.

Next, in the loop starting at line 16, M additional features are created – each of which is the output predicted by a RSF model. More specifically, previously each RSF was trained using the k-th learning set k and its predictions were computed on $valid\_Y_k$; and so in line 17, for each RSF, its predictions on all K validation folds are aggregated to create a new feature. As a result, this step creates M new features. Afterward, at line 20 the original feature set and the additional feature set are concatenated to be used as input features for training the RSFs in the next layer.

Finally, line 22 determines whether or not there has been any improvement in the predictive performance (C-index value) from the previous layer to the current

layer. If the C-index for the current layer is better than the C-index for the previous layer, then the algorithm will construct the next layer of RSFs, otherwise, it terminates the training process.

2.5 Conclusion

This chapter provided a comprehensive review of the background on survival analysis and machine learning algorithms relevant to this thesis. This section concludes this chapter by describing the rationale about why the new methods proposed in this thesis could lead to better predictive accuracy results.

The state-of-the-art methods in survival analysis, such as Cox regression, RSF, and DSF, have made significant contributions to the survival-modeling field. However, when considering biomedical datasets and the prediction of actual survival times, there are several gaps in these existing approaches that motivate the contributions of this thesis.

First, biomedical datasets often exhibit unique characteristics, such as high-dimensional features, complex interactions, and potential confounding factors. Traditional statistical approaches like Cox regression may struggle to effectively handle these complexities, potentially leading to suboptimal predictions. Therefore, machine learning approaches are more practical to capture these characteristics of biomedical datasets.

Second, biomedical datasets often contain non-linear relationships among the features and the target variable. Cox regression assumes a linear relationship between the features and the hazard function, which often is not true in real-world datasets. RSF and DSF offer more flexibility in capturing non-linear relationships, but they may not fully exploit the complex relationships present in biomedical datasets — e.g., they may not exploit the important role of the Age variable to improve predictive accuracy in survival datasets of age-related diseases (like the

datasets used in this thesis).

Third, while Cox regression and RSF are commonly used for survival analysis, they primarily focus on estimating relative hazards (risks) or survival probabilities rather than predicting the actual survival times. Predicting accurate and precise survival times is vital in biomedical research, since it could support clinicians and researchers to make some crucial decisions such as patient management, treatment strategies, and clinical trial design. The development of methods specifically aimed at predicting actual survival times would be a significant advancement in the field.

Through the contributions in this thesis, it is anticipated that the proposed method will enrich not only the performance, but also interpretability of survival models in the biomedical domain. The application of these novel approaches to real-world biomedical datasets holds the potential to support clinical decision-making, patient care, and advancements in biomedical research.

Chapter 3

Data Preparation

This chapter provides information about the datasets used in the experiments to evaluate the proposed variations of random forests for survival analysis, and the data preparation that was performed for the purposes of running the survival analysis algorithms. It describes in particular the creation of the target variable for each dataset, as well as the predictive features in the datasets.

3.1 Introduction

This chapter describes the creation of 11 survival analysis datasets, all in the application domain of biomedical data. This is in general the main application domain for survival analysis methods, since right-censored data naturally occurs in this domain. Although there are some repositories of biomedical survival analysis datasets Desmedt et al. (2007); Kalbfleisch and Prentice (2011), this thesis focuses on creating new survival analysis datasets mainly in order to exploit the relatively large amount of data in longitudinal studies of ageing, which has been unexplored so far in the survival analysis literature. Such studies are particularly important for biomedical research, as discussed in more detail below. In addition,

the creation of new datasets is more likely to lead to the discovery of new biomedical knowledge or patterns, by comparison with the use of biomedical datasets which have already been extensively used in the survival analysis literature.

Hence, 10 out of the 11 created datasets contain data about age-related diseases from longitudinal studies of ageing. The motivation for focusing on such datasets of age-related diseases is as follows.

As mentioned in the Introduction, according to a World Health Organization's report on World Population Ageing (WHO, 2022), one in six human beings on earth will be 60 or older by 2030. The rise in the proportion of elderly people will place significant strain on healthcare systems due to the higher prevalence of age-related diseases among the elderly, considering that the elderly tend to suffer from multiple age-related diseases (George, Elliott and Stewart, 2008). Consequently, there is an increasing demand for statistical or supervised machine learning methods to analyse ageing-related data, including information about age-related diseases. The objective is to gain a better understanding of these diseases and potentially enhance the corresponding diagnoses and treatments.

The last created dataset described in this chapter is the haemodialysis dataset. This dataset is important because about 65% of people commencing renal replacement therapy in the UK also commence haemodialysis UK Renal Registry (2020), and in total over 25,000 people receive haemodialysis in the UK.

The author is investigating the possibility of making the datasets available to the research community.

The next three sections describe in detail the dataset creation process for each of the datasets created in this thesis.

3.2 The ELSA datasets

Most of the created datasets (9 out of 11) were derived from the English Longitudinal Study of Ageing (ELSA) (Clemens et al., 2019) — `https://www.elsa-project.ac.uk/`. The ELSA study is a longitudinal survey of ageing and quality of life among older people that explores the dynamic relationships between health and functioning, social networks and participation, and economic position as people plan for, move into and progress beyond retirement. In this work, however, we focus only on the biomedical data in ELSA, such as the results of blood tests and other data collected by nurses, and the relationship between that data and the health status of patients, as will be described in more details later.

There are four important issues about the data collection process used to create the ELSA database. First, the ELSA subjects were recruited from a representative sample of the English population, who live in private households, aged 50 and over. Follow-up interviews were subsequently conducted with these participants. Second, the data has been collected every two years, where each data collection period is known as a 'wave'. In total, eight waves of data have been collected and have well-documented data. Third, follow-up interviews were conducted with these participants in subsequent waves, so that we can observe the variation of each feature's values for each individual across those waves. It should be noted that efforts are made to track and re-interview participants in subsequent waves, even if they have moved or experienced changes in their circumstances. Attrition, or the loss of participants over time, is a common challenge in longitudinal studies. However, ELSA has implemented various strategies to mitigate attrition and maintain a representative sample. Last but not least, ELSA incorporates nurse visits in selected waves. These nurse visits provide an opportunity to collect detailed health-related information, including physical measurements, biological samples (such as blood and saliva), and functional assessments. These data enhance the understanding of health conditions, biomarkers, and physical functioning among

the study participants.

Since the data in the ELSA database was not collected specifically for machine learning purposes, it was necessary to spend a large amount of time with data preparation for the survival analysis. The first step was to define the instances (subjects) used in the created datasets, the target variables to be predicted and the predictive features used for prediction. In essence, the instances represent individuals in the ELSA database, the target variables represent the time passed until the first diagnosis of some age-related diseases, and the predictive features represent biomedical information collected by nurses or other relevant characteristics of an individual (age and gender).

3.2.1 Creating target variables and censoring variables

There are two special types of variables which have been generally used together for each survival prediction problem in most survival studies in the literature. The first one is the "target" variable, whose values are to be predicted. In order to create the target variables, we looked into the ELSA core data, and then identified some age-related diseases. Unfortunately, some diseases, such as Parkinson's, did not have the variables we needed to use as base variables in order to construct the target variables, as explained below. At last, we ended up with 8 diseases which had all necessary variables provided. These diseases are Angina, Heart Attack, Diabetes, Stroke, Arthritis, Alzheimer's, Cancer and Psychiatric.

Hence, we created 8 target variables, one for each of these diseases. In this work, each target variable takes a numerical value indicating the time passed (in months) from the date when a subject received a nurse visit in wave 2 (for collecting the subject's biomedical data) until the date when the subject was first diagnosed with the disease corresponding to that variable.

The second special type of variable typically used in survival analysis is the uncensored status variable, which takes the values "1" or "0" to represent the

uncensored or censored values (respectively) of a target variable for each subject. In this work, the uncensored ("1") and censored ("0") values indicate whether or not (respectively) we know the date when a patient was first diagnosed with the corresponding disease within the study period.

Note that target and uncensored status variables come in pairs, with one such pair for each of the above 8 diseases. Hence, the uncensored status value "1" means the value stored in the corresponding target variable for a given disease and a given subject represents the true, known value for the time passed until the first diagnosis; whilst the uncensored status value "0" means the value stored in the corresponding target variable is just a "lower bound" of the true, unknown value, since that value is censored. Therefore, a survival-time prediction method has to interpret the value of the target variable in the context of the value of the uncensored status variable for each disease and each subject (instance) in the dataset. To determine the value of each target variable and each uncensored status variable for each subject, we distinguish between two cases, depending on whether or not the information about that subject's first date of diagnosis for the disease corresponding to the target variable is censored. These two cases are discussed in detail next.

First Case: Determining the value of a target variable for uncensored subjects

For each of the 8 diseases used to create the target variables in this thesis, the ELSA database contains two variables indicating the year and the month the subject was first diagnosed with that disease. These variables are hereafter called first_diag_year and the first_diag_month, respectively. Hence, if the values of both these variables for a given disease are known (i.e., not missing) for a given subject, then the date of that disease's first diagnosis for that subject is not censored, and so the value of the target variable for that subject is directly determined by

Equation 25:

$$Target = (first\_diag\_year - nurse\_year) \times 12 + (first\_diag\_month - nurse\_month) \tag{25}$$

where the variables nurse_year and nurse_month represent the year and the month of the nurse visit to the subject in wave 2.

Note that wave 2 (rather than wave 1) is the baseline wave for our task of predicting the time passed until the first diagnosis of some age-related disease, because wave 2 is the first wave for which there are biomedical variables in the Nurse data section of the ELSA database, and those biomedical variables were used to create the predictive features in our dataset, as explained later.

When using Equation 25, since there is no censorship for the date of the first diagnosis for the current subject and disease, the value of the corresponding uncensored status variable is set to 1.

As an example of the use of this equation, suppose that for a certain disease (target variable) a subject has: "first_diag_year = 2010", "first_diag_month = 11", "nurse_year = 2005", "nurse_month = 1". Then, using Equation 25 for this subject, the target variable's value is computed as follows:

$$Target = (2010 - 2005) \times 12 + (11 - 1) = 70\ months$$

This means that we know that the subject was not diagnosed with the disease (i.e. she/he "survived" in the terminology of survival analysis) for 70 months since the nurse visit in the baseline wave 2.

Second Case: Determining the value of a target variable for censored subjects

Unfortunately, in many cases the information about the year and the month when the subject was first diagnosed with a certain disease is censored in the ELSA database, i.e., the values of the first_diag_year and the first_diag_month variables for that disease are missing for that subject. In this case, the computation of the value to be assigned to the corresponding target variable is considerably more complex than in the above case of uncensored subjects, as follows.

First of all, for each instance (subject), for each target variable (disease) and for each wave (time point), we compute the value of a binary variable that takes the value "1" or "0" to indicate whether or not (respectively) that subject was known to have a diagnosis of that disease in that wave. These created variables have a name of the form:

$$wX\_DiseaseName\_known\_diag,$$

where wX denotes the wave number X (for $X = 2, \ldots, 8$). For instance, the variable w2_Angina_known_diag indicates whether or not the subject was known to have a diagnosis of Angina at wave 2.

Hereafter we use the term known_diag as a shorthand notation to refer to the above type of *wX_DiseaseName_known_diag* variables when discussing them in a generic way, referring to any wave and any disease. We will explain later the (quite complex) computation of such *known_diag* variables. Before that explanation, let us describe how these variables are used as intermediate variables for the computation of the value of each target disease, in the case of censored data. The basic idea is that, by comparing the values of these *known_diag* variables for a given subject and a given target disease across all waves, we can determine the last date (year and month) when the subject was observed and still did not have

the diagnosis for the target disease; and then we can finally directly compute the value of the (censored) target variable as the number of months passed between the data of the nurse visit for that subject in the baseline wave 2 and the last date when the subject was observed without a diagnosis for that disease. This last date acts as a "lower bound" for the date of the first diagnosis for that disease, in the context of a censored target variable.

More precisely, the *known_diag* variables for a disease are used to compute the value of a new *last_no_diag_year* variable, which indicates the year when the subject was last observed without a diagnosis. The value of the *last_no_diag_year* variable for each subject and each disease is computed in two steps, as follows.

Firstly, the system looks for the last wave number in which the corresponding *known_diag* has the value '0'. This means that we know that the subject was observed in that wave, but unfortunately, there is no variable in the ELSA database recording the date when the subject answered the questionnaire about disease diagnoses. Hence, the information about lack of diagnosis for that disease for that subject could have occurred in any month out of a two-year period associated with that last wave where "$known\_diag = 0$". Therefore, we only know that the subject was not diagnosed with the disease in the first month of that two-year period, i.e., we cannot conclude that the subject was not diagnosed with the disease at later months in that two-year period.

Hence, the second step for computing the value of the *last_no_diag_year* variable consists of assigning to it the first year out of the two years associated with that wave based on the ELSA's timetable as shown in Table 4. Finally, the value of the target variable is then computed by Equation 26:

$$Target = (last\_no\_diag\_year - nurse\_year) \times 12 + (1 - nurse\_month) \quad (26)$$

This is analogous to Equation 25, with the difference that the *first_diag_month*

Table 4: Timetable of the ELSA project from wave 1 to wave 8

Wave number	Years
1	2002 - 2003
2	2004 - 2005
3	2006 - 2007
4	2008 - 2009
5	2010 - 2011
6	2012 - 2013
7	2014 - 2015
8	2016 - 2017

month in Equation 26 is replaced by "1" (referring to January of the year stored in *last_no_diag_year*) in Equation 25 since the information about whether the subject was diagnosed later is censored, as discussed earlier. To clarify how the above procedure for computing the target variable works in the case of censored subjects, we will use as an example the data for two subjects shown in Table 5. As shown in that table, subject 1 did not have a diagnosis of Angina in waves 2, 3 and 4, which was the last wave when the subject was observed - which is indicated by the fact that the *Angina_known_diag* variables for waves 5 through 8 have missing values (denoted by "?" in the table). As shown in Table 4, the first year associated with wave 4 is 2004, and so the variable *last_year_no_diag* is set to 2008. In addition, the date of the nurse visit for that subject was December 2004. Hence, according to Equation 26, the value of the target variable for subject 1 is computed as:

$$Target_i = (2008 - 2004) \times 12 + (1 - 12) = 37 months$$

This means that we know that subject 1 was not diagnosed with the disease (i.e. she/he "survived" in the terminology of survival analysis) for at least 37 months since the nurse visit in wave 2. We emphasize that in this case the target

Table 5: Values of the variables used for calculating the value of the target Angina variable (as an example target disease), for two subjects used as examples. The variables used in this table are needed to compute the value of the target variable only for subjects whose first disease diagnosis date is censored, as explained in the text. The symbol "?" denotes a missing value for a variable.

Subject	Angina known diag							nurse year	nurse month
	w2	w3	w4	w5	w6	w7	w8		
1	0	0	0	?	?	?	?	2004	12
2	0	0	0	1	1	1	1	2005	4

variable is assigned a value representing only a lower bound for the true, unknown number of months until the first diagnosis of a disease, since the subject's target variable for that disease is censored. Therefore, in this case, the system also sets the value of the uncensored status variable to 0 for that subject and that disease.

A similar computation is performed for subject 2 in Table 5. Although the values of the Angina_know_diag variables for this subject are known for all waves, we still cannot assign any precise date to the first diagnosis of Angina for this subject. That is, subject 2 could have been first diagnosed with Angina at any time during wave 5 (the first wave with "Angina_diag_known = 1") or at any time during wave 4, which is the last wave with "Angina_diag_known = 0", since the ELSA database does not record the precise date when the subject answered the questionnaire about disease diagnosis, as mentioned earlier. We only know that at the start of wave 4 subject 2 did not have her/his first Angina diagnosis yet, and subject 2 had her/his first Angina diagnosis before the end of wave 5.

Hence, as explained above for subject 1 in Table 5, in order to compute the target variable's value for subject 2 in that table, we also set the value of the variable last_no_diag_year to 2008, the first year of wave 4 as shown in Table 4, since wave 4 is the last wave when the subject was observed not to have her/his first Angina diagnosis.

The computation of the target variable's value for subject 2 proceeds in the

same way as for subject 1, i.e. by applying Equation 26 we get:

$$Target_2 = (2008 - 2005) \times 12 + (1 - 4) = 33 months$$

Now that the role of the *know_diag* variables has been explained, we turn to a detailed description of the procedure used to compute the values of these variables.

To compute such variables, we needed a variable in the ELSA database that indicated when an individual was diagnosed with the disease. Unfortunately, there were no such variables directly providing this information in the database. However, this information was rather represented indirectly by several related variables whose values depend on the individual's answer to questions like whether or not the individual still had a previously diagnosed disease, whether the previously diagnosed disease was confirmed or whether the disease was newly reported. Therefore, we needed to create a well-defined *know_diag* variable for each disease separately, by combining information from the several related variables associated with that disease in the ELSA database. These variables obtained directly from the ELSA database were called "base" disease variables.

Table 6 shows the set of rules used for creating the *know_diag* variables for each disease and each wave. The first column of this table shows the names of the diseases, which correspond to the aforementioned 8 target variables. The second column shows the names of the *known_diag* variables, where each name has a prefix denoting the wave number and the disease name, as described earlier. The last column (Rule) of this table shows the precise rules used to compute the values of the *know_diag* variables, by combining information from the base disease variables for each disease and each wave separately. In that last column, the base disease variables occurring just before each "=" sign in an "IF" part of the rule refers to base variables in ELSA's core data from the corresponding waves.

Taking the assignment of the *w3_HeartAtt_known_diag* variable as an example, the variables in the three conditions ($w3hedacmi = 1$), ($w3hediami = 1$) and

(*w3dhedimmi* = 1) represent "whether confirms heart attack diagnosis" = "yes", "whether still has heart attack" = "yes" and "heart attack diagnosis newly reported" = "yes" respectively. Joining these three conditions together with the "OR" operator means that if the value of the variable in any of these conditions is set to "yes", then the *w3_HeartAtt_known_diag* variable is assigned the value "yes". Otherwise, the *w3_HeartAtt_known_diag* variable takes the value "no".

Note that the base variables' names in wave 2 for every disease are different from those in the other waves. For instance, both *w2HeDiaC2* and *w3hedacan* represent "Whether confirms angina diagnosis". The variable names in wave 3 are very similar to the names from wave 4 onward, and the variable names in waves 4 through 8 are the same. Hence, to avoid redundancy, the Rule column of Table 6 shows just one entry using the symbol "X" to denote a variable's wave number varying in the range from 4 to 8. In addition, the full list of the base disease variables used in these rules is reported in Table 7.

3.2.2 Creating the "Any-disease" target variable

The "Any-disease" target variable represents the time passed until the first diagnosis for any of the eight diseases. The value of this variable is computed based on the values of the uncensored status variable (representing censorship status) and the target variable (representing survival times) for each of the eight diseases. The diseases included Angina, Heart Attack, Diabetes, Stroke, Arthritis, Alzheimer's, Cancer and Psychiatric Disorder.

The main motivation for creating the ELSA "Any-disease" dataset is to create a different type of dataset with a larger uncensoring ratio, since 7 out of the 8 other ELSA datasets have uncensoring ratios below 10% (heavily censored datasets). Another motivation for this dataset creation lies in a potentially more comprehensive understanding of disease progression and its impact on individuals' health and well-being – i.e., to gain insights into the overall disease burden and

Table 6: The rules used for computing the values of the known_diag variables for each disease and each wave

Disease (target)	Intermediate Variable Name	Rule
Angina	w2_Angina_known_diag	IF (w2HeDiaC2 = 1) OR (w2HeDiaS2 = 1) THEN w2Angina_known_diag = 1 ELSE w2Angina_known_diag = 0
	w3_Angina_known_diag	IF (w3hedacan = 1) OR (w3hedasan = 1) OR (w3hediaan = 1) OR (w3dhediman = 1) THEN w3Angina_known_diag = 1 ELSE w3Angina_known_diag = 0
	wX_Angina_known_diag	IF (wXhedacan = 1) OR (wXhedasan = 1) OR (wXhediaan = 1) OR (wXhediman = 1) THEN wXAngina_known_diag = 1 ELSE wXAngina_known_diag = 0
HeartAtt	w2_HeartAtt_known_diag	IF (w2HeDiaC3 = 1) THEN w2HeartAtt_known_diag = 1 ELSE w2HeartAtt_known_diag = 0
	W_3HeartAtt_known_diag	IF (w3hedacmi = 1) OR (w3hediami = 1) OR (w3dhedimmi = 1) THEN w3HeartAtt_known_diag = 1 ELSE w3HeartAtt_known_diag = 0
	wX_HeartAtt_known_diag	IF (wXhedacmi = 1) OR (wXhediami = 1) OR (wXhedimmi = 1) THEN wXHeartAtt_known_diag = 1 ELSE wXHeartAtt_known_diag = 0
Diabetes	w2_Diabetes_known_diag	IF (w2HeDiaC7 = 1) OR (w2HeDiaS7 = 1) THEN w2Diabetes_known_diag = 1 ELSE w2Diabetes_known_diag = 0
	w3_Diabetes_known_diag	IF (w3hedacdi = 1) OR (w3hediadi = 1) OR (w3dhedimdi = 1) THEN w3Diabetes_known_diag = 1 ELSE w3Diabetes_known_diag = 0
	wX_Diabetes_known_diag	IF (wXhedacdi = 1) OR (wXhediadi = 1) OR (wXhedimdi = 1) THEN wXDiabetes_known_diag = 1 ELSE wXDiabetes_known_diag = 0
Stroke	w2_Stroke_known_diag	IF (w2HeDiaC8 = 1) THEN w2Stroke_known_diag = 1 ELSE w2Stroke_known_diag = 0
	w3_Stroke_known_diag	IF (w3hedacst = 1) OR (w3hediast = 1) OR (w3dhedimst = 1) THEN w3Stroke_known_diag = 1 ELSE w3Stroke_known_diag = 0
	wX_Stroke_known_diag	IF (wXhedacst = 1) OR (wXhediast = 1) OR (wXhedimst = 1) THEN wXStroke_known_diag = 1 ELSE wXStroke_known_diag = 0
Arthritis	w2_Arthritis_known_diag	IF (w2HeDiaD3 = 1) OR (w2HeDiDS3 = 1) OR (w2HeDiaS3 = 1) THEN w2Arthritis_known_diag = 1 ELSE w2Arthritis_known_diag = 0
	w3_Arthritis_known_diag	IF (w3hedbdar = 1) OR (w3hedbsar = 1) OR (w3dhedibar = 1) THEN w3Arthritis_known_diag = 1 ELSE w3Arthritis_known_diag = 0
	wX_Arthritis_known_diag	IF (wXhedbdar = 1) OR (wXhedbsar = 1) OR (wXhedibar = 1) THEN wXArthritis_known_diag = 1 ELSE wXArthritis_known_diag = 0
Alzheimer	w2_Alzheimer_known_diag	IF (w2HeDiaD8 = 1) OR (w2HeDiDS8 = 1) OR (w2HeDiaS8 = 1) THEN w2Alzheimer_known_diag = 1 ELSE w2Alzheimer_known_diag = 0
	w3_Alzheimer_known_diag	IF (w3hedbdad = 1) OR (w3dhedibad = 1) THEN w3Alzheimer_known_diag = 1 ELSE w3Alzheimer_known_diag = 0
	wX_Alzheimer_known_diag	IF (wXhedbdad = 1) OR (wXhedibad = 1) THEN wXAlzheimer_known_diag = 1 ELSE wXAlzheimer_known_diag = 0
Cancer	w2_Cancer_known_diag	IF (w2HeDiaD5 = 1) OR (w2HeDiDS5 = 1) THEN w2Cancer_known_diag = 1 ELSE w2Cancer_known_diag = 0
	w3_Cancer_known_diag	IF (w3hedbdca = 1) OR (w3hedbsca = 1) OR (w3dhedibca = 1) THEN w3Cancer_known_diag = 1 ELSE w3Cancer_known_diag = 0
	wX_Cancer_known_diag	IF (wXhedbdca = 1) OR (wXhedbsca = 1) OR (wXhedibca = 1) THEN wXCancer_known_diag = 1 ELSE wXCancer_known_diag = 0
Psychiatric	w2_Psychiatric_known_diag	IF (w2HeDiaD7 = 1) OR (w2HeDiDS7 = 1) THEN w2Psychiatric_known_diag = 1 ELSE w2Psychiatric_known_diag = 0
	w3_Psychiatric_known_diag	IF (w3hedbdps = 1) OR (w3dhedibps = 1) THEN w3Psychiatric_known_diag = 1 ELSE w3Psychiatric_known_diag = 0
	wX_Psychiatric_known_diag	IF (wXhedbdps = 1) OR (wXhedibps = 1) THEN wXPsychiatric_known_diag = 1 ELSE wXPsychiatric_known_diag = 0

Table 7: Base disease variables' description used in the rules in Table 6

Base Variable Name	Label
w2HeDiaC2	Whether confirms angina recorded in wave 1
w2HeDiaS2	Whether still has angina at wave 2
w3hedacan	Whether confirms angina diagnosis
w3hedasan	Whether still has angina
w3hediaan	CVD: angina diagnosis newly reported
w3dhediman	CVD: angina diagnosis newly reported (merged)
wXhedacan	Whether confirms angina diagnosis
wXhedasan	Whether still has angina
wXhediaan	CVD: angina diagnosis newly reported
wXhediman	CVD: angina diagnosis newly reported (merged)
w2HeDiaC3	Whether confirms heart attack recorded in wave 1
w3hedacmi	Whether confirms heart attack diagnosis
w3hediami	CVD: heart attack diagnosis newly reported
w3dhedimmi	CVD: heart attack diagnosis newly reported (merged)
wXhedacmi	Whether confirms heart attack diagnosis
wXhediami	CVD: heart attack diagnosis newly reported
wXhedimmi	CVD: heart attack diagnosis newly reported (merged)
w2HeDiaC7	Whether confirms diabetes recorded in wave 1
w2HeDiaS7	Whether still has diabetes at wave 2
w3hedacdi	Whether confirms diabetes or high blood sugar diagnosis
w3hediadi	CVD: diabetes or high blood sugar diagnosis newly reported
w3dhedimdi	CVD: diabetes or high blood sugar diagnosis newly reported (merged)
wXhedacdi	Whether confirms diabetes or high blood sugar diagnosis
wXhediadi	CVD: diabetes or high blood sugar diagnosis newly reported
wXhedimdi	CVD: diabetes or high blood sugar diagnosis newly reported (merged)
w2HeDiaC8	Whether confirms stroke recorded in wave 1
w3hedacst	Whether confirms stroke diagnosis
w3hediast	CVD: stroke diagnosis newly reported
w3dhedimst	CVD: stroke diagnosis newly reported (merged)
wXhedacst	Whether confirms stroke diagnosis
wXhediast	CVD: stroke diagnosis newly reported
wXhedimst	CVD: stroke diagnosis newly reported (merged)
w2HeDiaD3	Whether confirms arthritis recorded in wave 1
w2HeDiDS3	Whether still had arthritis at wave 2
w2HeDiaS3	Whether still has arthritis at wave 2
w3hedbdar	Whether confirms arthritis diagnosis
w3hedbsar	Whether still has arthritis
w3dhedibar	Chronic: arthritis diagnosis newly reported
wXhedbdar	Whether confirms arthritis diagnosis
wXhedbsar	Whether still has arthritis
wXhedibar	Chronic: arthritis diagnosis newly reported
w2HeDiaD8	Whether confirms Alzheimers disease recorded in wave 1
w2HeDiDS8	Whether still had Alzheimers disease at wave 2
w2HeDiaS8	Whether still has Alzheimers disease at wave 2
w3hedbdad	Whether confirms Alzheimers Disease diagnosis
w3dhedibad	Chronic: Alzheimers Disease diagnosis newly reported
wXhedbdad	Whether confirms Alzheimers Disease diagnosis
wXhedibad	Chronic: Alzheimers Disease diagnosis newly reported
w2HeDiaD5	Whether confirms cancer recorded in wave 1
w2HeDiDS5	Whether still had cancer at wave 2
w3hedbdca	Whether confirms cancer diagnosis
w3hedbsca	Whether still has cancer
w3dhedibca	Chronic: cancer diagnosis newly reported
wXhedbdca	Whether confirms cancer diagnosis
wXhedbsca	Whether still has cancer
wXhedibca	Chronic: cancer diagnosis newly reported
w2HeDiaD7	Whether confirms psychiatric problems recorded in wave 1
w2HeDiDS7	Whether still had psychiatric problems at wave 2
w3hedbdps	Whether confirms psychiatric condition diagnosis
w3dhedibps	Chronic: psychiatric condition newly reported
wXhedbdps	Whether confirms psychiatric condition diagnosis
wXhedibps	Chronic: psychiatric condition newly reported

Table 8: the censoring distribution of instances based on different diseases

Disease	**Censored**	**Uncensored**	**Missing**	**Before-nurse-visit**
Angina	5774	307	788	390
HeartAtt	6099	287	788	271
Diabetes	5738	575	796	370
Stroke	6154	407	790	244
Arthritis	3158	1276	498	2892
Alzheimer	6706	78	830	11
Cancer	5695	775	739	541
Psychiatric	5915	422	768	707

its associated risk factors. Analysing this engineered variable may help to identify common patterns and shared risk factors among several diseases at the same time, which could potentially have broader implications for healthcare systems and public health strategies.

Before starting explaining the creation process of the Any-disease target variable, it should be noted that, for each disease, the subjects in the dataset could be categorised into four cases depending on the values of their corresponding target variable. Table 8 reports the number of instances for each case using four columns, where Censored means censored instances whose target variables took the lower-bounds; Uncensored means uncensored instances whose target variables took the observed survival times; Missing represents instances with missing values on the target variables; Before-nurse-visit refers to instances being diagnosed with the corresponding disease before the date of nurse visit in wave 2 of the ELSA survey and having the target variables with negative values or zeros.

Because of this, if we were to create the Any-disease target variable using all four cases of instances, then the resulting dataset would have 2,190 uncensored instances out of 6,837 with the following diseases as the cause:

- Arthritis 697
- Cancer 462
- Diabetes 333

- Stroke 212
- Psychiatric 175
- HeartAtt 137
- Angina 125
- Alzheimer 49

However, this would ruin the definition of the Any-disease target variable by including the before-nurse-visit instances, since these subjects were diagnosed with the corresponding disease before the time at which the features of the datasets had been collected. Furthermore, the same issue could be considered for instances in the missing case as both their survival times and lower-bound were unknown.

Hence, in order to create a new dataset with the new target variable "Any-disease", only instances where all 8 diseases were categorized into the censored or uncensored cases were included in the dataset; i.e. all instances where any disease was categorized into the Missing or Before-nurse-visit cases were discarded. This led to a substantial loss in the number of instances in the new dataset (979 uncensored instances out of 3,280), with the following distribution:

- Arthritis 443
- Cancer 202
- Diabetes 116
- Stroke 67
- Psychiatric 64
- HeartAtt 56
- Angina 45
- Alzheimer 10

Recall that the target and uncensored status variables come in pairs for each of the 8 diseases. These pairs were used to determine the value of the Any-disease

Table 9: Example with two diseases considering all possible pairs of cases of censored and uncensored statuses

Subject	Cancer		Diabetes	
	Uncens.	Time	Uncens.	Time
1	1	6 months	1	9 months
2	0	6 months	0	9 months
3	1	6 months	0	9 months
4	0	6 months	1	9 months

target variable for each subject as well as to distinguish between four possible cases, depending on the values of those two variables for each disease.

To explain the computation of the value of the Any-disease target variable, Figure 5 illustrates the procedure for engineering the Any-disease target variable. It provides a visual representation of the steps involved in determining the value of the Any-disease target variable for each subject and how it relates to the pairs of target and uncensored status variables for the 8 diseases.

To illustrate this point, we will describe this variable creation for a simple hypothetical scenario involving just two diseases, but the variable creation process can be straightforwardly generalized to the scenario of 8 diseases in the ELSA dataset. In this two-disease scenario, there are four possible cases for each instance, regarding whether or not the instance is censored for each of the two diseases. Considering, for instance, Cancer and Diabetes as the two diseases, the four possible cases are:

1. both diseases uncensored,
2. both diseases censored,
3. one disease (Cancer) uncensored at an earlier time than the time of other censored disease (Diabetes),
4. one disease (Cancer) censored at an earlier time than the time of the other uncensored disease (Diabetes).

Table 9 shows an example dataset with 4 subjects, representing the above

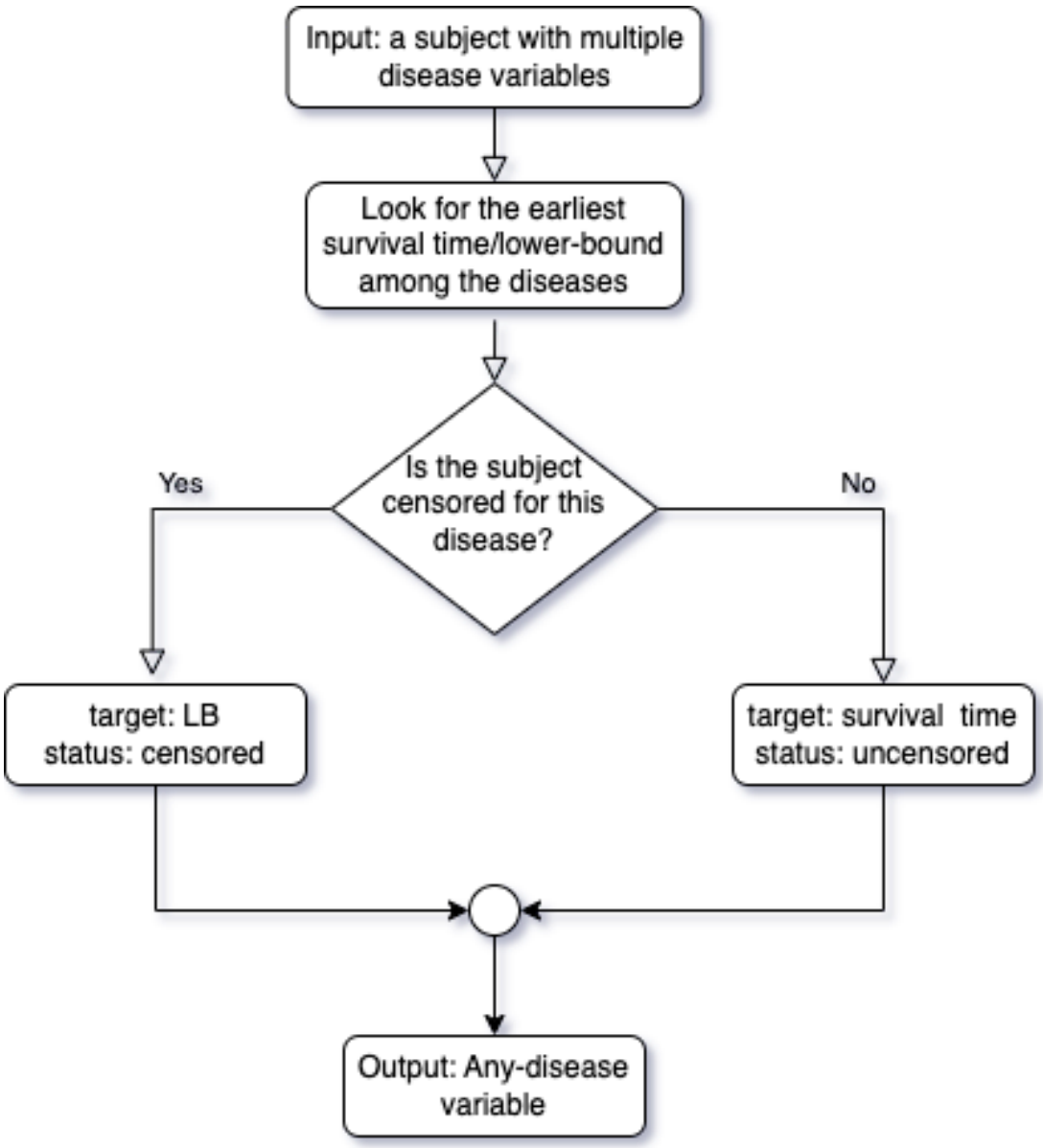


Figure 5: Procedure for engineering the Any-disease target variable

Table 10: the extended example from Table 9 showing the calculation of values of the created Any-disease variables.

Subject	Cancer				Diabetes				Any-disease			
	LB	UB	Uncens.	Time	LB	UB	Uncens.	Time	LB	UB	Uncens.	Time
1	-	-	1	6	-	-	1	9	-	-	1	6
2	6	20	0	6	9	10	0	9	6	10	0	6
3	-	-	1	6	9	10	0	9	-	-	1	6
4	6	20	0	6	-	-	1	9	6	9	0	6

4 combinations of censored and uncensored status for Cancer and Diabetes. In order to explain the procedure for the creation of the Any-disease target variable, the example dataset in Table 9 is expanded in Table 10, which has additional columns reporting the Lower Bound (LB) and Upper Bound (UB) for the value of the target variable (time) of each of the two diseases, as well as new columns with the LB, UB, Uncensored status and value (time) of the created Any-disease target variable.

Recall that, for each subject and base disease (Cancer and Diabetes in this example), the LB and UB variables are defined only when the subject is censored for that disease. If the patient is uncensored for that disease, the target variable (diagnosis time) is completely known, and there is no need to consider LB and UB. When the subject is censored for a given base disease, the LB is the value of the target variable (representing the 'censorship time', i.e. the last date when the subject was observed without a diagnosis of the disease). The computation of the UB for the diagnosis time of a given disease will be described later; for the purposes of the example dataset in Table 10, let us simply assume that the UBs are as given in this Table.

Hence, the computation of the Any-disease target variable is performed as follows, for each of the 4 subjects in Table 10. Subject 1 is uncensored for both base diseases, so LB and UB are not applicable, since we know precisely when the subject was diagnosed with each disease. Hence, the Uncensored status variable for the new Any-disease variable is set to 1 (uncensored subject), and the value

(time) of the Any-disease target is set to the minimum of the target values for the two base diseases, i.e., set to 6 months (minimum of 6 and 9).

Subject 2 is censored for Cancer at time 6 and censored for Diabetes at time 9. Given the LB and UB values, the subject may have been diagnosed with Cancer any time between months 6 and 20, and may have been diagnosed with Diabetes any time between months 9 and 10. Hence, regarding the Any-disease variable, the LB for any diagnosis is set to the minimum of the LBs for Cancer and Diabetes (i.e., 6), the UB is set to the minimum of the UBs for Cancer and Diabetes (i.e. 10), the Uncensored status is set to 0 (censored subject), and the value (time) of the Any-disease target variable is set to 6, which is the LB for having any of the two diseases.

Subject 3 is uncensored for Cancer at time 6 (i.e. her/his cancer diagnosis time is completely known), but censored for Diabetes at time 9, with LB = 9 and UB = 10. Since it is known that the subject was diagnosed with Cancer before the LB for the possible diagnosis of Diabetes, the date of the latter is irrelevant for computing the Any-disease variable. Hence, regarding the Any-disease variable, LB and UB are not applicable, the Uncensored status variable is set to 1 (uncensored subject), and the value (time) of the Any-disease target is set to 6, which is the minimum between the uncensored target value for Cancer and the LB for the censored target value for Diabetes.

Finally, subject 4 is censored for Cancer at time 6, with LB = 6 and UB = 20; and uncensored for Diabetes at time 9. Hence, regarding the Any-disease variable, the LB is set to 6 (the LB for Cancer), the upper bound is set to 9, which is the minimum between the Cancer's UB of 20 and the uncensored Diabetes' time of 9, the Uncensored status variable is set to 0 (censored subject), and the value (time) of the Any-disease target variable is set to 6, which is the LB for having any of the two diseases.

This procedure for computing the Any-disease target variable can be generalized to the case of N base diseases ($N \geq 2$), by generalizing the description of the previous 4 cases of combinations of censored and uncensored statuses, as follows:

Case (1): the subject is uncensored for all N diseases

In this simplest case, when creating the Any-disease target variable, LB and UB are not applicable, the Uncensored status variable is set to 1 (uncensored subject), and the value of the Any-disease target is set to the minimum of the value (time) of the target variable among all N diseases.

Case (2): The subject is censored for all N diseases

In this case, the LB for any disease diagnosis is set to the minimum of the LBs for all N diseases, the UB for any disease diagnosis is set to the minimum of the UBs for all N diseases, the Uncensored status variable is set to 0 (censored subject), and the value (time) of the Any-disease target variable is set to the minimum of the LB among all N diseases.

The next two cases are more complex, and involve cases where the subject is uncensored for some disease(s) and censored for another disease(s). We refer to such cases as 'mixed uncensored/censored statuses' across all diseases. For the purposes of creating the Any-disease target variable, what matters is whether the earliest target variable value (time) among all N diseases is associated with an uncensored or censored status. This leads to the following two cases.

Case (3): The subject has mixed uncensored/censored statuses, and the earliest target variable value (time) among all N diseases corresponds to an uncensored disease $D_{1st-unc}$

In this case, the Any-disease target variable value (time) is simply set to the value of the target variable for $D_{1st-unc}$, which specifies the completely known time

when the subject was diagnosed with any of the N diseases. The diagnosis time or censorship time of the other diseases is irrelevant. There is no need to compute LB and UB for the Any-disease target variable, and the Uncensored status variable is set to 1 (uncensored subject).

Case (4): The subject has mixed uncensored/censored statuses, and the earliest target variable value (time) among all N diseases corresponds to a censored disease $D_{1st-cen}$

In this case, the Any-disease target variable value (time) is set to the value of the target variable for D1st-Cen, which specifies the lower bound for the time of diagnosis for that disease, which is also, of course, the lower bound for any disease diagnosis. The UB for any disease diagnosis is the minimum among the UB of all censored diseases and the completely known target variable value of all uncensored diseases. The Uncensored status variable is set to 1 (censored subject).

Therefore, in the case of censored subjects, the lower-bound of the Any-disease simply took the smallest value among those of diseases, whilst the upper-bound required a more complex calculation. To clarify how the above procedure worked, we take Table 10 as an example where subjects 1 and 3 were uncensored and their lower-bound and upper-bound were non-applicable (N/A). With regard to subjects 2 and 4, the censored subjects, both took the lower-bound from that of Cancer (6 months) due to it being smaller than that of Diabetes (9 months). Next, subject 2 took the upper-bound from that of Diabetes (10 months) for the same reason, whereas subject 4 had to compare between the smallest upper-bound (20) and the smallest uncensored value of the target variable (9 from that of Diabetes). Hence, subject 2 took the survival time for Diabetes as its upper-bound.

Creating predictive features based mainly on the Nurse data

The methodology used to create predictive features described in this section is heavily based on the methodology described in more detail in (Pomsuwan, 2017), since the features are similar - although the target variables being predicted in this thesis are very different from the target variables predicted in that work (which addresses the classification task of machine learning, predicting nominal values of a class variable).

In the created datasets, most features were created from raw variables available in the Nurse Visit data, which is part of the previously discussed ELSA database (Clemens et al., 2019). Those raw variables represent several types of biomedical information collected by a nurse, including for instance many types of blood sample tests. In addition, the nurse took several physical performance measurements that involved asking a patient to move his/her body in different ways. If a particular movement could not be done by the participant or he/she felt that it was unsafe to try to do it, the attempt was marked as 'Not attempted' or 'Test not completed'. These features are then used to predict the target variables representing the time at first diagnosis of age-related diseases. Although the Nurse variables are available at ELSA waves 2, 4, 6 and 8, our created datasets contain only features for wave 2. The main reason is that many subjects were diagnosed with one of the diseases of interest in wave 3, and using the predictive features in wave 4 or 6 for predicting such diagnoses would lead to "predicting the past", which is not useful in practice.

As mentioned earlier, the raw biomedical variables collected by the nurses were not collected specifically for machine learning, and they contain a large amount of obviously redundant or irrelevant information. Hence, we have created predictive features (for our task of predicting the time passed until the diagnosis of some age-related disease) by extracting and combining information from the raw variables in the Nurse data files, as follows. First of all, we kept potentially predictive

variables from the Nurse data, whilst many other variables which are intuitively useless for predicting age-related diseases were removed because such variables were collected mainly to record problems in data collection for other variables. For example, several variables capturing information such as the reasons why taking a blood sample test was refused by a patient, and information about several types of problems in some physical performance measurements, were discarded.

In addition, many variables in the Nurse data represented clearly redundant information, in cases where the same variable (e.g. the result of a blood test) was measured three different times in the same wave, in order to represent the variability in test results. This resulted in the duplication of variables representing the same biomedical property in each wave, and none of those three measures can be considered 'better' than the other two. Hence, instead of using any of the three underlying variables, we created a feature defined as the mean value over those three measures, for each individual (instance), for each wave.

Another point to consider is the occurrence of different types of missing values in many raw variables in the Nurse data, which were originally labelled as different negative values, as follows (using as an example a blood test result variable):

- 1 = Not applicable
- 6 = Period between collection and receipt in the lab ¿ 5 days
- 8 = Don't know
- 9 = Refusal
- 11 = Blood sample not taken

Considering all these types of missing values separately would considerably complicate the task of the algorithms for predicting the time passed until the diagnosis of some age-related disease. Hence, to simplify, all these different negative values were assumed to have the same meaning of "missing value", so we treated them in the same way by replacing all of them with the special missing value symbol "?".

Besides the features created from the raw variables in the Nurse data files, we also included in our datasets two features directly extracted from the Core files in ELSA which intuitively represent potentially very relevant information for predicting age-related diseases, namely the features "w2indager" (age) and "indsex" (gender).

Finally, an important point is that, when creating the instances used in our datasets, only data from "core" members were used, so the ELSA records of their partners were ignored. The ELSA variable "idauniq", which is a unique id for each individual, was added to our datasets to match up data about the same core member in different dataset files (across different waves). This variable was not used for prediction purposes, of course, since it has no predictive power. Note that an instance was created for an individual only if that individual was not diagnosed with the disease of interest before wave 3 and participated in the ELSA study at least up to wave 3, so that the target variable's values are available for all individuals (instances) in the created datasets.

3.3 The SHARE dataset

The Survey of Health, Aging, and Retirement in Europe (SHARE) (Börsch-Supan et al., 2013) is a longitudinal study conducted across many European countries. SHARE aims to investigate the health, socio-economic, and demographic aspects of individuals aged 50 and older.

Similar to ELSA, SHARE collects data through waves, with each 'wave' representing a specific time point of data collection. Each wave reports the changing circumstances of the participants as they age. In addition, SHARE incorporates health modules in its surveys to capture detailed health-related information. These modules cover topics such as self-reported health, chronic diseases, functional limitations, cognitive functioning, and mental health. These data provide

insights into the health profiles by older individuals in Europe, and thus they are used as the feature set for the SHARE dataset.

In terms of participants, the SHARE database is significantly larger than the ELSA database. SHARE initially included around 30,000 individuals aged 50 and older from 11 European countries in its baseline sample. Over time, additional countries have been included, further increasing the sample size to over 100,000 participants. As a result, while both databases provide valuable insights into aging and related factors, SHARE's larger sample size allows for more robust analyses and provides a broader representation of the older adult population in Europe.

Unlike the ELSA data, we only created the Any-disease target variable for the SHARE dataset, with some slight differences in its definition. First, instead of the target variable representing the number of months passed (for ELSA data), the Any-disease target variable for the SHARE data represents the number of "waves" passed until the first diagnosis for any of the given diseases. This is mainly due to the variable availability in the SHARE database as will be explained next. Second, there were 11 chronic diseases or medical conditions involved in the creation of the SHARE's target variable, instead of 8 diseases like in the ELSA data. Specifically, these 11 diseases are:

1. A heart attack
2. High blood pressure or hypertension
3. High blood cholesterol
4. A stroke or cerebral vascular disease
5. Diabetes or high blood sugar
6. Chronic lung disease
7. Cancer or malignant tumor
8. Stomach or duodenal ulcer, peptic ulcer
9. Parkinson disease
10. Cataracts

Table 11: Example dataset with censored and uncensored subjects

Subject	wave	wavepart	chronic_mod
1	1	1,2	0
1	2	1,2	1
2	4	4,5,6	2
2	5	4,5,6	2
2	6	4,5,6	2
3	4	4,5,6,7	0
3	5	4,5,6,7	0
3	6	4,5,6,7	1
3	7	4,5,6,7	0
4	1	1,2,3	0
4	2	1,2,3	0
4	3	1,2,3	0
5	7	7	0

11. Hip fracture or femoral fracture

To explain the computation of the value of the Any-disease target variable, we use an example dataset, as shown in Table 11, which contains 5 unique instances (subjects). In this table, the "wavepart" column shows the numbers of the waves where the subject participated, and the "chronic_mod" column shows the number of chronic diseases a subject was diagnosed with at each wave (indicated in the second column). In the SHARE survey data, the data were stored in "long format", where data of the same subject were recorded in multiple rows and each row contained information for one wave. Therefore, any variables that did not change across time (i.e. across different waves), such as the subject's date of birth and gender, would have the same value in all the rows for a given subject in the SHARE file.

Although SHARE was a longitudinal study, we used cross-sectional features (predictor variables) in this project. Specifically, the features for each subject were derived from the data at the first wave of that subject's participation, i.e., the wave when he or she joined the SHARE survey. For example, the features

used for subjects 1 and 4 were from wave 1 and the features for subjects 2 and 3 were from wave 4.

In order to create the Any-disease target variable, we used multiple values of chronic_mod variable, corresponding to the same subject, across the waves. Recall that the Any-disease target variable represents the number of waves passed until the first diagnosis of any of the above 11 diseases. Hence, the value of the target variable for each subject is set to the number of the wave at which the value of chronic_mod was greater than 0 for the first time for that subject.

To explain this procedure, consider Table 12 as an example of how the target variable was created based on the dataset shown in Table 11. First, subject 1 joined the SHARE survey in wave 1 and then was diagnosed with a disease in wave 2 (Table 11). This means that subject 1 "survived" (i.e. was not diagnosed with any disease) for 1 wave time, so that the target variable Any-disease took the value of 1 (Table 12). For subjects 2 and 3, their targets were 0 and 2, respectively. Since these three subjects developed at least one disease, their uncensored status variable was marked as 1 (uncensored). Moving on to subjects 4 and 5, who are censored subjects, we created two special variables: the lower-bound and upper-bound, to describe all the possible values that the true target variable Any-disease could take. As shown in Table 11, subject 4 joined the survey in wave 1 and dropped out after wave 3, meaning the survival time was at least 3. The upper-bound was set at the end of wave 7, the last wave of the study, so if the subject survived throughout the study, then he/she survived for at least 7 waves. Next, subject 5 participated in wave 7 only and no disease-diagnosis event occurred, so both the lower-bound and upper-bound took the value of 1 for that subject.

Table 12: An example of how the Any-disease target variable was created based on the example dataset in Table 11. LB and UB stand for the lower bound and the upper bound of the target variable, and Uncens is the uncensored status variable.

Subject	Any-disease			
	LB	UB	Uncens.	Target
1	-	-	1	1
2	-	-	1	0
3	-	-	1	2
4	3	7	0	3
5	1	1	0	1

3.4 The Haemodialysis dataset

We used raw variables available in different data files to create the feature set for the Hemodialysis dataset. Those raw variables represent several types of biomedical information similar to that of the ELSA data collected by a nurse, including for instance many types of blood sample tests. These features are then used to predict the target variable representing the time at which the patient died from kidney disease.

Similar to many other medical-survey databases, the Hemodialysis database contains a few variables which are obviously redundant or irrelevant information. Hence, we have created predictive features (for our task of predicting the time passed until the patient died) by extracting and combining information from the raw variables in the data files, as follows. First, we kept potentially predictive variables from the data, whereas some other variables which are present to join multiple data files together, such as Episode ID, were discarded.

Furthermore, there is another type of non-informative feature in the Haemodialysis dataset where the variables contain a large number of patients (instances) with missing values as well as a large number of instances with the value 0, which were wrongly used to represent a “missing value”. Again, these variables are discarded.

As a result, the following lists of variables have been finalised from multiple

data files.

From the file ExpPats:

- AgeFD: Age at first dialysis
- DxCat: Dialysis category
- TxpStat: Transplant status
- Diabetes: has diabetes (yes/no) (Note: missing values will be interpreted as "no")
- RDE1: Diagnosis list 1
- FallsRisk

From the file ExpDialysis:

- PatStatusType: patient status type, at the time of first dialysis (baseline value)
- PatStatusDesc: patient status description, at the time of first dialysis (baseline value)
- PostSyBP: post-dialysis systolic blood pressure, post the first dialysis (baseline value)

From the file ExpKtV:

(again, values at the time of first dialysis, baseline value)

- UreaDiff: difference between first and second urea tests (using this instead of 1st, 2nd urea values)
- URR: calculated URR
- SimpleKTV: calculated KtV

From the file ExpMeasures:

- BMI: body mass index (kg / m2)

- estMetabolicRate: estimated metabolic rate
- GripStrength: grip strength
- MAC: Mid-arm circumference
- MAMC: Mid-arm muscle circumference
- TSF: Triceps skinfold thickness

From the file ExpPatCharlson:

- CharlsonScore: Charlson score, an index of co-morbidity

Encoding of categorical variables

Another common data cleansing method that often needs to be applied to real-world data is one-hot encoding. The Haemodialysis dataset contains many categorical (nominal, non-numeric) variables, each typically with a large number of distinct values (often tens of values), such as the variable RDE1 - describing a diagnosis list, containing 80 distinct categorical values. By using the one-hot encoding technique, each of these variables is transformed into a set of binary variables, each one indicating whether or not (1 or 0) the patient has the corresponding value of the original categorical variable. This leads to a large number of new binary variables. Note that some of these new binary variables contain the value 0 in nearly all instances because the corresponding categorical value has a very small frequency in the original dataset. Therefore, these binary variables, i.e. rare categorical values, are discarded from the dataset. Note that a 'rare' value is defined as a frequency below the threshold of 10.

Creating the target variable for the Haemodialysis dataset

In the Haemodialysis dataset, the "target" variable, the variable to be predicted, represents the time passed until the patient's death from kidney disease (measured in the number of days), or the survival time, which is defined as follows.

$$SurvivalTime = EndDate - StartDate,$$

where Start Date is the date of the first dialysis and the End Date is the lower value (earlier date) between the date of death and the study end date (31/12/2019).

In the Haemodialysis survey (whose data was used for creating the Haemodialysis dataset used in this thesis' experiments), patients can be categorised into five types depending on the time period of their participation in this survey, as follows.

(1) The patient started dialysis before the start of the study, has been on dialysis within the study for at least 2 years, and died within the study period. This type of patient has no records of their health status before starting the dialysis, thus, they will be excluded from this thesis' experiments.

(2) The patient started dialysis after the start of the study, has been on dialysis within the study for at least 2 years, remained in the study until its end, and died after the end of the study. In this case, the End Date is the end of the study, the patient is censored, and this patient will be used in this thesis' experiments.

(3) The patient started dialysis after the start of the study, has been on dialysis within the study for at least 2 years, and died within the study period. In this case, the End Date is the date of death, the patient is uncensored, and this patient will be used in this thesis' experiments.

(4) The patient started dialysis after the start of the study, has been on dialysis within the study for less than 2 years, and died within the study period. In this case, the End Date is the date of death, the patient is uncensored, and this patient will be used in this thesis' experiments.

(5) The patient started dialysis after the start of the study, has been on dialysis within the study for less than 2 years, and died after the end of the study. This patient will not be used in this thesis' experiments, as decided in discussions with a medical expert on the data.

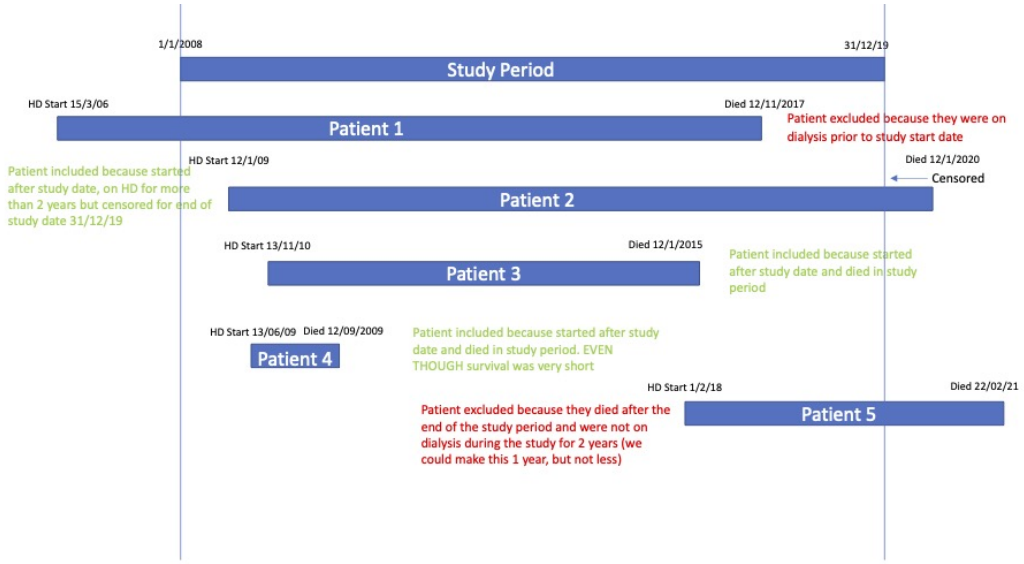


Figure 6: An example of the different types of patients in the Haemodialysis database.

In summary, only three out of five types of patients are included in the dataset created for the machine learning (survival analysis) experiments performed in this thesis, namely Patient types 2, 3 and 4. Type 2 patients are censored (did not die before the end of the study) and, thus, their survival times are a lower-bound for the unknown survival time. Types 3 and 4 patients are uncensored and their survival times are defined as the number of days from their first dialysis to death.

3.5 Computation of lower and upper bounds for the target variable

This section will describe how to compute the lower and upper bounds for the value of the target variable in the case of censored subjects. These bounds will be used as part of the specification of the new variations of random forests for survival data proposed in the next two sections.

Recall that the main challenge of survival analysis is to cope with censoring in the target variable. This thesis focuses on right-censoring (as opposed to left-censoring) (Kleinbaum and Klein, 2012), which is common in medical research and occurs very often in our datasets (described in the previous chapter). Right-censoring occurs when the subject dropped out of the study before its end and no event of interest occurred before the dropout, or when the study ends before the event of interest occurred for a subject. Note that in right-censoring the last observed time for a censored subject is a lower-bound for the unknown event occurrence time.

The target value's upper-bound for each censored subject (i.e. each instance in the dataset) is computed as the number of months passed between the date of the nurse visit to the patient when the patient joined the study (*nurse_year* and *nurse_month*) and the end of the study, which is wave 8 (last wave) for the datasets derived from the ELSA database and wave 7 for the SHARE dataset.

In contrast with the ELSA and SHARE datasets, the Haemodialysis dataset does not have a nurse visit date or an end-of-study date. Therefore, in the Haemodialysis dataset, we have simply taken the longest survival time of the censored patient as the upper-bound for every other patient in the dataset.

3.6 Conclusion

This chapter has described in detail the creation of 11 survival analysis datasets, which will be used in the experiments reported in the next three chapters. In 10 of these datasets (derived from data in the ELSA and SHARE studies) the target variable to be predicted is the time passed until an individual is diagnosed with some age-related disease, whilst in the haemodialysis dataset the target variable is the time passed until the death of an individual. Table 13 provides a summary of the main characteristics of the created datasets; namely the number of instances

Table 13: Main characteristics of the datasets used in the experiments

Dataset Name	**Data Source**	**#Instances**	**#Features** (Num.,Cat.)	**Uncens. Ratio**
Alzheimer	ELSA	6825	44 (27,17)	1.0%
Angina	ELSA	6488	44 (27,17)	2.5%
HeartAtt	ELSA	6607	44 (27,17)	2.8%
Psychiatric	ELSA	5972	44 (27,17)	3.5%
Stroke	ELSA	6632	44 (27,17)	4.1%
Diabetes	ELSA	6500	44 (27,17)	6.4%
Cancer	ELSA	6386	44 (27,17)	8.8%
Arthritis	ELSA	4276	44 (27,17)	18.3%
Any-disease (ELSA)	ELSA	3280	44 (27,17)	29.8%
Any-disease (SHARE)	SHARE	139522	15 (4,11)	72.6%
Haemodialysis	n/a	1097	38 (27,11)	71.4%

(individuals); number of features, also reporting the number of numerical and categorical features; and the proportion of uncensored instances (uncensored ratio) in each dataset.

Chapter 4

New Variants of Random Forests for Survival Data based on the Imputation of Censored Target Variables

4.1 Introduction

This chapter will explore the potential of using imputation techniques to address censoring issues. This approach involves substituting the censored target value with an estimate of an uncensored value. This modification allows for the use of all other components of the RF algorithm without any alterations. This chapter proposes two novel RF algorithm variants for this data transformation. Both variants involve the basic idea of imputing the value of censored target variables in the data used for learning each tree in the random forest. It will also report the results of computational experiments evaluating the proposed RF variants.

This chapter is organised as follows. Section 4.2 explains the Random Target-Imputation Forests, the variant that generates uncensored target values randomly

within certain lower and upper bounds. Section 4.3 describes the K-Nearest Neighbours-based Imputation for Random Forests, the variant that uses the K-Nearest Neighbour (KNN) algorithm to estimate the uncensored target value of each censored instance. Section 4.4 describes the experimental methodology. Section 4.5 reports experimental results, including both predictive accuracy results and an analyis of the most important predictive features in the best models learned in our experiments, for our age-related datasets.

4.2 Random Target-Imputation Forests (RTIF)

This section will describe the proposed RTIF method, a variant of random forests where censored values of the target variable are randomly imputed (within certain bounds) in the training data, before learning each tree in the forest. RTIF is designed specifically to solve the censoring issue in survival analysis.

More precisely, the method is based on the idea of imputing the value of the target variable of each censored instance, based on lower and upper bounds for that instance's target value. These lower and upper bounds are calculated as explained in the previous section. This means that an imputed value of a censored instance is a uniformly random value between its lower-bound and upper-bound.

In addition, in order to increase the diversity of the trees in the forest, this target-variable imputation process is applied independently for every bootstrap training set, as shown in Figure 7. Therefore, the same censored instance may contain different imputed target values in different bootstrap samples.

Note that this imputation allows all other procedures of the Random Forest algorithm to be used without modification.

The pseudo-code of the RTIF method is shown in Algorithm 2, where $input\_target_i$, the i-th censored instance's target value (survival time), is replaced by a randomly generated value between LB_i and UB_i, which are the i-th censored instance's lower

bound and upper bound, respectively.

Algorithm 2: RTIF.

```
for each Bootstrap Sample do
    for each censored instance i in the current Bootstrap Sample do
        input_target_i ← random(LB_i, UB_i)
    end
end
```

4.3 K-Nearest Neighbours-based Imputation for Random Forests (KNN-RF)

This section will describe in detail the KNN-RF method, where a variant of the K-NN method is used for the imputation of censored values of the target variable in the training data, before learning each tree in the forest.

The K-Nearest Neighbours Random-Forests (KNN-RF) method performs a more sophisticated imputation of censored values than the previously described RTIF method. More precisely, KNN-RF replaces the random generation of target variable values with a deterministic imputation method based on the actual target values from the uncensored subjects which are the nearest neighbours of the current censored subject (whose target variable value needs to be imputed). Furthermore, in order to identify the nearest neighbours, the KNN-RF method looks for uncensored subjects with the most similar age to the age of the current censored subject. Then, the mean value of the target variable among all those uncensored neighbours is used as the imputed value of the target variable for the current censored subject.

In order to find the nearest uncensored neighbours for a censored subject based on age, we propose the pseudo-code shown in Algorithm 3, which is based on the following notation.

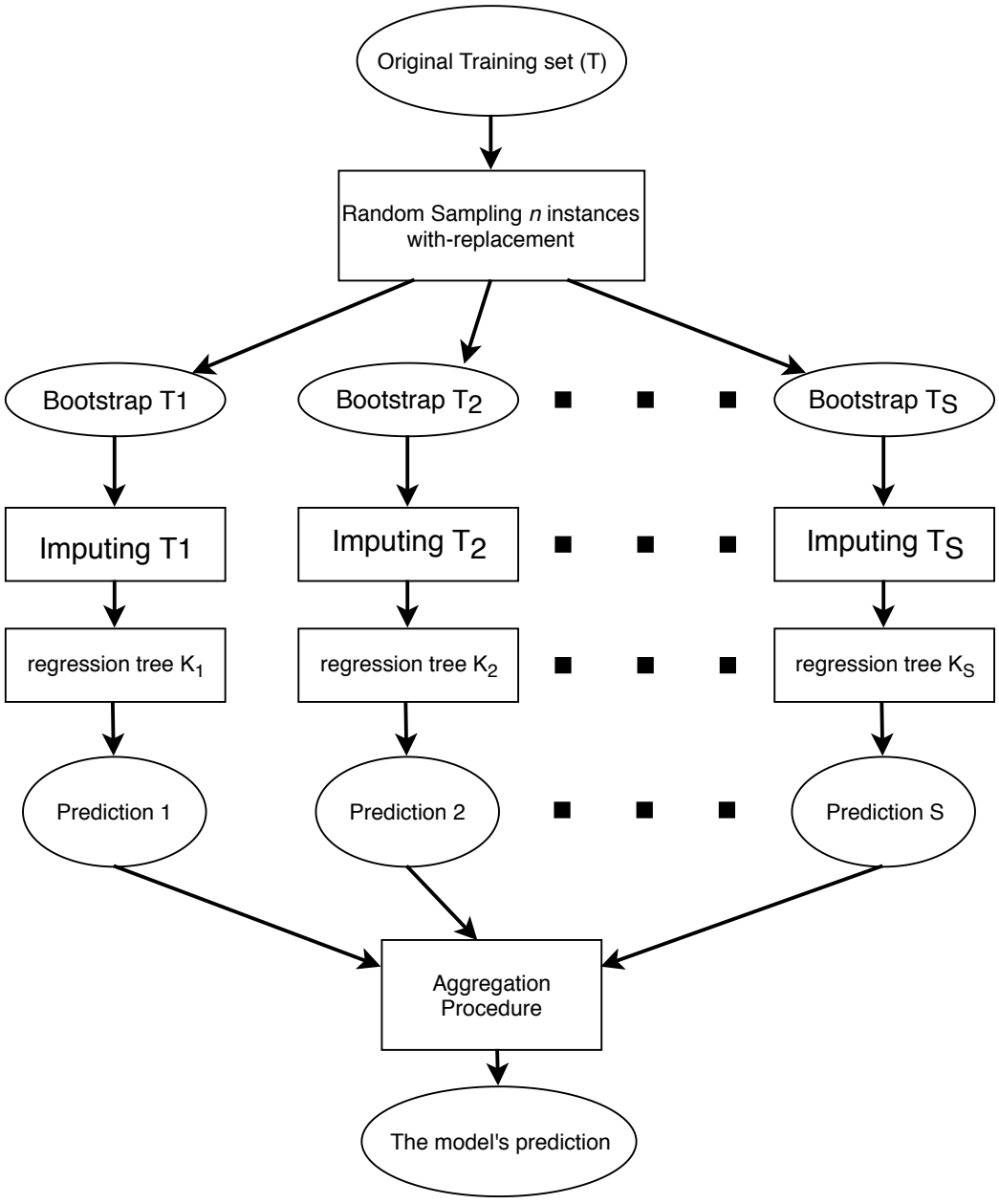


Figure 7: Overview of the proposed Random Target-Imputation Forest (RTIF) method, a variation of random forests for survival data analysis

Notation used in Algorithm 3

- (x_i, y_i) = feature value vector x_i and target variable value y_i of the i-th instance
- $x_i.age$ = age value of the i-th instance
- $x_i.uncens$ = uncensored status of the i-th instance (1 = uncensored, 0 = censored)
- Nb_i = valid neighbours of the i-th instance: neighbours which are uncensored and have target value within the lower and upper bounds of the i-th instance (LB_i and UB_i)
- $Hi\_Age\_Nb_i$ = valid neighbours with higher age as close as possible to the i-th instance's age
- $Lo\_Age\_Nb_i$ = valid neighbours with lower age as close as possible to the i-th instance's age
- Min_age = minimum age value among all uncensored instances in the training set
- Max_age = maximum age value among all uncensored instances in the training set
- LB_i = lower bound for the target value of the i-th instance
- UB_i = upper bound for the target value of the i-th instance
- $Default_i$ = the default imputation value of the i-th instance using the middle value between LB_i and UB_i

There are three possibilities for computing the neighbours of the censored subject in Algorithm 3, as follows.

First, the algorithm looks for valid neighbours of the same age as the censored subject. If such neighbour(s) is(are) found, the algorithm imputes the target value of the current censored subject with the median of target values among valid neighbours. This is implemented in lines 6–8 of Algorithm 3. As shown in Line 6, a training instance j is a valid neighbour of the current instance i (whose

target value is being imputed) if instance j is uncensored and j's target value is within the lower and upper bounds of instance i.

Second, if there are no such neighbours, the algorithm looks for valid neighbours with ages as close as possible to the censored subject's age. If such valid neighbour(s) is(are) found, the algorithm imputes the target value in the same way as described for the above first case. This is implemented in lines 10–24 of Algorithm 3. These lines implement a loop which increases the "age gap" by one unit at each iteration of the loop. Hence, at each iteration, the algorithm produces two new candidate ages to try to find valid neighbours, by adding and subtracting the age gap to and from the age of the current censored subject. This process of increasing the age gap and looking for valid neighbours terminates when some valid neighbours are found or when both adding and subtracting the age gap produces new candidate ages which are greater than the maximum and lower than the minimum age, respectively – where the maximum and minimum ages are identified among all uncensored instances in the training set.

Finally, as in the third case, if there are no valid neighbours at all, the algorithm uses a default imputation value generated by using the middle value between LB_i and UB_i. This is implemented in lines 26–28 of Algorithm 3.

4.4 Experimental Methodology

This section is divided into four parts: the first subsection provides a summarised description of the datasets used in the experiments; the second subsection mentions the predictive performance measure used; the third subsection describes the hyper-parameter tuning procedure, based on nested cross-validation; and the fourth subsection describes the statistical significance tests used to analyse the results.

Algorithm 3: KNN-RF.

$N\_median\_sameAge\_input \leftarrow 0$
$N\_median\_diffAge\_input \leftarrow 0$
$N\_mean\_bounds\_input \leftarrow 0$
for *each* *Bootstrap Sample* **do**
  for *each* *censored instance i in the current Bootstrap Sample* **do**
    $Nb_i \leftarrow \{ (x_k, y_k) \mid x_k.uncens = 1, x_k age = x_j.age, LB_i \leq y_k \leq UB_i \}$
    if $Nb_i \neq \emptyset$ **then**
      $y_i \leftarrow$ median of y in Nb_i
      $N\_median\_sameAge\_input \leftarrow N\_median\_sameAge\_input + 1$
    else
      $Age\_gap \leftarrow 1$
      repeat
        if $x_i.age + Age\_gap \leq Max\_age$ **then**
          $Hi\_Age\_Nb_i \leftarrow \{(x_k, y_k) \mid x_k.uncens = 1, x_k.age + Age\_gap = x_i.age, LB_i \leq y_k \leq UB_i\}$
        end
        if $x_i.age - Age\_gap \geq Min\_age$ **then**
          $Lo\_Age\_Nb_i \leftarrow \{(x_k, y_k) \mid x_k.uncens = 1, x_k.age - Age\_gap = x_i.age, LB_i \leq y_k \leq UB_i\}$
        end
        $Nb_i \leftarrow Hi\_Age\_Nbi \cup Lo\_Age\_Nbi$
        if $Nb_i \neq \emptyset$ **then**
          $y_i \leftarrow$ median of y in Nb_i
          $N\_median\_diffAge\_input \leftarrow N\_median\_diffAge\_input + 1$
          $Age\_gap \leftarrow Age\_gap + 1$
        end
      until $Nb_i \neq \emptyset$ *OR* $((x_i.age + Age\_gap > Max\_age)$ *AND* $(x_i.age - Age\_gap < Min\_age))$
    end
    if $Nb_i \neq \emptyset$ **then**
      $y_i \leftarrow Default_i = \frac{LB_i + UB_i}{2}$
      $N\_mean\_bounds\_input \leftarrow N\_mean\_bounds\_input + 1$
    end
  end
end

4.4.1 Datasets used in the experiments

The experiments used 11 datasets for 11 different age-related diseases (i.e. 11 separate survival prediction problems). 10 of these 11 datasets were created from two different surveys, ELSA (English Longitudinal Study of Ageing) (Clemens et al., 2019) and the Survey of Health Ageing and Retirement in Europe (SHARE) (Börsch-Supan et al., 2013). The other dataset involves predicting the survival time for haemodialysis patients. The creation of these datasets was described in detail in Chapter 3.

More precisely, 9 out of the 11 datasets were constructed from the ELSA data, containing between 3,000 and 7,000 instances (depending on the target variable), with exactly the same 44 predictive features, but different target variables. On the other hand, the dataset constructed from the SHARE data is much larger, containing almost 140,000 instances but only 15 predictive features. In essence, the instances represent individuals (subjects) in these surveys, and the target variables represent the 'survival times', more precisely, the time passed (in months) until an individual is diagnosed with a certain disease (for 8 datasets) or any of several diseases (for two datasets); whilst the predictive features represent biomedical information collected by nurses or other relevant characteristics of an individual (age and gender). The haemodialysis dataset has 1,097 instances with 38 features, and in this dataset the target variable represents the survival time of patients undergoing haemodialysis.

In general, the ELSA datasets have a small proportion of uncensored instances (i.e. the large majority of their instances are censored). By contrast, the SHARE dataset and the Haemodialysis dataset have a large proportion of uncensored instances.

4.4.2 Predictive Performance Measure

In the experiments being reported in this chapter, the predictive performance of the learned survival models was estimated by the concordance index (C-index), which is a measure accounting for censored data, and is probably the most used measure of performance for survival prediction tasks. Recall that the C-index can be interpreted as the probability of correctly ordering the predicted survival times for a randomly chosen pair of subjects whose actual survival times are different. For a precise formal definition of the C-index, the reader is referred to Section 2.2.5 in Chapter 2.

4.4.3 Hyper-parameter tuning with nested cross-validation

All experiments were performed using nested cross-validation, where 5-fold inner cross-validation performs hyper-parameter tuning and 10-fold outer cross-validation estimates predictive performance. That is, for each of the 10 training sets of the outer cross-validation, each candidate configuration (combination of hyper-parameter settings) of the algorithm is evaluated via a 5-fold inner cross-validation applied to that training set only — without using the corresponding test set. Hence, in each run of the algorithm during the 5-fold inner cross-validation, the algorithm is trained on 80% of the training data and evaluated on the remaining 20% of the training set, called the validation set. The algorithm configuration with the highest average C-index value over the 5 validation sets of that inner cross-validation is chosen as the best configuration for the current training set. Then, the algorithm is re-trained, with that configuration, on the entire training set, and the learned model is evaluated on the current test set of the outer cross-validation. The result returned by the nested cross-validation is the average C-index value computed over the 10 test sets of the outer cross-validation, as usual.

Regarding the Random Forest hyper-parameters to be optimised by the inner

cross-validation, first, we set the number of trees in the forests to a constant of 500 as suggested by (Probst, Wright and Boulesteix, 2019). Then, in order to optimise the performance of the Random Forests algorithm, we selected two more important hyper-parameters as suggested by (Lin and Jeon, 2006) and (Lynch et al., 2017): the minimum node size (the minimum number of instances allowed at leaf nodes) and mtry (the number of randomly sampled candidate features at each tree node) for all experiments performed in this chapter.

We used 5 folds for the inner cross-validation for all datasets, whilst the number of folds for the outer cross-validation was set to 10 for the ELSA datasets and the haemodialysis dataset and set to 5 for the SHARE dataset. The SHARE dataset has fewer folds to save computational time since it is much larger than the other datasets.

The tuning procedure tried all possible combinations of 3 minimum node size values (5, 7 and 10) for all datasets. However, we consider a different set of candidate *mtry* values for the ELSA datasets and SHARE dataset separately, since there is a difference between their numbers of features, where the former contains 44 and the latter contains 15 predictive features. For ELSA, we specify four candidate values for *mtry* (4, 7, 10, 13). The first two values were calculated as $ceil(ln(44)) = 4$ and $ceil(sqrt(44)) = 7$, where the natural logarithm (ln) and the square root (sqrt) are often considered default functions for specifying the value of mtry in random forests, and *ceil(x)* returns the 'ceiling' of x, i.e. the lowest integer that is greater than or equal to x (i.e. it rounds x up to the nearest integer). Similarly, the set of candidate values for SHARE is (3, 4, 6, 8), where the first two values were calculated as $ceil(ln(15)) = 3$ and $ceil(sqrt(15)) = 4$. Third, I use mtry = 5, 6, 8, 10 for the Haemodialysis dataset. Therefore, for all methods and each dataset, at each iteration of the outer cross-validation, the inner cross-validation is run 12 times on the training set, considering 12 candidate random survival forest configurations (4 candidate *mtry* values times 3 candidate

node size values).

4.4.4 Statistical significance tests

In this section, all statistical significance tests used to analyse our experimental results are described. Besides average ranking and win count, statistical tests are helpful to assure whether or not the performance gaps among the supervised models are significant. In statistical words, we attempt to strengthen our confidence as well as deny the claim that an improvement in the observed results is a coincidence. We focus on pairwise comparisons for machine learning models.

In order to evaluate whether the null hypothesis of the test may be rejected or not, statisticians generally use p value as a measure of probability to compare against a significance level called α, which is a pre-defined parameter: 5% is the most commonly used value. In the context of machine learning, a null hypothesis (H_0) can be defined as all of the models being equally accurate in terms of predictive performance. As such, we can assert an alternative hypothesis (H_1 or H_A) that there is a significant difference among the models' performances at $1-\alpha$ confidence (Demšar, 2006).

Friedman's test

Friedman's test is a rank-based non-parametric test for determining whether or not there are significant differences in the performance of multiple supervised machine learning models across multiple datasets (Friedman, 1940). Non-parametric means that the test makes no assumption about the dataset having a particular distribution, e.g., the normal distribution. The null hypothesis for the test is that all the prediction models have identical predictive accuracy. The alternative hypothesis is that the prediction models have different predictive accuracy.

Friedman's test involves several steps (Demšar, 2006), which can be summarised as follows. First of all, it ranks the predictive accuracy values of the

models being compared for each dataset (row) separately. That is, the model with the highest predictive accuracy is assigned a rank of 1. In the case of a tie, the corresponding average rank is assigned to the tied models. Afterwards, the ranks are averaged for each survival method. The next step is to calculate the Friedman's test statistic (χ_F^2) as follows:

$$\chi_F^2 = \frac{12N}{k(k+1)} \left[\sum_j R_j^2 - \frac{k(k+1)^2}{4} \right] \tag{27}$$

where N is the number of datasets, k is the number of methods and R_j is the average ranks of the models being compared. However, as pointed out in (Iman and Davenport, 1980), a better statistic can be derived, as shown in equation 28,

$$F_F = \frac{(N-1)\chi_F^2}{N(k-1) - \chi_F^2} \tag{28}$$

which is distributed according to the F distribution with $(k-1)$ and $(k-1)\times(N-1)$ degrees of freedom and F_F is the Friedman's test value.

After that, there are two equivalent ways to determine the test result. First, compute the critical value based on the given degrees of freedom; if the Friedman's test value is greater than the critical value, then the null hypothesis can be rejected. Second, convert the Friedman's test value into a p value; if the p value is smaller than the significance level α, then the null hypothesis can be rejected. Otherwise, the null hypothesis cannot be rejected, and thus the comparison is deemed non-significant.

If the null hypothesis is rejected, which means that there is a significant difference in predictive performance among the different methods, we need to apply a post-hoc test to determine which pairs of methods have significantly different performances. The post-hoc test used in this thesis, the Holm procedure, is described next.

Holm Procedure

The Holm procedure is a post-hoc and non-parametric test for multiple comparisons — i.e., determining whether or not there are significant differences in the predictive performance of the best (control) method against each of the other methods (Holm, 1979). Specifically, a post-hoc test is applied if and only if a pre-hoc test (like the Friedman's test) has determined that there is a significant difference in the predictive performances of several methods. In addition, the Holm procedure is a multiple-hypothesis test where the number of null hypotheses is the same as the number of method comparisons, which is the number of methods against which the best (control) method is compared. That is, this procedure compares the best method against each other i-th method, $i = 1,...,k-1$, where k is the number of methods.

Let p_i denote the p-value for the comparison between the control method and the i-th method. In order to compensate for multiple comparisons, the Holm procedure compares each p_i value against an adjusted significance level α_i based on a pre-defined target significance level, denoted $target\_\alpha$ (0.05 in general), as follows.

$$\alpha_i = \frac{target\_\alpha}{k-i} \tag{29}$$

The test result of each comparison is determined by the p_i value derived from the z_i score, computed as shown in equation 30:

$$z_i = \frac{R_i - R_0}{SE} \tag{30}$$

where R_0 and R_i are the average ranks of the control method and the compared method, respectively; and SE is the standard error which can be computed from the following equation 31,

Table 14: Results of the calculations performed by the post-hoc Holm procedure, for the example scenario discussed in the main text

i	method	$z_i = \frac{R_i - R_0}{SE}$	p_i	α_i
1	B	$\frac{2.6-1.4}{0.426} = 2.683$	0.0037	0.025
2	C	$\frac{2.0-1.4}{0.426} = 1.342$	0.0899	0.050

$$SE = \sqrt{\frac{k(k+1)}{6N}} \tag{31}$$

where N is the number of datasets, k is the number of methods.

As such, the Holm procedure decides whether to reject the null hypotheses in a step-wise fashion, starting with the smallest (most significant) p value (Demšar, 2006). If that p_i value is larger than the α_i, then the corresponding null hypothesis cannot be rejected. Therefore, the test result concludes a non-significant difference, and then the whole procedure terminates, and all the other comparisons are also deemed non-significant. However, if that p_i value is smaller than the adjusted α_i, then the corresponding null hypothesis is rejected. Then, the test moves to the next null hypothesis (the second smallest p-value) and repeats the process.

For example, consider the case where three different methods are compared in an experiment with 11 datasets (which is the case in some experiments reported later in this thesis). Suppose that methods A, B and C obtained the average ranks of 1.4, 2.6 and 2.0, respectively. Since method A is the best among the three based on the average rank, it is selected as the control method, and it will be compared against the other two methods, B and C, using the Holm procedure to adjust the α values for multiple comparisons. Suppose that $target\_\alpha$ is 0.05 (as usual). Then, applying equation 29 with $k = 3$, $\alpha_1 = 0.025$ and $\alpha_2 = 0.05$. Next, in order to compute the corresponding statistics and p values, we need to calculate the standard error $SE = \sqrt{\frac{3(3+1)}{6\times 11}} = 0.426$ and each p value is calculated from the corresponding z score based on consulting a table of probabilities for the standardised normal distribution.

The calculations of the z_i and p_i values for the above example are shown in Table 14. As a result, the Holm procedure rejects the first null hypothesis (H_0), but does not reject the second H_0. Hence, there is sufficient evidence to support the claim that method A significantly outperformed method B, but there is not enough evidence to support the claim that A is significantly better than C.

4.5 Computational Results

This section reports the computational results obtained by the proposed Random Forests variants and other methods, including the results of statistical significance tests. This section is divided into five subsections, as follows.

Subsection 4.5.1 compares the predictive performance of one of the RF variants proposed in this chapter, namely Random Target-Imputation Forest (RTIF) against the performance of two simple baselines: a standard Random Forest (RF) for the regression task and the IPC-weight RF (Vock, Wolfson et al., 2016) – described in Chapter 2. The standard RF for regression uses a simple approach to cope with censored instances, dropping the censored instances. The IPC-weight RF consists of removing the censored instances and assigning weights to the uncensored instances, as described in Subsection 2.3.2. Hence, both these baseline methods essentially involve modifying the data rather than modifying the RF method, i.e. they use a standard RF method. It is important to note that hyperparameter tuning was performed not only for the proposed RTIF, but also for these two baseline methods. The same candidate hyperparameter settings were used to tune these 3 algorithms.

Subsection 4.5.2 compares the predictive performance of the other RF variant proposed in this chapter, K-Nearest Neighbour RF (KNN-RF), against the performance of the same two baselines.

Subsection 4.5.3 compares the predictive performance of the two proposed RF

variations against the standard Random Survival Forest (RSF) method. RSF is one of the most popular and powerful methods for survival analysis in the area of machine learning, and it uses sophisticated survival analysis concepts and techniques to cope with censored data. More precisely, in RSF the procedures for selecting the best variable at each node of a decision tree and the procedure for computing the predictions at leaf nodes are based on specific survival analysis concepts for coping with censored data (as discussed in Chapter 2), unlike the standard RF for regression and the IPC-weight RF.

Subsection 4.5.4 compares the predictive performance of the two proposed RF variations against the Cox Proportional Hazards (PH) method. The Cox PH method is a classical and probably the most popular method for survival analysis in the area of statistics.

We present these results in four separate subsections, as opposed to comparing all 5 methods in a single section, in order to more clearly identify the relative strength or weakness of the proposed RF variants by comparison with different types of baseline methods, with different degrees of sophistication in their approach to cope with censored data.

Finally, subsection 4.5.5 reports the hyper-parameter settings most frequently chosen for each RF method by the previously described nested cross-validation approach for hyper-parameter optimisation. In this case, there is no predictive performance comparison across methods, rather the goal is simply to identify patterns involving the best hyper-parameter settings for all the RF variants used in the experiments, so it is appropriate to discuss this in a single subsection.

4.5.1 Results comparing the proposed RTIF against two baseline Random Forest methods

Table 15 reports the C-index values obtained by three variants of Random Forests: the standard Random Forest for regression task, the IPC-weight Random Forests

Table 15: Predictive performance obtained by three variants of Random Forests

Dataset		RF_regressor	IPC weight	RTIF
Disease	uncensoring ratio	c-index	c-index	c-index
Alzheimer	69/6825 (1.0%)	0.7160	0.6576	**0.7742**
Angina	165/6488 (2.5%)	0.5472	0.5691	**0.5723**
HeartAtt	186/6607 (2.8%)	0.5856	0.5894	**0.6228**
Psychiatric	219/5972 (3.5%)	0.4709	**0.4834**	0.4692
Stroke	270/6632 (4.1%)	0.5919	0.5983	**0.6366**
Diabetes	416/6500 (6.4%)	0.6395	0.6796	**0.7443**
Cancer	562/6386 (8.8%)	0.5129	0.5071	**0.5135**
Arthritis	784/4276 (18.3%)	0.5019	0.5068	**0.5078**
Any-disease (ELSA)	979/3280 (29.8%)	0.5249	0.5316	**0.5384**
Any-disease (SHARE)	101300/139522 (72.6%)	0.6597	0.6552	**0.7061**
HD	783/1097 (71.4%)	0.4874	0.4927	0.5435
Average Rank		2.64	2.18	**1.18**

(Vock, Wolfson et al., 2016) and the proposed Random Target-Imputation Forest (RTIF) method, breaking down by each disease used as the target variable to be predicted. Recall that the target variable in each dataset represents the time passed (in months) until a subject is diagnosed with the corresponding disease.

The second column of this table shows the number and ratio of uncensored instances in each dataset. The datasets are listed in the table in increasing order of their uncensoring ratio. Note that in nearly all datasets the uncensoring ratio is smaller than 50%, i.e. the majority of instances were censored. The only two exceptions are the SHARE (Any-disease) dataset, where 72.6% of the instances are uncensored, and the Haemodialysis (HD) dataset, where 71.4% of the instances are uncensored. The uncensoring ratio is smaller than 10% in 7 of the 11 datasets, which makes them particularly challenging datasets for survival analysis.

The last three columns of Table 15 report the C-index values obtained by the three RF variants. For each dataset (i.e. in each row), the highest C-index value is highlighted in boldface font; and the last row shows the average rank obtained

by each method. To compute this, first each method was assigned a rank for each dataset, with the best method (highest C-index) assigned rank 1 and the worst method (lowest C-index) assigned rank 3; and then the rank of each method was averaged across the 11 datasets.

As shown in Table 15, the proposed RTIF performed considerably better than the other two baseline variants, with 9 wins across the 11 datasets and the best average rank of 1.18. Regarding the performance of the IPC-weight RF method, its lower predictive accuracies were presumably partly due to the small proportions of non-zero-weight instances – e.g., in 7 datasets the proportion of non-zero-weight (uncensored) instances is smaller than 10%, as mentioned earlier. Recall that the IPC weight is 0 for censored instances. Hence, for most datasets, the large majority of instances were censored and had absolutely no participation in the model training. In any case, the IPC-weight RF at least outperformed the standard RF regressor, with the former obtaining 1 win and an average rank of 2.18, whilst the latter did not obtain any win and had the worst average rank, 2.64. The poor performance of the RF regressor was expected since this is standard a standard RF method for regression which was not designed for survival analysis (unlike IPC-weight RF, which was designed for survival analysis).

Intuitively, we would expect that, in general, the performance of all these three RF methods would be better (with a higher C-index) in datasets with larger uncensoring ratios than in datasets with very small uncensoring ratios, since the latter provides substantially less information about the precise values of the target variable. However, the results do not support this intuition. In particular, for all the 3 RF variants, their two highest C-index values were obtained on the Alzheimer and Diabetes datasets, both of which have very low proportions of uncensored instances, 1% and 6.4%, respectively (as shown in the second column of Table 15). In addition, for all the three RF variants, in general, their lowest C-index values were obtained on the Psychiatric, Arthritis, Cancer and HD datasets, where

the latter three have an uncensoring ratio of 18.3%, 8.8%, and 71.4% respectively. Although 18.3% and 8.8% are low ratios, they are higher than the uncensoring ratios of the two aforementioned datasets with the best results; and 71.4% is by far the highest uncensoring ratio in our 11 datasets. Hence, it is clear that a higher uncensoring ratio is not associated with a higher C-index value in general, for the results in Table 15.

The non-parametric Friedman test (Demšar, 2006) was used to determine whether or not there is a significant difference among the average ranks of the three methods and the mean rank of 2.0 under the null hypothesis. The calculated value of F_F is 12.407. With 3 methods and 11 datasets, F_F is distributed according to the f distribution with 3 - 1 = 2 and (3-1) x (11-1) = 20 degrees of freedom. The critical value of F(2,20) for $\alpha = 0.05$ is 3.493 (p-value = 0.002). F_F is greater than the critical value, and so the null hypothesis is rejected at the conventional significance level of $\alpha = 0.05$. Hence, there is a statistically significant difference between the performances of the three methods as a whole.

Therefore, we proceed with the Holm post-hoc test (Demšar, 2006), which compares the average rank of the best (control) method, RTIF, against each of the other two methods, by adjusting the significance level of $\alpha = 0.05$ to compensate for multiple comparisons. The results were that RTIF significantly outperformed both the RF-regressor (p-value = 0.0003, smaller than adjusted $\alpha = 0.025$) and the IPC-Weight RF variant (p-value = 0.010, smaller than $\alpha = 0.050$). Hence, there is sufficient evidence to support that the RTIF significantly outperforms the other two baseline RF variants, since the two null hypotheses can be rejected.

Table 16: Predictive performance obtained by three variants of Random Forests

Dataset		RF_regressor	IPC weight	KNN-RF
Disease	uncensoring ratio	c-index	c-index	c-index
Alzheimer	69/6825 (1.0%)	0.7160	0.6576	**0.7747**
Angina	165/6488 (2.5%)	0.5472	**0.5691**	0.5677
HeartAtt	186/6607 (2.8%)	0.5856	0.5894	**0.6269**
Psychiatric	219/5972 (3.5%)	0.4709	**0.4834**	0.4748
Stroke	270/6632 (4.1%)	0.5919	0.5983	**0.6421**
Diabetes	416/6500 (6.4%)	0.6395	0.6796	**0.7029**
Cancer	562/6386 (8.8%)	0.5129	0.5071	**0.5093**
Arthritis	784/4276 (18.3%)	0.5019	**0.5068**	0.5020
Any-disease (ELSA)	979/3280 (29.8%)	0.5249	0.5316	**0.5351**
Any-disease (SHARE)	101300/139522 (72.6%)	0.6597	0.6552	**0.7034**
HD	783/1097 (71.4%)	0.4874	0.4927	**0.5742**
Average Rank		2.64	2.00	**1.36**

4.5.2 Results comparing the proposed KNN-RF against two baseline Random Forest methods

Table 16 reports the C-index values obtained by three variants of Random Forests (RF): the standard Random Forest for the regression task, the IPC-weight Random Forest (Vock, Wolfson et al., 2016) and the proposed KNN-RF method, breaking down by each disease used as the target variable to be predicted – whose value is the number of months passed until a subject is diagnosed with the corresponding disease, as described earlier. The second column of this table shows the number and ratio of uncensored instances in each dataset, which also appears in Table 15 but is repeated here for the reader's convenience. The boldfaced C-index values highlight the best result for each dataset, and the average ranks in the last row are computed as previously described for Table 15.

As reported in the C-index columns of Table 16, the proposed KNN-RF performed, overall, substantially better than the other two baseline methods. More

specifically, KNN-RF was the winner in 8 of the 11 datasets, followed by IPC-weight RF, the winner in the three other datasets. KNN-RF also obtained the best average rank, 1.4.

Again (similarly to the results in Table 15), all of the RF variants had very poor performance (C-index around 0.5) in 3 datasets (Arthritis, Cancer and Psychiatric disorder); and the two baseline RF methods also had very poor performance on the HD dataset. The three highest C-index values in the entire Table 16 were 0.7747, 0.7034 and 0.7029, obtained by the proposed KNN-RF in the Alzheimer, SHARE and Diabetes datasets, respectively.

We applied the non-parametric Friedman test (Demšar, 2006) to determine whether or not there is a significant difference between the average ranks of the three methods and the mean rank of 2.0 under the null hypothesis. The value of F_F is 6.806 while the critical value is 3.493 (p-value = 0.002). F_F is greater than the critical value, and so the null hypothesis is rejected at the conventional significance level of $\alpha = 0.05$. Hence, there is a statistically significant difference between the performances of the three methods, KNN-RF included, as a whole.

Therefore, we proceed with the Holm post-hoc test (Demšar, 2006), which compares the average rank of the best (control) method, KNN-RF, against each of the other two methods, by adjusting the significance level of $\alpha = 0.05$ to compensate for multiple comparisons. The results were that KNN-RF significantly outperformed RF-regressor (p-value = 0.0014, smaller than adjusted $\alpha = 0.025$) whereas it was not statistically better than the IPC-Weight RF (p-value = 0.0678, greater than $\alpha = 0.05$). Hence, there is a sufficient evidence to support that the KNN-RF significantly outperforms only the RF-regressor, and not the IPC-Weight RF.

Table 17: Predictive performance obtained by three variants of Random Forests: the proposed RTIF and KNN-RF, as well as Random Survival Forest

Dataset		RSF	RTIF	KNN-RF
Disease	uncensoring ratio	c-index	c-index	c-index
Alzheimer	69/6825 (1.0%)	0.7632	0.7742	**0.7747**
Angina	165/6488 (2.5%)	**0.5854**	0.5723	0.5677
HeartAtt	186/6607 (2.8%)	**0.6369**	0.6228	0.6269
Psychiatric	219/5972 (3.5%)	**0.5390**	0.4692	0.4748
Stroke	270/6632 (4.1%)	0.6099	0.6366	**0.6421**
Diabetes	416/6500 (6.4%)	**0.7907**	0.7443	0.7029
Cancer	562/6386 (8.8%)	**0.5349**	0.5135	0.5093
Arthritis	784/4276 (18.3%)	**0.5443**	0.5078	0.5020
Any-disease (ELSA)	979/3280 (29.8%)	**0.5489**	0.5384	0.5351
Any-disease (SHARE)	101300/139522 (72.6%)	**0.7118**	0.7061	0.7034
HD	783/1097 (71.4%)	**0.5794**	0.5435	0.5742
Average Rank		**1.36**	2.27	2.36

4.5.3 Results comparing the new RF variants against Random Survival Forest

Table 17 reports the C-index values for Random Survival Forest (RSF), which is a popular and powerful method for survival analysis in the area of machine learning; and it compares its results against the two proposed Random Forest (RF) variants, RTIF and KNN-RF, which use different imputation approaches for the target variable of the censored training instances. Recall that the large majority of instances were censored in all datasets, except in the SHARE and HD datasets.

Regarding the overall predictive accuracy of the three methods, the table shows that RSF achieved the highest C-index values for 9 out of the 11 datasets. Furthermore, it also achieved the best average rank, 1.36, substantially better than the average ranks for RTIF (2.27) and KNN-RF (2.36). RTIF did not achieve a single win for any of these datasets, whilst KNN-RF obtained two wins but had

the worst predictive performance in general, with the largest rank (2.36).

The non-parametric Friedman test was used to determine whether or not there is a significant difference among the average ranks of the three different methods and the mean rank of 2.0 under the null hypothesis. The calculated value of F_F is 4.405 and the critical value of F(2,20) for $\alpha = 0.05$ is 3.493. Note that F_F is greater than the critical value, and so the null hypothesis is rejected. Hence, there is statistical evidence to support the claim that there is a significant difference between the performance of the three methods.

Therefore, we proceed with the Holm post-hoc test (Demšar, 2006), which compares the average rank of the best method, RSF, against each of the other two methods, by adjusting the significance level of $\alpha = 0.05$ to compensate for multiple comparisons. The results were that RSF significantly outperformed the other two methods: (1) against KNN-RF (p-value = 0.0095, which is smaller than adjusted $\alpha = 0.025$); and (2) against RTIF (p-value = 0.0165, which is smaller than $\alpha = 0.05$). Hence, there is sufficient evidence to confirm that RSF, a popular and powerful method for survival analysis in the area of machine learning, has significantly better performance than our two proposed methods in these datasets.

4.5.4 Results comparing the new RF variants against Cox Proportional Hazards Regression

Table 18 reports the C-index values for Cox's Proportional Hazard (PH) Regression, which is a very popular method for survival analysis in the area of statistics; and it compares its results against the two proposed Random Forest (RF) variants, RTIF and KNN-RF, which use different imputation approaches for the target variable of the censored training instances.

Regarding the overall predictive accuracy of the three methods, both the newly proposed variants outperform the Cox PH regression. The RTIF method obtained the best average rank (1.91), and KNN-RF came second with an average rank of

Table 18: Predictive performance obtained by Cox's PH regression and two proposed variants of Random Forests: RTIF and KNN-RF

Dataset		CoxPH	RTIF	KNN-RF
Disease	uncensoring ratio	c-index	c-index	c-index
Alzheimer	69/6825 (1.0%)	0.6175	0.7742	**0.7747**
Angina	165/6488 (2.5%)	0.5672	**0.5723**	0.5677
HeartAtt	186/6607 (2.8%)	0.6219	0.6228	**0.6269**
Psychiatric	219/5972 (3.5%)	**0.5333**	0.4692	0.4748
Stroke	270/6632 (4.1%)	0.5952	0.6366	**0.6421**
Diabetes	416/6500 (6.4%)	0.7300	**0.7443**	0.7029
Cancer	562/6386 (8.8%)	**0.5163**	0.5135	0.5093
Arthritis	784/4276 (18.3%)	**0.5399**	0.5078	0.5020
Any-disease (ELSA)	979/3280 (29.8%)	0.5345	**0.5384**	0.5351
Any-disease (SHARE)	101300/139522 (72.6%)	**0.7242**	0.7061	0.7034
HD	783/1097 (71.4%)	0.5736	0.5435	**0.5742**
Average Rank		2.09	1.91	2.00

2.00. Cox PH regression obtained the worst average rank of 2.09.

The Friedman test determines that, with the F_F of 0.083 the critical value of 3.493, F_F is smaller than the critical value (p-value = 0.913), and so the null hypothesis cannot be rejected. Hence, there is no statistical evidence to support the claim that any of the three methods has better predictive performance than the others.

4.5.5 Most Frequently Selected Hyper-parameter Values

Table 19 reports the hyper-parameter settings most frequently selected by the nested cross-validation procedure (which was used for hyper-parameter optimisation). Recall that the nested cross-validation procedure selected one configuration of hyper-parameter settings for each RF variant for each of the training sets of the external cross-validation, for each dataset. Hence, among those selected hyper-parameter settings per dataset, the most frequently selected one is reported in the

Table 19: Most Frequently Selected Hyper-parameter Settings, for each RF variant and each dataset

Dataset		RF_regressor	IPC weight	RSF	RTIF	KNN-RF
Disease	uncensoring ratio	(mtry/ node size)	(mtry/ node size)	(mtry/ node size)	(mtry/ node size)	(mtry/ node size)
Alzheimer	69/6825 (1.0%)	(4, 5)	(10, 5)	(4, 10)	(4, 5)	(4, 10)
Angina	165/6488 (2.5%)	(4, 10)	(4, 10)	(7, 5)	(4, 10)	(4, 5)
HeartAtt	186/6607 (2.8%)	(13, 10)	(7, 7)	(4, 10)	(4, 10)	(4, 7)
Psychiatric	219/5972 (3.5%)	(4, 10)	(13, 5)	(10, 5)	(13, 5)	(13, 5)
Stroke	270/6632 (4.1%)	(4, 10)	(4, 10)	(4, 10)	(4, 10)	(4, 10)
Diabetes	416/6500 (6.4%)	(4, 10)	(13, 5)	(4, 10)	(4, 10)	(4, 10)
Cancer	562/6386 (8.8%)	(4, 10)	(4, 5)	(13, 5)	(13, 5)	(13, 5)
Arthritis	784/4276 (18.3%)	(4, 7)	(4, 10)	(4, 10)	(4, 10)	(4, 10)
Any-disease (ELSA)	979/3280 (29.8%)	(4, 10)	(4, 10)	(4, 10)	(4, 10)	(4, 10)
Any-disease (SHARE)	101300/139522 (72.6%)	(3, 10)	(4, 10)	(4, 10)	(4, 10)	(4, 10)
HD	783/1097 (71.4%)	(5, 5)	(5, 5)	(5, 5)	(8, 5)	(10, 5)

table.

Recall also that all RF variants had two hyper-parameters optimised: mtry and the minimum number of instances per leaf node (node size, for short). The candidate node size values were 5, 7, 10 for all RF variants and datasets. The candidate mtry values were the same for all RF variants and nearly the same across the datasets; more precisely: mtry = 4, 7, 10 or 13 for the nine ELSA datasets, mtry = 3, 4, 6, 8 for the SHARE dataset (due to the substantially smaller number of features in this dataset, as mentioned earlier), and mtry = 5, 6, 8, 10 for the Haemodialysis dataset.

As shown in Table 19, the most frequently selected hyper-parameter setting combination was (mtry = 4, node size = 10), which was the winner 31 times out of 55 (considering the five RF methods and the 11 datasets). In addition, all five RF methods consistently chose the (4,10) combination in two datasets: Stroke and Any-disease ELSA.

Moreover, considering the selection frequency of each hyper-parameter setting separately, the mtry value of 4 was particularly effective: it was selected 37 out of 55 times – or more precisely out of 50 cases where it could possibly be selected

since the value 4 was not a candidate mtry value in the HD dataset. This mtry value was the smallest of the 4 candidate values in the ELSA datasets, and the second smallest candidate mtry value in the SHARE dataset. Hence, these results show that, in these datasets, smaller mtry values (which tend to increase the diversity of trees in the RF) are more effective in general.

In addition, the node size value of 10 was also very effective by itself; it was selected 33 out of 55 times. This value of 10 was the largest of the 3 candidate node size values. Hence, it seems that it is better to use relatively large node size values for these datasets, so that the number of instances in each leaf node is large enough to allow the decision trees in the RF to generalise well from those instances.

4.6 Conclusion

This section offers a concise summary of the computational results for the two novel variants of the Random Forest (RF) algorithm proposed in this chapter – the Random Target-Imputation Forests (RTIF) and the K-Nearest-Neighbours-Imputation Random Forests (KNN-RF).

Both these RF variants aimed at imputing censored target variables. This small yet effective modification to the standard RF algorithm allows it to perform survival-analysis tasks, enhancing its capability while maintaining its simplicity. The efficacy of these variants was assessed by comparing them against four baseline methods: a standard RF for regression, a RF with Inverse Probability of Censoring (IPC), Cox regression with the Proportional Hazard (PH) assumption, and a standard Random Survival Forest (RSF). Performance was evaluated using the C-index on 11 real-world biomedical datasets.

In summary, the computational results were as follows. First, RTIF significantly outperformed both the standard RF regressor and RF with IPC in terms

of predictive accuracy. Second, KNN-RF outperformed the standard RF regressor significantly but showed no significant difference when compared with RF with IPC. Finally, the last experiment revealed that RTIF and KNN-RF both had a higher predictive accuracy than the Cox PH regression, with RTIF slightly leading, although the differences were not statistically significant. However, RTIF and KNN-RF were significantly outperformed by RSF.

These results evidently confirm the potential and effectiveness of RTIF and KNN-RF in enhancing the standard RF algorithm's performance in survival analysis tasks, although RTIF and KNN-RF were outperformed by RSF. The next chapter will focus on proposing new variations of RSF to try to improve RSF's predictive performance.

Chapter 5

New Variants of Random Survival Forests

5.1 Introduction

This chapter describes in detail two proposed types of modifications of the Random Survival Forest (RSF) algorithm, which is a type of random forest algorithm designed for survival analysis with censored data (Ishwaran et al., 2008). More precisely, the proposed variations of RSF involve the modification of the node-split criterion and the leaf-node-prediction criterion of the algorithm. This chapter also reports the results of computational experiments evaluating the proposed variations of the RSF algorithm.

This chapter is organised as follows. Section 5.2 reviews related work on different variants of RSF. Section 5.3 describes the proposed variants of RSF. Section 5.4 describes the experimental methodology. Section 5.5 reports experimental results, including both predictive accuracy results and an analysis of the most important predictive features in the best models learned in our experiments.

5.2 Related Work on Random Survival Forests

There are a number of machine learning studies dealing with censored data in survival analysis using the standard Random Survival Forest (RSF) algorithm. To mention a few, the work of (Akai et al., 2018b; Mogensen, Ishwaran and Gerds, 2012; Pang et al., 2012) employs the RSF method to analyse survival problems from different application domains. Additionally, some biomedical studies such as (Hamidi et al., 2016; Dietrich et al., 2016; Miao et al., 2018a) use RSF to predict rates of deaths in patients who developed age-related diseases as well as attempting to identify the most relevant factors in their respective exposures. Some very recent work (Li et al., 2022) uses RSF to predict the recurrence of breast cancer from a long-term clinical dataset over 8 years in order to identify the high-risk patients, and (Zhang et al., 2022) creates a RSF model for prognosis prediction in patients with sepsis.

In contrast to the aforementioned works (which in general use a standard RSF algorithm), there are several studies that propose new variants of the RSF algorithm that involve the modification of the leaf-node-prediction criterion and/or the node-split criterion of the algorithm, as follows.

The work of (Weeraddana et al., 2020) proposes a modified leaf-node-prediction-criterion where, instead of outputting the CHF at each leaf node, the lower and upper bounds of the prediction uncertainty are calculated from the conditional probability distribution of instances.

In (Wang and Li, 2017) the authors comprehensively review several node-splitting-criteria of tree-based methods for survival analysis. They compare 9 different node-splitting-criteria for survival trees (i.e. decision trees for survival analysis), and then 4 of those criteria are implemented in the original RSF method (Ishwaran and Kogalur, 2007). Besides this, the other studies have proposed alternative criteria, namely AUC splitting, C-index splitting, L_1 splitting and Maximally selected rank splitting.

In (Miao et al., 2018b) the authors propose "improved Random Survival Forest" (iRSF) where the standard node-split-criterion of RSFs (the standard log-rank test) is replaced with improved log-rank-type tests from (Yang and Prentice, 2010). The said criterion uses adaptive weights based on the change of the hazard rates over time to handle the non-proportional hazard issue. iRSF was applied to a clinical dataset about heart failures and the learned models were able to identify the critical features responsible for heart failure. The authors concluded that their approach outperforms the conventional RSF — i.e., it obtained a higher C-index value.

In (Eifler, 2014) the authors proposed a RSF variant where the log-rank test was replaced with the C-index node-split criterion. Overall, their proposed variant outperforms the original RSF in their datasets, especially ones with high censoring ratios.

In (Wright, Dankowski and Ziegler, 2017) the authors proposed Conditional Inference Forests (CIF); where the standard node-split-criterion was replaced with the maximally selected rank statistic, which is another statistical test based on log-rank scores. Specifically, for each feature the test selects the split point with maximally selected rank statistics and obtains the corresponding p-value, and then the feature with the smallest p-value is selected for splitting the data at the current node in the current survival tree. They concluded that said method removes the bias towards selecting features with multiple candidate split points in standard RSFs.

In (Wang and Liu, 2018) the authors used RSF to predict gene expression and identify biomarkers in survival analysis. Their RSF variant uses the standard log-rank test as the node-split-criterion, but samples candidate features (genes) using topological weights. That is, *mtry* features are randomly sampled as candidate features for each node splitting, based on the topological weights of the genes that they represent. These topological weights are computed by a directed

random walk algorithm, which is biased to sample the most highly connected genes. Hence, this RSF variant is suitable only for application domains where the features' topological weights can be computed, unlike most other RSF variants proposed in the literature, which have more general applicability.

Besides, a few studies replaced both the standard node-split-criterion and the standard leaf-node-prediction-criterion by new variants of these criteria, as follows.

In (Wongvibulsin, Wu and Zeger, 2019) the authors proposed Random Forests for Survival, Longitudinal, and Multivariate data analysis (RF-SLAM). They use Poisson regression with log-likelihood for the node-splitting criterion and Bayes estimate of the event rates for the leaf-node-prediction criterion. Broadly speaking, the RF-SLAM algorithm builds decision trees using data binned based on user-defined lengths of time – counting process information units (CPIUs). Each individual can have many CPIUs. The algorithm assumes that an individual's event hazard is constant within each CPIU, but it does not assume a proportional hazard across the entire follow-up time for an individual. The algorithm assembles the predictions for each CPIU for an individual to obtain the hazard function.

In (Jaeger et al., 2019) the authors proposed the Oblique Random Survival Forest (ORSF) algorithm, where subsampling is used rather than bootstrap sampling with replacement and both the node-splitting and the leaf-node-prediction criteria are extended to enrich the accuracy of CHF. Furthermore, the log-rank test, the classic node-split-criterion for RSF, is used together with the oblique splits (regularized Cox's proportional hazards models): it uses the elastic net (Zou and Hastie, 2005), a combination of Lasso and Ridge regularizations, to learn the regularized Cox's proportional hazard models at each parent tree node. Also, the leaf-node-prediction-criterion becomes the Nearest Neighbour aggregation scheme as in Conditional Inference Forests (CIF) (Wright, Dankowski and Ziegler, 2017). The authors concluded that, regarding predictive performance, although ORSF outperformed the gradient boosting method in simulated datasets, their proposed

method lost to gradient boosting when tested in real-world datasets.

Last but not least, there are some studies that enrich RSF by other means, as follows.

In (Utkin et al., 2019b) the authors proposed a weighted Random Survival Forest (WRSF) where a weight is assigned to every tree in the forest for aggregating the trees' predictions; where weights are computed to maximise C-index values.

In (Tollenaar and Van Der Heijden, 2019) the authors compared two tree-based ensemble methods in an application involving criminal recidivism data. The two methods were RSF and the Gradient-boosted Cox proportional hazard loss with regression trees as the base learner. They concluded that RSF's prediction error increases relative to the Gradient-boosting's error; the longer the observation time, the longer the increase. Hence, the Gradient-boosting method obtained slightly higher overall performance.

In (Gul et al., 2020) the authors proposed Optimal Ensemble of Survival Tree (OSTE), an extension of Optimal Trees Ensemble (OTE). Broadly speaking, the algorithm selects the best subset of survival trees for prediction: it starts with the top-ranked tree and adds one of the top-M trees at a time (where M is a user-defined parameter), as long as this improves predictive performance.

5.3 Modifying the Node-Splitting and Leaf-Node-Prediction Criteria of Random Survival Forests

We propose new variants of the Random Survival Forest (RSF) algorithm, in order to try to improve this type of algorithm's predictive performance. Since RSF is a decision tree-based learning algorithm, we propose modifying the two key components of the algorithm: (1) the node-splitting criterion, and (2) the leaf-node-prediction criterion.

Table 20: weights used in Equations (32) and (33) by different Log-rank statistics variants

Test Statistics	Weight
Log-rank	1
Wilcoxon	n
Tarone-Ware	sqrt(n)

5.3.1 Modifying the Node-Splitting Criterion

We propose to replace the log-rank statistics, the default node-splitting criterion used in RSF, with its weighted versions, replacing the $O_i - E_i$ term in the numerator and the denominator of Equation (19) by Equations (32) and (33), respectively. Note that Equations (32) and (33) have weights w_j and w_j^2, respectively, multiplying the term within the scope of the summation symbol. Hence, the effect of using the weighted Equations (32) and (33) to implement Equation (19) will depend on how those weights are determined. In this work, the weight w_j is varied according to the Log-rank variants in Table 20, including the Wilcoxon and Tarone-Ware criteria, where n is the number of subjects in the current risk set $(n_{1j} + n_{2j})$ and *sqrt(n)* is the square root of n.

$$O_i - E_i = \sum_{j=1}^{k} w_j \left(m_{ij} - e_{ij}\right) \tag{32}$$

$$\mathrm{Var}\left(O_i - E_i\right) = \sum_{j=1}^{k} w_j^2 \frac{n_{1j} n_{2j} \left(m_{1j} + m_{2j}\right) \left(n_{1j} + n_{2j} - m_{1j} - m_{2j}\right)}{\left(n_{1j} + n_{2j}\right)^2 \left(n_{1j} + n_{2j} - 1\right)} \tag{33}$$

Note that in Equations (32) and (33) the summation is performed over the k distinct failure times. Hence, the original Log-rank node-splitting criterion assigns the same importance (weight 1) to all failure times, whilst the Wilcoxon

and Tarone-Ware node-splitting criteria emphasize earlier failure times. One motivation for this emphasis is that the value of n (the size of the risk set) tends to be greater in earlier failure times, increasing the statistical support for the calculations of the test statistics.

5.3.2 Modifying the Leaf-Node-Prediction Criterion

As discussed earlier, in the standard RSF algorithm, the prediction made by the leaf nodes of the trees for the current instance (subject) being classified is the value of the ensemble Cumulative Hazard Function (CHF) for that instance, which is the average of the CHF values over all trees in the forest, as shown in Equation 34. Note that a CHF value is essentially a sum of the "failure rates" across all observed failure times, but it was not designed to directly answer the fundamental question about how long a subject will "survive", i.e. how much time will pass until the event of interest occurs for a given subject.

$$H(t) = \sum_{j=0}^{t} \left(\frac{m_j}{n_j} \right) \tag{34}$$

Although Random Survival Forests generally produce high predictive accuracy as measured by the C-index, ensemble mortality is another type of measure designed to describe the population's survival distribution against time. In other words, it is not conceptually suitable with the goal of predicting a survival time: how long until the event of interest will occur to a given individual.

Therefore, we propose a series of leaf-node-prediction criteria that are inspired by the standard Random Forest algorithm for regression (rather than for survival analysis), where the value predicted at a leaf node is an estimate of the mean of the target variable over the instances at that leaf node.

However, the standard Random Forest algorithm for regression cannot cope with censored data. Therefore, we propose a modification of the Random Survival

Forest algorithm where, at each leaf node in a decision tree, the value predicted at that leaf node will be an estimate of the mean survival time of the instances at that leaf node by taking into account censored data. The proposed modification will be in general denoted by "constant-hazard-mean" prediction criteria.

There are four proposed variants of the leaf-node-prediction criteria (one of which is just a naive baseline approach, not recommended in general), as follows.

The Naive Mean criterion: Disregarding an instance' censorship status

First, this naive approach is meant to be used as a very simple baseline method in comparison with the other three variants. In this approach, we make the very strong and unrealistic assumption that all of the censored instances experienced the event of interest at the time of censorship. In other words, the predicted value for the target variable at each leaf node will be simply calculated as the mean of the values of the target variable over all instances assigned to that leaf node regardless of the censoring status. That is, the calculation ignores the value of the binary uncensored-status variable – which indicates whether an instance is censored or uncensored.

Hence, the predicted value for the target variable at each leaf node will be an under-estimate of "real", unknown mean survival time for that node; and it is expected to reduce predictive accuracy – particularly in datasets with a large proportion of censored instances, like in our datasets. The only advantage of this naive approach is its conceptual and computational simplicity. Hence, as mentioned earlier, this is a very simple baseline approach.

The "constant-hazard-mean" criterion: a new mean formula based on the assumptions of constant hazard rates and non-informative censoring

This approach modifies the classical formula for computing the mean value of the target variable at each leaf node, in order to be able to handle censored instances.

Table 21: The linear correlation coefficients between age and survival time for the uncensored instances, for each dataset (disease).

Dataset	Alzheim.	Angina	Heart attack	Psychiat.	Stroke	Diabetes	Cancer	Arthritis	Any-dise. ELSA	Any-dise. SHARE
Correl. Coeff.	-0.196	-0.251	-0.168	-0.044	-0.163	-0.113	-0.164	-0.081	-0.125	-0.175

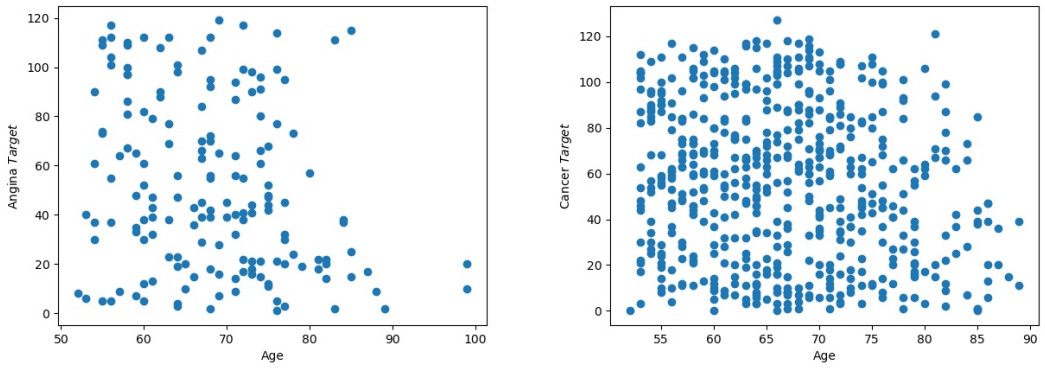
This can be done by making the following two assumptions:

- the hazard rate is constant throughout the study — i.e., a person's chance of experiencing the event of interest does not change with time (a strong assumption); and
- the censoring is non-informative — i.e., the time when an instance is censored is independent of its "failure" time, or in short, instances are censored at random (a common assumption in the survival analysis literature).

At first glance, the constant hazard rate assumption would seem unlikely to be satisfied in our datasets of age-related diseases, since the time passed until the diagnosis of age-related disease (our "survival time") tends to be smaller for older subjects. However, in our datasets this age effect is relatively small in general, and so that assumption can still be used to produce reasonable estimates of survival time in practice, as shown next.

To be precise, we measured the Pearson's linear correlation coefficient between the age and survival time of uncensored subjects, for each disease (target variable), i.e. for each dataset. Age was measured at the ELSA/SHARE survey's baseline, and survival time is the number of months passed from that baseline time until the diagnosis of a disease. Table 21 reports these correlations.

As expected, the correlations are negative, since older individuals are more likely to be diagnosed with an age-related disease sooner, resulting in a shorter 'survival time'. However, these correlations are quite weak in general. In addition, Fig 8 shows the scatterplots for two diseases (datasets) as examples: Angina (with



(a) Angina dataset (b) Cancer dataset

Figure 8: Survival time of uncensored individuals over different ages

the largest negative correlation, -0.251) and Cancer (with the 5th largest negative correlation, -0.164). Note that there is no clear correlation between age and survival time for ages below about 85. The (negative) correlation is clear and strong only for ages above about 85, representing a small minority of subjects in our datasets. Therefore, the constant hazard rate assumption is approximately valid in our datasets in general.

Given the aforementioned assumptions, we can conclude that all instances have identical remaining mean survival time μ, regardless of their previously observed survival time t (Selvin, 2008). With these two assumptions, as shown in (Selvin, 2008), the estimated mean survival time ($\hat{\mu}$) can be computed as shown in Equation (35):

$$\hat{\mu} = \frac{\sum_{j=1}^{n} t_j + m\hat{\mu}}{n} \tag{35}$$

where t_j is the value of the target variable (survival time) for the j-th individual, n is the total number of individuals (counting both uncensored and censored individuals), and m is the number of censored individuals. Recall that t_j is the true value of survival time if the j-th individual is uncensored, whilst it is a lower bound of the true, unknown survival time for censored individuals. Hence, in

Equation (35), the term $\sum_{j=1}^{n} t_j$ is simply the sum of all observed survival times, considering both uncensored and censored individuals, whilst the term $m\hat{\mu}$ adds the total "missing", unobserved survival time associated with all m censored individuals — assuming that each censored individual has a remaining expected survival time (after censorship) of μ — as implied by the above two assumptions (Selvin, 2008). By applying some simple algebraic operations to Equation (35), we derive Equation (35):

$$n\hat{\mu} = \sum_{j=1}^{n} t_j + m\hat{\mu}$$
$$\hat{\mu}(n-m) = \sum_{j=1}^{n} t_j$$

$$\hat{\mu} = \frac{\sum_{j=1}^{n} t_j}{(n-m)} \tag{36}$$

Hence, the estimated mean survival time at each leaf node of the survival trees in a RSF model is computed using Equation (35), where the summation of all survival times, censored and uncensored included, is divided by the number of uncensored instances.

The "weighted-age-mean" criterion: Extending the new mean formula with the age-based weights for censored instances

To take into account the dependence of an individual's survival time on their age (measured at the individual's last observed time), we can assign different weights to different censored individuals depending on their age, when computing the term $m\hat{\mu}$ – estimating the "missing" survival time for censored subjects – in Equation (35). The basic idea is that the older an individual is when she or he is censored, the smaller her or his expected remaining survival time is. Hence, the

term $m\hat{\mu}$ in Equation (35) is replaced by the term $\sum_{i=1}^{m} w_i \mu$, w_i in [0..1], where older individuals are assigned smaller weight values than younger individuals, to reflect the fact that the former is expected to have a smaller remaining survival time (after the censorship time point).

More precisely, the weight for a censored individual j (w_i) is calculated from Equation 37, so $w_i = 1$ if the age is the minimum (youngest) and $w_i = 0$ if the age is the maximum (oldest) among the set of last observed ages for all individuals in the training set.

$$w_i = \frac{age_{max} - age_i}{age_{max} - age_{min}} \tag{37}$$

To compute Equation 37, the last observed age for each i-th individual is the age of that individual at the time point where the individual was last observed - i.e., either the time when the individual was diagnosed with the disease corresponding to the target variable (for uncensored individuals) or the time when the individual was censored (for censored individuals). All the values of age_{max}, age_{min} and age_i are computed with respect to this set of last observed ages. Hence, for censored individuals, age_i is the age of the i-th individual at the time point when she or he was censored. Note that Equation 37 is used to compute the weights of censored individuals only, the weights of uncensored individuals are simply 1.

The rationale to compute w_i based on the set of last observed ages, rather than the ages at the baseline wave, is that the former takes into account more information about each individual's survival time. A simple example shows this point. Consider two censored individuals: Ind1's age was 55 at the baseline wave, and it was censored when her/his age was 70; whilst Ind2's age was 60 at the baseline wave, and it was censored at age 65. Clearly, although Ind1 was younger than Ind2 at the baseline wave, Ind2 is expected to have a larger remaining survival time than Ind1, considering their ages at the times when the individuals were

censored.

Hence, replacing the term $m\hat{\mu}$ in Equation 35 by the term $\sum_{i=1}^{m} w_i \mu$ and performing algebraic operations similar to the ones previously used to derive Equation 36, the new age-dependent estimate of mean survival time is given by Equation 38:

$$\hat{\mu} = \frac{\sum_{j=1}^{n} t_j}{(n - \sum_{i=1}^{m} w_i)} \tag{38}$$

The approach using this equation will be referred to as the "weighted-age-mean approach".

The "KNN-Mean" criterion: Computing the mean at a leaf node by using K-NN for estimating censored target values

Last but not least, although the weighted-age-mean approach takes into account the individual's age in the mean formula, the strong assumption of the hazard rate being constant throughout the study period is still required to utilise the equation. Hence, in order to avoid that assumption, we propose to use the K-Nearest Neighbour (K-NN) algorithm, a non-parametric machine learning method, within the procedure for computing the predicted value at each leaf node. The idea is to use K-NN to estimate the survival time of censored individuals, i.e. imputing target values based on K-NN's predictions. Note that K-NN is used only as part of the leaf-node prediction criterion of Random Survival Forests (RSF). That is, K-NN is not used during other procedures of the RSF algorithm – like the creation of bootstrap samples, selection of the best feature for node splitting, etc – which will continue to work with censored target values as usual.

More precisely, this approach works as follows. At each leaf node, for each censored subject, we replace the target value (number of months survived until censorship) but an estimate of the unknown survival time. That estimate will be computed essentially by using the K-NN algorithm to find the nearest uncensored

neighbour of the current censored subject, and assigning that uncensored neighbour's survival time to the survival time (target value) of the current censored subject.

In all the experiments reported in this chapter, the value of K in the K-NN algorithm was set to 1. The motivation for this setting is as follows. Recall that, in the proposed method, the K-NN algorithm is used to retrieve the K nearest *uncensored* neighbours. However, in general the datasets used in our experiments have a very small proportion of uncensored instances (e.g., only 1% in the Alzheimer's dataset). Hence, it is difficult to find several uncensored "neighbours" which are really "close" to a given instance. Therefore, setting $K = 1$ helps to avoid the risk of using inappropriate neighbours that are quite different from the current instance whose target variable value is being inputted by the K-NN algorithm.

To avoid the problem that each leaf node may contain very few or even just one uncensored subjects, which would not be enough to reliably compute nearest neighbours, the nearest neighbour can be computed from the entire training set. This introduces, however, the problem that this is a completely "global" computation, ignoring the characteristics of local subjects in the current leaf node. There is also the problem that, in high-dimensional datasets, the computation of distances is not very meaningful – i.e., all instances tend to be far away from each other in a very large dimensional space.

Therefore, as a solution to both problems, it is proposed to adapt the K-NN algorithm to consider only a subset of features to measure the distances between instances. More precisely, it is proposed to measure distance based on only the features that occur in the path from the root node to the current leaf node. This means that the local context of the leaf node is taken into account, i.e., the distance computation is not completely global, and it is partly based on local information. Furthermore, the high-dimensionality problem is greatly reduced

because the distance will be computed using only a subset of features, rather than all features.

Another problem that needs to be addressed is that the returned nearest neighbour may have a target value (survival time) smaller than the censored time of the current subject, which would be an invalid target-value estimation. This is avoided by further modifying the K-NN algorithm to find the nearest uncensored neighbour only among the uncensored subjects whose target value (survival time) is greater than or equal to the target value of the current censored subject.

In summary, we propose the pseudo-code shown in Algorithm 4, based on the just-described adapted K-NN algorithm, for computing the predicted mean at each leaf node of the RSF algorithm.

Notation used in Algorithm 4

- S_X = the set of uncensored subjects in the training set whose value of the target variable (their known survival time) is greater than or equal to the value of the target variable for subject X (i.e., the number of months that X "survived" until censorship)
- (x_i, y_i) = feature value vector x_i and target variable value y_i of the i-th instance
- $x_i.uncens$ = uncensored status of the i-th instance (1 = uncensored, 0 = censored)
- LB_X = lower bound for the target value of instance (subject) X
- NUN_X = the Nearest Uncensored Neighbour of the subject X among all training instances that are uncensored and have a target value higher than the lower bound of X (LB_X)

In this Chapter, we have proposed two different variants of Random-Forest-based approaches involving the Age variable and/or the KNN algorithm. In addition, recall that in Chapter 4 we assumed the Age variable to be highly correlated

Algorithm 4: RSF-KNN-mean

```
for each leaf node L do
    for each censored subject X do
        S_X ← { (x_k, y_k) | x_k.uncens = 1, LB_i ≤ y_k }
        NUN_X ← Nearest_Neighbours of X in S
        imputation ← arithmetic mean of all y in NUN
        y_X ← max(LB_X, imputation)
    end
    prediction ← arithmetic mean of all y at L
end
```

Table 22: The differences among the three Random Forests based approaches proposed in this work.

Comparison Criterion	KNN-imputation in RF	RSF with K-NN-mean	age-weighted mean RSF
Imputation technique	Yes	Yes	Technically yes
Involved in	Pre-processing step	Leaf-node-prediction	Leaf-node-prediction
Split criterion	RMSE	Log-rank	Log-rank
Use the Age variable	To find the closest uncensored neighbours	To estimate the remaining survival time	To estimate the mean survival time
Global information used	Valid neighbours' survival times	Max_Age, Min_Age,	Max_Age, Min_Age,
Local information used	Lower and upper-bounds of instance i	Max_Surv_Time(i), Cens_Surv_Time(i)	Lower-bound of instance i
If invalid	Use the default value	Always possible	possible when d0>0
Assumption	The event occurred by the end of the study	The event occurred by the end of the study	constant hazard rate

to the actual survival times of the censored subjects, and in that chapter we proposed to apply a simplified version of the K-NN algorithm to impute the censored target values accordingly. Table 22 shows the comparison among these three approaches proposed in this thesis.

5.4 Experimental Methodology

This section is divided into four parts, analogous to the four subsections of Section 4.4 in Chapter 4. The first subsection provides a summarised description of the datasets used in the experiments. The second subsection mentions the predictive

performance measure used. The third subsection describes a hyper-parameter tuning procedure, based on nested cross-validation. The fourth subsection describes the statistical significance tests used to analyse the results.

5.4.1 Datasets Used in the Experiments

The experiments used 11 datasets for 11 different age-related diseases (i.e. 11 separate survival prediction problems); and the creation of these datasets was described in detail in Chapter 3.

Recall that, as summarised in Section 4.4.1, 9 out of the 11 datasets were constructed from the ELSA data, containing between 3,000 and 7,000 instances (depending on the target variable), with exactly the same 44 predictive features, but different target variables. On the other hand, the dataset constructed from the SHARE data is much larger, containing almost 140,000 instances but only 15 predictive features. In essence, the instances represent individuals (subjects) in these surveys, and the target variables represent the 'survival times', more precisely, the time passed (in months) until an individual is diagnosed with a certain disease (for 8 datasets) or any of several diseases (for two datasets); whilst the predictive features represent biomedical information collected by nurses or other relevant characteristics of an individual (age and gender). The haemodialysis dataset has 1,097 instances with 38 features, and in this dataset the target variable represents the survival time of patients undergoing haemodialysis.

5.4.2 Predictive Performance Measure

Similarly to the previous chapter, in the experiments being reported in this chapter, the predictive performance of the learned survival models was estimated by the concordance index (C-index), which is a popular measure accounting for censored data. Recall that the C-index can be interpreted as the probability of correctly ordering the predicted survival values for a randomly chosen pair of subjects whose

actual survival times are different. For a precise formal definition of the C-index, the reader is referred to Section 2.2.5 in Chapter 2.

5.4.3 Hyper-Parameter Tuning via Nested Cross-Validation

All experiments are performed using a nested cross-validation procedure, where an inner cross-validation performs hyper-parameter tuning and an outer cross-validation estimates the predictive performance of the survival models. For a more detailed description of this nested cross-validation procedure, the reader is referred to Subsection 4.4.3.

We used 5 folds for the inner cross-validation for all datasets, whilst the number of folds for the outer cross-validation was set to 10 for the ELSA datasets and the haemodialysis data, and set to 5 for the SHARE dataset. The SHARE dataset has fewer folds to save computational time since it is much larger than the other datasets.

We tune two hyper-parameters of the RSF algorithm, namely: (a) *mtry*, i.e., the number of features randomly sampled to be used as candidate features for selection at each decision tree node; and (b) *d0*, i.e., the minimum number of uncensored instances required at each leaf node. According to (Probst, Wright and Boulesteix, 2019), *mtry* has been recognized as the most influential hyper-parameter in general in random forests. In addition, *d0* can be seen as the survival task-related counterpart of the hyper-parameter *node size* in classical random forests. It is considered worth tuning according to, for instance, the experiments in (Lin and Jeon, 2006) and (Lynch et al., 2017).

We consider three candidate values for *d0*, namely 1, 2 and 3, for all datasets. However, we consider a different set of candidate *mtry* values for the ELSA datasets and SHARE dataset separately, since there is a difference between their numbers of features, where the former contains 44 and the latter contains 15 predictive features. For ELSA, we specify four candidate values for *mtry* (4, 7, 10, 13).

The first two values were calculated as $ceil(ln(44)) = 4$ and $ceil(sqrt(44)) = 7$, where the natural logarithm (ln) and the square root (sqrt) are often considered default functions for specifying the value of mtry in random forests, and *ceil(x)* returns the 'ceiling' of x, i.e. the lowest integer that is greater than or equal to x (i.e. it rounds x up to the nearest integer). Similarly, the set of candidate values for SHARE is (3, 4, 6, 8), where the first two values were calculated as $ceil(ln(15)) = 3$ and $ceil(sqrt(15)) = 4$. Third, I use mtry = 5, 6, 8, 10 for the Haemodialysis dataset.

We also tune two hyper-parameters of the RTIF algorithm: (a) *mtry*, with the same aforementioned candidate values used for RSF; and (b) *node size*, analogous to *d0* in RSF, with one difference: *node size* is the minimum number of instances (regardless of their original censorship status) in a leaf node. So, its candidate values, (5, 7, 10), are larger than RSF's candidate *d0* values.

Hence, for all methods (the RSF versions and RTIF), for each dataset, at each iteration of the outer cross-validation, the inner cross-validation is run 12 times on the training set, considering 12 candidate random survival forest configurations (4 candidate *mtry* values times 3 candidate *d0* or *node size* values).

All analyses were performed using Python 3 with the scikit-survival library version 0.14.0 (Pölsterl, 2020), a Python module built on top of the scikit-learn machine learning library (Pedregosa, Varoquaux et al., 2011). In addition, some parts of the program were written and customised in Cython-code, which played an important role in boosting the performance of RSF due to Python's relatively slow performance. The program code is publicly available in the following GitHub link: `https://github.com/mastervii/new_variants_of_RSF`.

5.4.4 Statistical significance tests

The statistical analysis of the results was performed using, in most cases, the statistical significance tests described in Section 4.4.4, namely the Friedman test

to compare the performance of multiple (more than 2) algorithms and the post-hoc Holm's test to perform multiple hypothesis correction. However, subsection 5.5.4 compares the predictive accuracies of only two algorithms, and in this case we apply instead the Wilcoxon signed-ranks test. This test will be used more often in Chapter 6, and hence, its description is provided in that chapter only (in Section 6.3.4), in order to avoid redundancy. In general, we attempt to reject the null hypothesis that different survival analysis methods (i.e., different variants of RSF or RTIF) have the same predictive performance across the 11 datasets used in the experiments. We used the tests with the usual significance level of $\alpha = 0.05$.

5.5 Computational Results

This section reports the computational results obtained by the proposed Random Survival Forest (RSF) variants and the baseline methods, including the results of statistical significance tests.

This Section is divided into four subsections, as follows. The first subsection compares the results of three RSF variants, with three different node-splitting criteria, with all these variants using the same standard leaf-node-prediction criterion. The second subsection compares the results of five RSF variants, with five different leaf-node-prediction criteria, with all these variants using the same standard node-splitting criterion. The third subsection compares the results of three RSF variants, with three different node-splitting criteria, with all these variants using the same "constant-hazard-mean" leaf-node prediction criterion, which led to the best results among all RSF variants compared in the second subsection. The fourth subsection compares the results of the RSF variant which obtained the best results across all previous subsections, namely RSF-constant-hazard-mean, against the very popular Cox Proportional Hazards (PH) regression.

5.5.1 Comparing the results of RSF with three node-splitting criteria and standard leaf-node-prediction criterion

Table 23 reports the C-index values obtained by three different variants of Random Survival Forests (RSF), using three different node-splitting criteria: the Log-Rank, the Wilcoxon and the Tarone-Ware statistics. In all these three RSF variants, the leaf nodes compute the CHF as in standard RSFs.

The first two columns of Table 23 describe the datasets, where the first column identifies the disease used as the target 'survival' variable (time passed until disease diagnosis) and the second column shows the uncensoring ratio of each dataset, which is the ratio of the number of uncensored instances over the total number of (censored or censored) instances. Note that most datasets have small uncensoring ratios, representing challenging survival analysis problems. The next 3 columns of this table report the C-index values obtained by the 3 RSF variants on the 11 datasets. In this and other tables reporting C-index values in this chapter, the last row shows the average rank obtained by each RSF variant (the lower the rank the better the result); and the best result (highest C-index) for each dataset is shown in boldface font.

The RSF Log-rank variant, the original one (with the default Log-rank node-splitting criterion), obtained the best average rank (1.82), and it achieved the highest C-index in 5 of the 11 datasets. The second best RSF variant was the one using the Wilcoxon node-splitting criterion, which obtained an average rank of 2.0 and the best C-index in 4 datasets. The RSF Tarone-Ware variant obtained the worst results, with an average rank of 2.18, and only 2 wins.

We applied the non-parametric Friedman test (Demšar, 2006) to determine whether or not there is a statistically significant difference between the average ranks of the three RSF variants and the mean rank of 2.0 under the null hypothesis. The calculated value of F_F is 0.342. With 3 variants and 11 datasets, F_F is distributed according to the f distribution with 3 - 1 = 2 and (3-1) x (11-1) = 20

Table 23: C-index values of three RSF variants with different node-splitting criteria. All RSF variants had their *mtry* and *d0* hyper-parameters tuned via nested cross-validation

Dataset (Disease)	RSF Log-rank	RSF Wilcoxon	RSF Tarone-Ware
Alzheimer	0.7725	**0.776**	0.7736
Angina	**0.6018**	0.6002	0.6013
HeartAtt	0.6351	**0.6370**	0.6343
Psychiatric	0.5372	0.541	**0.5462**
Stroke	**0.6373**	0.635	0.6335
Diabetes	0.7527	**0.7542**	0.7516
Cancer	**0.5473**	0.5436	0.5382
Arthritis	0.5340	**0.5383**	0.5380
Any-disease (ELSA)	0.5426	0.5425	**0.5455**
Any-disease (SHARE)	**0.6890**	0.6882	0.6883
HD	**0.5778**	0.5216	0.5358
Average Rank	**1.82**	2.00	2.18

degrees of freedom. The critical value of F(2,20) for $\alpha = 0.05$ is 3.493. Note that F_F is smaller than the critical value (p-value = 0.695), and so the null hypothesis cannot be rejected. Hence, there is no statistical evidence to support the claim that any of the three RSF variants have better predictive performance than the others.

5.5.2 Comparing the results of RSF with five different leaf-node-prediction criteria and standard node-splitting criterion

In the previous subsection, the choice of node-splitting criterion did not significantly affect the C-index values, when the RSF algorithm was using the standard leaf-node-prediction criterion (computing the CHF). This subsection investigates the complementary issue of whether the choice of leaf-node-prediction criterion affects the C-index, by fixing the node-splitting criterion to the Log-rank test, the original criterion, which is the most popular criterion and obtained the best

Table 24: C-index values of five RSF variants with different constant-hazard-mean criteria. All RSF variants had their *mtry* and *d0* hyper-parameters tuned via nested cross-validation

Dataset		RSF (Original)	RSF Naive-mean	RSF constant-hazard -mean	RSF Weight-age -mean	RSF KNN-mean
Disease	uncensoring ratio	c-index	c-index	c-index	c-index	c-index
Alzheimer	69/6825 (1.0%)	0.7725	0.7861	0.7564	**0.7919**	0.7906
Angina	165/6488 (2.5%)	0.6018	0.5911	**0.6085**	0.6015	0.5915
HeartAtt	186/6607 (2.8%)	0.6351	0.6347	**0.651**	0.6458	0.6348
Psychiatric	219/5972 (3.5%)	0.5372	0.544	0.5596	**0.5677**	0.5494
Stroke	270/6632 (4.1%)	0.6373	0.6456	0.6425	0.6401	**0.6528**
Diabetes	416/6500 (6.4%)	0.7527	0.6959	**0.7594**	0.7152	0.704
Cancer	562/6386 (8.8%)	0.5473	0.5309	**0.5553**	0.5513	0.5321
Arthritis	784/4276 (18.3%)	0.5340	0.5081	**0.5462**	0.5381	0.5189
Any-disease (ELSA)	979/3280 (29.8%)	0.5426	0.5307	**0.5616**	0.5554	0.5306
Any-disease (SHARE)	101300/139522 (72.6%)	0.6890	0.7009	**0.7104**	0.6968	0.7012
HD	783/1097 (71.4%)	0.5778	0.5622	**0.5934**	0.5741	0.5696
Average Rank		3.36	4.18	**1.63**	2.45	3.36

predictive accuracy in the previous subsection – as reported in Table 23.

Table 24 shows the C-index values obtained by the original RSF with the Log-rank test (Ishwaran et al., 2008) and the four proposed RSF variants with different modified leaf-node-prediction criteria, focusing on estimating the survival times. Again, the first two columns of this table show the disease used as the target 'survival time' variable and the uncensoring ratio for each dataset; with the next 5 columns reporting the C-index values of the 5 RSF variants being compared.

The proposed RSF constant-hazard-mean obtained overall the best predictive performance (highest C-index values) among the 5 RSF variants, having 8 wins across the 11 datasets. The average rank of RSF constant-hazard-mean (1.63)

is much lower (better) than that of RSF Weight-age-mean, the runner-up with 2 wins across the 11 datasets, which had an average rank of 2.45. Moreover, both methods outperformed the original RSF (where the leaf nodes compute the CHF), the most popular RSF variant in the literature.

The non-parametric Friedman test (Demšar, 2006) was used to determine whether or not there is a significant difference between the average ranks of the five RSF variants and the mean rank of 3.0 under the null hypothesis. The calculated value of F_F is 6.176. With 5 variants and 11 datasets, F_F is distributed according to the f distribution with 5 - 1 = 4 and (5-1) x (11-1) = 40 degrees of freedom. The critical value of F(4,40) for $\alpha = 0.05$ is 2.606. Hence, F_F is greater than the critical value (p-value = 0.002), and so the null hypothesis can be rejected. Hence, there is strong statistical evidence to support the claim that there is a significant performance difference among these 5 RSF variants.

Therefore, we proceed with the Holm post-hoc test (Demšar, 2006), which compares the average rank of the best (control) method, viz. constant-hazard-mean RSF, against each of the other four methods, by adjusting the significance level of $\alpha = 0.05$ to compensate for multiple comparisons. The statistical results can be reported as follows:

- RSF constant-hazard-mean significantly outperformed RSF Naive-mean with p-value = 0.00008, smaller than adjusted $\alpha = 0.0125$. The null-hypothesis is rejected.
- RSF constant-hazard-mean significantly outperformed RSF KNN-mean with p-value = 0.005, smaller than adjusted $\alpha = 0.017$. The null-hypothesis is rejected.
- RSF constant-hazard-mean significantly outperformed the original RSF with p-value = 0.005, smaller than adjusted $\alpha = 0.025$. The null-hypothesis is rejected.

- comparing RSF constant-hazard-mean to RSF Weight-age-mean, the statistical test shows that the p-value = 0.112 is greater than adjusted $\alpha = 0.050$. Hence, the null hypothesis cannot be rejected.

Hence, based on the results of the Holm post-hoc test, RSF constant-hazard-mean was significantly better than 3 of the other RSF variants, whilst there was no significant difference between RSF constant-hazard-mean and RSF Weight-age-mean.

5.5.3 Comparing results of RSF variants with three different node-splitting criteria and the best proposed leaf-node-prediction criterion

Subsection 5.5.1 reported the results of RSF variants with different node-splitting criteria when using the standard leaf-node prediction criterion (CHF). This subsection investigates a complimentary issue. We conducted an experiment to compare results obtained by RSF variants with different node-splitting criteria when using the best modified leaf-node-prediction criterion, based on the results in Table 24 – i.e., the constant-hazard-mean criterion. The 3 node-splitting criteria used in this subsection are the same as the ones used in Subsection 5.5.1.

Table 25 reports the C-index values obtained by the 3 different RSF variants with the proposed constant-hazard-mean RSF leaf-node-prediction criterion. Similarly to the results reported in Table 23, the RSF variant with Log-rank obtained the best average rank (1.81), and also the largest number of wins (in 6 out of the 11 datasets).

Applying the Friedman test, the calculated value of F_F is 0.283. Therefore, the null hypothesis cannot be rejected – p-value = 0.739. Hence, there is no statistical evidence to support the claim that any of the 3 variants of the constant-hazard-mean RSF (varying the node-splitting criteria) has better performance than the

Table 25: C-index values of three RSF variants of constant-hazard-mean with different node-splitting criteria. All RSF variants had their *mtry* and *d0* hyper-parameters tuned via nested cross-validation

Dataset (Disease)	constant-hazard -mean Log-rank	constant-hazard -mean Wilcoxon	constant-hazard -mean Tarone-Ware
Alzheimer	0.7564	**0.7738**	0.7576
Angina	**0.6085**	0.6054	0.6073
HeartAtt	0.651	0.6537	**0.6538**
Psychiatric	**0.5596**	0.5539	0.557
Stroke	**0.6425**	0.6424	0.6417
Diabetes	0.7594	**0.7641**	0.7611
Cancer	**0.5553**	0.5476	0.5532
Arthritis	**0.5462**	0.5423	0.5458
Any-disease (ELSA)	**0.5616**	0.5578	0.5583
Any-disease (SHARE)	0.7104	**0.7126**	**0.7126**
HD	0.5934	**0.6054**	0.5907
Average Rank	**1.81**	2.14	2.05

others.

5.5.4 Comparing the best RSF variant (RSF-constant-hazard-mean) against Cox Proportional Hazards regression

Table 26 reports the C-index values for the standard Cox's Proportional Hazard (PH) Regression and the proposed RSF variant with constant-hazard-mean (using the default Log-rank test for node splitting), which achieved the best predictive performance among all five RSF variants in Table 24 and all three RSF variants in Table 25. The results are shown for each disease used as the target 'survival' variable. Recall that the large majority of instances were censored in all datasets, except in the SHARE and HD datasets.

As shown in Table 26, RSF-constant-hazard-mean outperforms the Cox regression in 10 out of the 11 datasets. The difference in the C-index values of these two methods is particularly large in the Alzheimer dataset (a difference of 13.9%); and

Table 26: C-index values obtained by Cox's PH regression and RSF-constant-hazard-mean

Dataset (Disease)	CoxPH	RSF-constant-hazard-mean
Alzheimer	0.6175	**0.7564**
Angina	0.5672	**0.6085**
HeartAtt	0.6219	**0.6510**
Psychiatric	0.5333	**0.5596**
Stroke	0.5952	**0.6425**
Diabetes	0.7300	**0.7594**
Cancer	0.5163	**0.5553**
Arthritis	0.5399	**0.5462**
Any-disease (ELSA)	0.5345	**0.5616**
Any-disease (SHARE)	**0.7242**	0.7104
HD	0.5736	**0.5934**
Average Rank	1.91	**1.09**

the difference is about 4% or 5% in a couple of other datasets, namely: Angina (a difference of 4.1%) and Stroke (4.7%).

In addition, the average rank of RSF-constant-hazard-mean (1.09) is remarkably lower (better) than that of the Cox model (1.91). Applying the Wilcoxon signed-ranks test, the null hypothesis of equal rank for both methods is rejected at the usual 5% significance level with a p-value of 0.003. Hence, there is statistical evidence supporting the conclusion that, overall, the RSF-constant-hazard-mean performed significantly better than the Cox PH regression across these 11 survival datasets.

5.5.5 Most Frequently Selected Hyper-parameter Values

Table 27 reports the hyper-parameter settings most frequently selected by the nested cross-validation procedure (which was used for hyper-parameter optimisation). Recall that the nested cross-validation procedure selected one configuration of hyper-parameter settings for each RSF variant for each of the training sets of

the external cross-validation, for each dataset. Hence, among those selected hyper-parameter settings per dataset, the most frequently selected one is reported in the table.

Recall also that all RSF variants had two hyper-parameters optimised: *mtry* and d_0 (the minimum number of uncensored instances per leaf node). On one hand, the candidate *mtry* values for all RSF variants exactly were the same as that for all RF variants in Section 4.4; *mtry*= 4, 7, 10 or 13 for the nine ELSA datasets, *mtry*= 3, 4, 6, 8 for the SHARE dataset, and *mtry*= 5, 6, 8, 10 for the Haemodialysis dataset. On the other hand, the candidate d_0 values were noticeably smaller than the candidate node size values for RF variants, since in the context of RSFs the d_0 hyper-parameter refers to uncensored instances. We used the candidate values $d_0 = 1$, 2 or 3 for all datasets.

As shown in Table 27, the most frequently selected hyper-parameter setting combination was (*mtry* = 4, node $d_0 = 3$), which was the winner 11 times out of 22 (considering the 2 RSF methods and the 11 datasets). In addition, both RSF methods consistently chose the (4,3) combination in 4 datasets: Heart Attack, Psychiatric, Stroke and Any-disease ELSA.

Moreover, considering the selection frequency of each hyper-parameter setting separately, the *mtry* value of 4 was particularly effective: it was selected 13 out of 22 times – or more precisely out of the 20 cases where it could possibly be selected, since the value 4 was not a candidate *mtry* value in the HD dataset. This *mtry* value was the smallest of the 4 candidate values in the ELSA datasets, and the second smallest candidate *mtry* value in the SHARE dataset. Hence, these results show that, in these datasets, smaller *mtry* values (which tend to increase the diversity of trees in the forest) are more effective in general.

In addition, the d_0 value of 3 was also very effective by itself; it was selected in 17 out of 22 times. This value of 3 was the largest of the 3 candidate d_0 values. Hence, it seems that it is better to use relatively large d_0 values for these datasets,

Table 27: Most Frequently Selected Hyper-parameter Settings, for each RSF variant and each dataset

Dataset		RSF	RSF-constant -hazard-mean
Disease	uncensoring ratio	(mtry/d_0)	(mtry/d_0)
Alzheimer	69/6825 (1.0%)	(10, 1)	(13, 3)
Angina	165/6488 (2.5%)	(13, 1)	(4, 3)
HeartAtt	186/6607 (2.8%)	(4, 3)	(4, 3)
Psychiatric	219/5972 (3.5%)	(4, 3)	(4, 3)
Stroke	270/6632 (4.1%)	(4, 3)	(4, 3)
Diabetes	416/6500 (6.4%)	(13, 3)	(4, 1)
Cancer	562/6386 (8.8%)	(13, 3)	(4, 3)
Arthritis	784/4276 (18.3%)	(4, 3)	(4, 1)
Any-disease (ELSA)	979/3280 (29.8%)	(4, 3)	(4, 3)
Any-disease (SHARE)	101300/139522 (72.6%)	(3, 3)	(3, 3)
HD	783/1097 (71.4%)	(10, 3)	(10, 1)

so that the number of uncensored instances in each leaf node is large enough to allow the survival trees in the RSF to generalise well from those instances.

5.5.6 Computational Runtimes

The performance of the proposed RSF variants primarily depends on the size of the dataset. As a result, when comparing different variants using the same dataset, their runtimes are generally similar.

Regarding dataset size, the datasets derived from the ELSA database share the same feature set and exhibit relatively similar numbers of instances, ranging from 3,000 to 7,000. Therefore, the experiment runtimes for each dataset were not much different, varying from about 2 to 3 days to run a nested cross-validation for each RSF variant, for each dataset. It should be noted that all experiments were conducted using the Myrtle computer cluster (with 80 CPU threads distributed inside the department) hosted by the University of Kent. In contrast, the SHARE dataset comprises over 100,000 instances, leading to experiment durations of approximately 14 days (2 weeks) for the nested cross-validation for each RSF variant. On the other hand, the haemodialysis dataset achieved the fastest experiment results, completing a nested cross-validation for each RSF variant in just over a day.

5.6 Conclusion

This Chapter introduced several variants of Random Survival Forests (RSFs) to improve the algorithm for survival analysis tasks using a machine learning approach. To evaluate these proposed RSF variants, we compared them against the standard RSF and the Cox Proportional Hazards (PH) regression using 11 real-world biomedical datasets. The predictive accuracy, measured by C-index values, was calculated in four different types of experiments.

The first experiment compared three RSF variants using different node-splitting criteria, with all variants employing the standard Cumulative Hazard Function for leaf-node-prediction. The Log-rank test delivered the best predictive accuracy, but no statistically significant difference was observed among the three variants.

The second experiment compared five RSF variants with different leaf-node-prediction criteria, all using the Log-rank test. In this experiment, the RSF with the "constant-hazard-mean" criterion demonstrated the highest predictive accuracy across all datasets. Statistically, it was found to be significantly better than three out of the four other RSF variants.

In the third experiment, we compared the three RSF variants using different node-splitting criteria once again, but this time using the "constant-hazard-mean" for the leaf-node-prediction, as it had shown the best results previously. The RSF variant with Log-rank criterion performed the best, with no statistically significant difference observed among the three variants.

The fourth and final experiment compared the RSF variant with the best performance across all previous experiments (Log-rank test and constant-hazard-mean predictions) against the well-known Cox Proportional Hazards (PH) regression. The RSF variant outperformed the Cox PH regression in 10 out of 11 datasets, a statistically significant result.

In summary, this study suggests that a new version of the RSF algorithm, particularly using the Log-rank test for node-splitting and the "constant-hazard-mean" criterion for leaf-node-prediction, tends to improve the predictive accuracy in survival analysis tasks.

Chapter 6

New Variants of Deep Survival Forests

6.1 Introduction

This section will briefly recap the main parts of the Deep Survival Forest (DSF) algorithm (described in Subsection 2.4.4) that will be modified by the proposed versions of DSFs in this chapter.

Before discussing the DFS algorithm, it is worth recalling the basic characteristics of the Deep Forest algorithm, which was designed for supervised learning in general, not survival analysis. (Zhou and Feng, 2019) have designed the Deep Forest algorithm based on some principles of deep neural networks, which are a state-of-the-art type of supervised learning method in general. The principles taken into account were in particular layer-by-layer processing, in-model feature transformation (or construction) and sufficient model complexity to learn complex relationships in the data.

Essentially, (Zhou and Feng, 2019) proposed the following principles for the Deep Forests (DF) algorithm. First, a set of Random Forests (RFs) are organized into a sequence of layers similar to that in deep neural networks, where each layer

consists of RFs rather than neurons. Second, the values of the target variable predicted by the RFs in one layer are fed forward to the next layer as additional features. In other words, these additional features are concatenated with the original features, and this concatenated set of features is used as input for the RFs in the next layer. The number of additional feature sets created for the next layer is equal to the number of RFs in the current layer — i.e., the set of predictions output by each RF will form one set of additional features. Note that the additional features are fed forward only to the next layer. Note also that, for classification tasks, a set of additional features contains L features, where L is the number of class labels. More specifically, each feature contains the probability of a given class label.

A few studies have adopted the Deep Forest algorithm for classification tasks, and new algorithms have been built upon it (Utkin et al., 2019a; Ganaie, Hu et al., 2021; Chen et al., 2020). Most importantly, (Utkin et al., 2020, 2021) developed a new variant of Deep Forest which is particularly relevant to this thesis, and is the basis for the algorithms proposed in this chapter. Specifically, they extended the Deep Forest algorithm for survival analysis tasks, proposing the Deep Survival Forest (DSF) algorithm. In this algorithm, each layer consists of Random Survival Forests (RSFs), rather than Random Forests as in the original Deep Forest algorithm.

In the experiments reported by (Utkin et al., 2020, 2021), DSF was shown to perform as well as Random Survival Forests, achieving high predictive accuracy as measured by the C-index.

It should be noted that the DSF algorithm was designed specifically for survival analysis based on RSFs, i.e., predicting the "Ensemble Mortality" which is the type of outcome predicted by the RSF. As discussed in Section 2.4.3, Ensemble Mortality represents the expected rate of deaths or the expected cumulative hazard rate as defined in Equation 24. Again, instead of focusing on the survival times

of the instances, the DSF method uses the Nelson-Aalen method to estimate an ensemble Cumulative Hazard Function (CHF), a sum of the "failure rates" across all observed failure times. That is, it was not designed to directly answer the fundamental question about how long a subject will "survive", i.e. how much time will pass until the event of interest occurs for a given subject. This limitation of the concepts of Ensemble Mortality and CHF is a motivation for the proposed variations of DSF described in the next Section.

6.2 Description of the new variations of Deep Survival Forests

As mentioned earlier, since the DSF algorithm is based on predicting Ensemble Mortality, it is not conceptually suitable for the goal of predicting the survival time for a specific individual, i.e., how long until the event of interest will occur to a given individual.

To cope with this issue, recall that Section 5.3.2 has proposed several variants of the leaf-node-prediction criteria of RSFs, inspired by the standard Random Forest algorithm for regression (rather than for survival analysis), where the value predicted at a leaf node is an estimate of the mean of the target variable over the instances at that leaf node, by taking into account censored data.

Therefore, we also employ this proposed modification of the RSF algorithm in the context of the DSF algorithm. That is, we modify the DSF algorithm by replacing the base RSF algorithm with our proposed variations of RSF described in Section 5.3.2, based on some variants of the "mean-at-leaf" prediction criteria.

Out of the four proposed "mean-at-leaf" prediction criteria, we choose two variants in order to conduct experiments in this chapter, as follows.

The first variant is the RSF with a new mean formula based on the assumptions of constant hazard rates and non-informative censoring since this variant produced

the highest predictive accuracy overall in the experiments reported in Chapter 5. Recall that this constant-hazard-mean criterion estimates a mean survival time at each leaf node of the survival trees in a RSF model using Equation (35), where the summation of all survival times, censored and uncensored included, is divided by the number of uncensored instances.

The second chosen variant is the RSF variant with KNN-mean replacing the base RSF in the DSF algorithm, due to the KNN-mean criterion being the most sophisticated technique among all the proposed leaf-node-prediction criteria. This criterion makes no strong assumptions about the survival distribution of the dataset, and no assumption of constant hazard rate. As described earlier, unlike the constant-hazard-mean approach, this technique is designed such that it uses the K-Nearest Neighbour (K-NN) algorithm, a non-parametric machine learning method, within the procedure for computing the predicted value at each leaf node. The idea is to use K-NN to estimate the survival time of censored individuals, i.e. imputing target values based on K-NN's predictions. Recall that K-NN is used only as part of the leaf-node prediction criterion of RSF. For details of the K-NN leaf-node-prediction criterion, see Algorithm 4 and Section 5.3.2 in Chapter 5.

6.3 Experimental Methodology

This section describes the experimental methodology used for obtaining the computational results reported in this chapter; and it is divided into four parts, as follows. The first subsection provides a summarised description of the datasets used in the experiments. The second subsection mentions the predictive performance measure used. The third subsection describes the hyper-parameter tuning procedure. Finally, the fourth subsection mentions the statistical significance tests used.

6.3.1 Datasets used in the experiments

The experiments used 11 datasets for 11 different age-related diseases (i.e. 11 separate survival prediction problems). 10 of these datasets were created from two different surveys, the English Longitudinal Study of Ageing (ELSA) (Clemens et al., 2019) and the Survey of Health Ageing and Retirement in Europe (SHARE) (Börsch-Supan et al., 2013). The other dataset involves the prediction of the survival time of patients undergoing haemodialysis. The creation of these datasets was described in detail in Chapter 3.

Recall that 9 out of the 11 datasets are constructed from the ELSA data, containing between 3,000 and 7,000 instances (depending on the target variable), with exactly the same 44 predictive features, but different target variables. On the other hand, the dataset constructed from the SHARE data is much larger, containing almost 140,000 instances but only 15 predictive features. In essence, the instances represent individuals (subjects) in these surveys, and the target variables represent the 'survival times', more precisely, the time passed (in months) until an individual is diagnosed with a certain disease (for 8 datasets) or any of several diseases (for two datasets); whilst the predictive features represent biomedical information collected by nurses or other relevant characteristics of an individual (age and gender). Lastly, we have the Hemodialysis dataset which contains 1,097 instances with 38 features.

6.3.2 Predictive Performance Measure

Similarly to the two previous chapters, in the experiments being reported in this chapter, the predictive performance of the learned survival models was estimated by the concordance index (C-index), which is a popular measure accounting for censored data. Recall that the C-index can be interpreted as the probability of correctly ordering the predicted survival values for a randomly chosen pair of subjects whose actual survival times are different. For a precise formal definition

of the C-index, the reader is referred to Section 2.2.5 in Chapter 2.

6.3.3 Hyper-Parameter Tuning via Nested Cross-Validation

Similarly to the two previous chapters, all experiments are performed using a nested cross-validation procedure, where an inner cross-validation performs hyper-parameter tuning and an outer cross-validation estimates the predictive performance of the survival models. For a more detailed description of this nested cross-validation procedure, the reader is referred to Subsection 4.4.3.

Similarly to the previous two chapters, we used 5 folds for the inner cross-validation for all datasets, whilst the number of folds for the outer cross-validation was set to 10 for the ELSA datasets and the Haemodialysis dataset, and set to 5 for the SHARE dataset. The latter has fewer folds to save computational time, since the SHARE dataset is much larger than the other datasets.

The internal cross-validation is used to tune three hyper-parameters of the DSF algorithm, namely: (a) n_rfs_layer, i.e., the number of RSFs in each layer of the cascade; (b) n_fail_layers, i.e., the maximum number of rounds allowed for the training process to terminate when the validation performance on the training set fails to improve compared against the best validation performance achieved so far; and (c) *d0*, i.e., the minimum number of uncensored instances required at each leaf node of the trees in RSFs learned by the DSF algorithm.

According to (Zhou and Feng, 2019), both n_rfs_layer and n_fail_layers have been recognized as the most influential hyper-parameters in general in the original Deep Forest algorithm – which was designed for general supervised learning rather than survival analysis. Intuitively, these hyper-parameters are also important in DSFs, in the context of survival analysis, since they control the complexity (size) of the model in terms of the number of RSFs per layer and number of layers (note that n_fail_layers influences the number of layers added to the cascade).

In addition, *d0* can be seen as the survival task-related counterpart of the hyper-parameter *node size* in classical random forests.

For each hyper-parameter, there are predefined candidate values, with the same candidate values being available for all 11 datasets, as follows. First, we consider two candidate values for *d0*, namely 1 and 3. Second, n_fail_layers has 2 candidate values, 1 and 2. Basically, we would like to test whether or not giving a chance of failure to improve the accuracy would lead to a better predictive performance in the end. Third, the set of candidate values for n_rfs_layer is 1, 2 and 4, where the total number of trees across all forests in a layer is fixed to 500 per layer. This means that if $n\_rfs\_layer = 1$, then there are 500 trees in the sole RF in each layer. If $n\_rfs\_layer = 2$, then each RF contains 250 trees. If $n\_rfs\_layer = 4$, then each RF contains 125 trees.

Overall, there are 12 candidate configurations of the hyper-parameter setting for the DSF experiment: 3 candidate n_rfs_layer values times 2 candidate n_fail_layers values times 2 candidate *d0* values.

All computational experiments were performed using Python 3 with the scikit-survival library version 0.14.0 (Pölsterl, 2020), a Python module built on top of the scikit-learn machine learning library (Pedregosa, Varoquaux et al., 2011), together with (Weinstein et al., 2019). In addition, some parts of the program were written and customised in Cython-code, which played an important role in boosting the performance of RSF due to Python's relatively slow performance.

6.3.4 Statistical significance tests

In Subsection 4.4.4 we described the statistical tests of significance which were used for comparing the performance of more than two methods in the previous two chapters. In the statistical analysis of results in this current chapter, we used not only those tests, but also the Wilcoxon Signed-Rank test, which was used when directly comparing the performance of only two methods. Hence, this test

is described next.

The Wilcoxon-Signed Ranks Test

The Wilcoxon signed-ranks test (Wilcoxon, 1945) is a non-parametric statistical significance test used in this thesis for comparing the predictive accuracies (more precisely, C-index values) of two algorithms. The main advantage of this test is its non-parametric nature, making no assumption of normal distribution (Japkowicz and Shah, 2011), which is a strong assumption made in particular by the alternative paired t-test. Another advantage of the Wilcoxon signed-ranks test is its robustness against outliers, since it is based on the relative ranks of the predictive performances of two models, instead of being based on their raw performance such as the raw C-index values.

The null hypothesis for this test is that the medians of the two learned models' predictive performances are equal.

The Wilcoxon signed-ranks test involves several steps (Demšar, 2006), as follows. To begin, the difference (d_i) between the predictive accuracies of the two learned models is calculated for each i–th dataset, $i = 1, \ldots, N$, where N is the number of datasets. Next, the differences are ranked according to their absolute values (rank(d_i), i = 1,. . . ,N,), ignoring their signs; in the case of a tie, the corresponding average rank is assigned. Once the data have been prepared, we start to calculate the Wilcoxon signed-rank sums. The calculations proceed separately according to Equations 39 and 40 for the positive and negative differences in accuracy, respectively. That is, R^+ denotes the sum of ranks for positive differences and R^- denotes the sum of ranks of negative differences. It should be noted that the differences of 0 have their ranks split evenly among the sums; if there is an odd number of them, one is discarded.

$$R^+ = \sum_{d_i>0} \text{rank}\,(d_i) + \frac{1}{2} \sum_{d_i=0} \text{rank}\,(d_i) \tag{39}$$

$$R^- = \sum_{d_i<0} \text{rank}\,(d_i) + \frac{1}{2}\sum_{d_i=0} \text{rank}\,(d_i) \tag{40}$$

Afterwards, the smaller of R^+ or R^- is used for the test statistic, T. Let T and $T_{critical}$ be the smaller of the rank sums and the exact critical value respectively. The null hypothesis is rejected if T is greater than or equal to $T_{critical}$, accepted otherwise. In general, the exact value of $T_{critical}$ can be found in a precomputed table (available e.g. in (Bruning and Kintz, 1987)) for values of N up to 25. For a larger number of datasets, the distribution of the test statistic can be approximated by a normal distribution, with the following equation for calculating the *z-score*:

$$z = \frac{T - \frac{1}{4}N(N+1)}{\sqrt{\frac{1}{24}N(N+1)(2N+1)}} \tag{41}$$

where T is the test statistic and N is the number of datasets. Subsequently, the null hypothesis is rejected if z is smaller than the critical value for *z-score*.

6.4 Computational Results

This section reports the computational results obtained by the proposed Deep Survival Forest (DSF) variants and the baseline methods, including the results of statistical significance tests.

This section is divided into four subsections, as follows. The first subsection reports results comparing the original DSF algorithm against the original Random Survival Forest (RSF) algorithm. The second subsection reports results comparing three different variants of the DSF algorithm with different leaf-node-prediction criteria. The third subsection reports results comparing DSF-KNN-mean (the DSF variant which achieved the best results in the second subsection) against the very popular Cox Proportional Hazards regression. Finally, the fourth subsection

Table 28: Comparison of the predictive performance of the Random Survival Forest (RSF) and that of the Deep Survival Forest (DSF)

Dataset		RSF	DSF
Disease	uncensoring ratio	c-index	c-index
Alzheimer	69/6825 (1.0%)	**0.7725**	0.7670
Angina	165/6488 (2.5%)	0.6018	**0.6151**
HeartAtt	186/6607 (2.8%)	0.6351	**0.6417**
Psychiatric	219/5972 (3.5%)	0.5372	**0.5578**
Stroke	270/6632 (4.1%)	**0.6373**	0.6335
Diabetes	416/6500 (6.4%)	0.7527	**0.7555**
Cancer	562/6386 (8.8%)	0.5473	**0.5493**
Arthritis	784/4276 (18.3%)	0.5340	**0.5418**
Any-disease (ELSA)	979/3280 (29.8%)	0.5426	**0.5557**
Any-disease (SHARE)	101300/139522 (72.6%)	0.6890	**0.7026**
HD	783/1097 (71.4%)	**0.5778**	0.5776
Average Rank		1.73	**1.27**

reports results comparing DSF-KNN-mean, which obtained the best results across all the previous subsections of this chapter, against RSF-constant-hazard-mean, which obtained the best results in Chapter 5.

6.4.1 Comparing the results of Deep Survival Forest (DSF) with Random Survival Forest (RSF)

Table 28 compares the C-index values obtained by the original RSF algorithm (Ishwaran et al., 2008) and the original DSF algorithm (Utkin et al., 2020), breaking down by each disease used as the target 'survival' variable. In this and other tables reporting C-index values in this chapter, the best result (highest C-index) for each dataset is shown in boldface font; and the last row shows the average rank (across all datasets) obtained by each method – the lower the rank the better the result.

It can be seen that DSF outperformed the original RSF in 8 out of 11 datasets, although the differences in the C-index values are in general small. The three highest C-index values across all datasets are 0.7725, 0.7555 and 0.7026, obtained in the Alzheimer (by RSF), Diabetes (by DSF) and SHARE (by DSF) datasets, respectively.

In addition, the average rank of DSF (1.27) is notably lower (better) than that of the original RSF (1.73). Applying the Wilcoxon signed-ranks test, the null hypothesis of equal rank for both methods is rejected at the usual 5% significance level with a p-value of 0.042. Hence, there is statistical evidence supporting the conclusion that, overall, the original DSF performed significantly better than the original RSF in these 11 survival datasets.

To further analyse these results, we tried to detect some association between high/low C-index values and high/low uncensoring ratios across datasets. The uncensoring ratio is simply the ratio of the number of uncensored instances over the total number of instances in a dataset. Uncensoring ratios are reported in the second column of Table 28 – as noted in previous chapters, most datasets have a very small uncensoring ratio (or equivalently, a very high censoring ratio), making the problem of survival-time prediction particularly challenging.

The three datasets with the highest C-index values had a great difference between their uncensoring ratios: just 1% for Alzheimer, 6.4% for Diabetes and 72.6% for Any-disease (SHARE). In addition, the three datasets with the lowest C-index values also had substantially different uncensoring ratios: 3.5% for Psychiatric, 8.8% for Cancer and 18.3% for Arthritis, respectively. Hence, there is no clear association between the predictive performance (C-index) of the two methods and the uncensoring ratio of the datasets.

6.4.2 Comparing the results of DSF with three different leaf-node-prediction criteria

Table 29 reports the C-index values obtained by the original DSF (Utkin et al., 2020) and the two proposed DSF variants with different modified leaf-node-prediction criteria, focusing on estimating the survival time of each individual (instance). The first column of this table shows the disease used as the target 'survival time' variable, and the next 3 columns report the C-index values of the 3 DSF variants being compared.

Overall, both the newly proposed DSF variants outperform the original DSF. The DSF-KNN-mean variant obtained the best average rank (1.82), and it achieved the highest C-index in 6 of the 11 datasets. The DSF-constant-hazard-mean comes second with an average rank of 2.0 and the best C-index in the 3 datasets. The original DSF variant obtained the worst results, with the worst average rank of 2.18 and only 2 wins.

We applied the non-parametric Friedman test (Demšar, 2006) to determine whether or not there is a statistically significant difference between the average ranks of the three DSF variants and the mean rank of 2.0 under the null hypothesis. The calculated value of F_F is 0.342. With 3 variants and 11 datasets, F_F is distributed according to the f distribution with 3 - 1 = 2 and (3-1) x (11-1) = 20 degrees of freedom. The critical value of F(2,20) for $\alpha = 0.05$ is 3.493. Note that

Table 29: C-index values of three DSF variants with different leaf-node-prediction criteria. All DSF variants had their *n_rfs_layer*, *n_fail_layers* and *d0* hyper-parameters tuned via nested cross-validation

Dataset (Disease)	DSF	DSF (constant-hazard-mean)	DSF (KNN-mean)
Alzheimer	0.767	0.7644	**0.7899**
Angina	**0.6151**	0.6087	0.6036
HeartAtt	0.6417	**0.6483**	0.6384
Psychiatric	0.5578	0.5521	**0.5891**
Stroke	0.6335	0.6436	**0.6504**
Diabetes	0.7555	**0.7649**	0.7060
Cancer	0.5493	0.5536	**0.5621**
Arthritis	0.5418	0.5468	**0.5582**
Any-disease (ELSA)	0.5557	0.5607	**0.5666**
Any-disease (SHARE)	0.7026	**0.7136**	0.7000
HD	**0.5776**	0.5742	0.5749
Average Rank	2.18	2.00	**1.82**

F_F is smaller than the critical value (p-value = 0.695), and so the null hypothesis cannot be rejected. Hence, there is no statistical evidence to support the claim that any of the three DSF variants have better predictive performance than the others.

6.4.3 Comparing the results of DSF with standard Cox Proportional Hazards regression

Table 30 reports the C-index values for Cox's Proportional Hazard (PH) Regression and the proposed DSF variant with KNN-mean, which achieved the best predictive performance among all three DSF variants in Table 29. The results are shown for each disease used as the target 'survival' variable. Recall that the large majority of instances were censored in all datasets, except in the SHARE and HD datasets.

As shown in this table, DSF-KNN-mean outperforms the Cox PH model in 9 out of the 11 datasets. The difference in the C-index values of these two methods

Table 30: C-index values obtained by Cox's PH regression and DSF with KNN-mean

Dataset (Disease)	CoxPH	DSF (KNN-mean)
Alzheimer	0.6175	**0.7899**
Angina	0.5672	**0.6036**
HeartAtt	0.6219	**0.6384**
Psychiatric	0.5333	**0.5891**
Stroke	0.5952	**0.6504**
Diabetes	**0.7300**	0.7060
Cancer	0.5163	**0.5621**
Arthritis	0.5399	**0.5582**
Any-disease (ELSA)	0.5345	**0.5666**
Any-disease (SHARE)	**0.7242**	0.7000
HD	0.5736	**0.5749**
Average Rank	1.82	**1.18**

is particularly large in the Alzheimer dataset (a difference of 17.2%); and the difference is about 5% in a few other datasets, namely: Psychiatric (a difference of 5.6%), Stroke (5.5%) and Cancer (4.6%).

In addition, the average rank of DSF-KNN-mean (1.18) is remarkably lower (better) than that of the Cox model (1.82). Applying the Wilcoxon signed-ranks test, the null hypothesis of equal rank for both methods is rejected at the usual 5% significance level with a p-value of 0.032. Hence, there is statistical evidence supporting the conclusion that, overall, the DSF variant with KNN-mean performed significantly better than the Cox's PH regression across these 11 survival datasets.

6.4.4 Comparing the results of the DSF method with KNN-mean against the constant-hazard-mean RSF method

Table 31 reports the C-index values for the pair of our best proposed methods, namely RSF-constant-hazard-mean and DSF-KNN-mean, which achieved the best

Table 31: C-index values obtained by constant-hazard-mean RSF and DSF with KNN-mean

Dataset (Disease)	RSF-constant-hazard-mean	DSF-KNN-mean
Alzheimer	0.7564	**0.7899**
Angina	**0.6085**	0.6036
HeartAtt	**0.6510**	0.6384
Psychiatric	0.5596	**0.5891**
Stroke	0.6425	**0.6504**
Diabetes	**0.7594**	0.7060
Cancer	0.5553	**0.5621**
Arthritis	0.5462	**0.5582**
Any-disease (ELSA)	0.5616	**0.5666**
Any-disease (SHARE)	**0.7104**	0.7000
HD	**0.5934**	0.5749
Average Rank	1.55	**1.45**

predictive performance from Chapter 5 and this chapter, respectively. Both methods have their leaf-node-prediction criteria modified to compute the mean survival times. Note that this comparison of the results of RSF-constant-hazard-mean and DSF-KNN-mean is intended to conclude what was the best method overall, out of all RSF and DSF variants proposed in this thesis and all standard (baseline) methods included in the thesis' experiments. To draw this conclusion, it is enough to compare the best methods in Chapters 5 and 6, because the best method in Chapter 4, the standard RSF, had a predictive accuracy worse than the best methods in Chapters 5 and 6.

As shown in Table 31, the predictive accuracy determined by C-index values achieved from the two different random-forest-based methods suggests a very little difference between the performance of the two approaches, 6 wins for DSF-KNN-mean and 5 wins for RSF-constant-hazard-mean. In addition, the average rank of DSF-KNN-mean (1.45) is slightly lower (better) than that of the RSF-constant-hazard-mean (1.55). By applying the Wilcoxon signed-ranks test, the statistical result reports that the null hypothesis of equal rank for both methods cannot be accepted, given the large p-value of 0.97. Hence, there is no significant difference

Table 32: Most Frequently Selected Hyper-parameter Settings, for each DSF variant and each dataset

Dataset (Disease)	DSF	DSF (constant-hazard-mean)	DSF (KNN-mean)
Alzheimer	(1, 2, 3)	(1, 1, 3)	(1, 2, 3)
Angina	(1, 1, 1)	(1, 1, 3)	(1, 1, 3)
HeartAtt	(1, 4, 3)	(1, 4, 3)	(1, 1, 3)
Psychiatric	(1, 1, 3)	(1, 1, 3)	(1, 2, 3)
Stroke	(1, 2, 3)	(1, 2, 3)	(1, 4, 3)
Diabetes	(1, 2, 3)	(1, 2, 3)	(1, 2, 3)
Cancer	(2, 1, 3)	(1, 1, 3)	(1, 1, 3)
Arthritis	(1, 4, 3)	(1, 4, 1)	(1, 4, 1)
Any-disease (ELSA)	(2, 4, 3)	(1, 1, 1)	(1, 2, 3)
Any-disease (SHARE)	(1, 4, 3)	(1, 4, 3)	(1, 4, 3)
HD	(1, 1, 1)	(1, 1, 1)	(1, 1, 1)

between these two best proposed methods.

6.4.5 Most Frequently Selected Hyper-parameter Values

Table 32 reports the hyper-parameter settings most frequently selected by the nested cross-validation procedure (which was used for hyper-parameter optimisation) for the original DSF (Utkin et al., 2020) and the two proposed DSF variants with different modified leaf-node-prediction criteria. Recall that the nested cross-validation procedure selected one configuration of hyper-parameter settings for each DSF variant for each of the training sets of the external cross-validation, for each dataset. Hence, among those selected hyper-parameter settings per dataset, the most frequently selected one is reported in the table. Thus, in each cell of the table, the three values in brackets are the most frequently selected settings for each of the 3 hyper-parameters being optimised.

As mentioned earlier, the 3 hyper-parameter optimised for these DSF variants are: (a) n_fail_layers, i.e., the maximum number of rounds allowed for the training process to terminate when the performance on the validation set fails to

improve compared against the best performance achieved so far on the validation set; (b) n_rfs_layer, i.e., the number of forests in each layer; and (c) *d0*, i.e., the minimum number of uncensored instances required at each leaf node of the trees in a forest. Recall also that n_fail_layers can take the values 1 or 2; n_rfs_layer can take the values 1, 2 or 4; and *d0* can take the values 1 or 3.

As reported in Table 32, the top 3 frequently selected hyper-parameter setting combinations (n_fail_layers, n_rfs_layer, $d0$) are (1,2,3), (1,1,3) and (1,4,3), which were the winners 9 times, 8 times and 7 times, respectively, each out of 33 cases (considering the 3 DSF methods and the 11 datasets). Hence, the combination of $n\_fail\_layers = 1$ and $d0 = 3$ seems to pair well together, while the n_rfs_layer hyper-parameter tends to take different values depending on the dataset and the DSF variant.

Furthermore, considering the selection frequency of each hyper-parameter setting separately across all DSF variants and all datasets, the n_fail_layers value of 1 is particularly effective: it was selected 31 out of 33 times. This value means that no more layers should be added once the accuracy has dropped from that of the last round. An advantage over the n_fail_layers value of 2 is the efficiency of the algorithm as this will remarkably reduce the computational time during the training process. Additionally, these results show that, in these datasets, shallower models (where the algorithm tends to end up with fewer layers in the DSF models) are more effective in general.

Next, regarding the n_rfs_layer parameter, the value of 1 seems to be slightly preferred overall; it was selected 14 out of 33 times. Interestingly, this means that one forest in each layer is sufficient to create a single additional feature to be fed-forward to the next layer. Hence, it seems that, overall, it is slightly better to use a single forest with 500 trees for these datasets; although the optimal value of this hyper-parameter is clearly dependent on the dataset and DSF variant.

Layer Count

Recall that the DSF algorithm dynamically adjusts the number of layers based on the characteristics and complexity of training data, in order to optimise predictive performance. During the experiments conducted on the nine ELSA datasets and the haemodialysis dataset using 10-fold cross-validation, and the SHARE dataset where 5-fold cross-validation was employed, the three variants of the DSF algorithm, namely DSF, DSF-constant-hazard, and DSF-kNN-mean, generated varying numbers of layers. The number of layers in each variant is reported in Table 33.

Table 33: Frequency of selection for different numbers of layers in the DSF models

Variant	1 Layer	2 Layers	3 Layers	4 Layers	5 Layers	6 Layers
DSF	47	43	5	8	1	1
DSF-const	71	29	3	2	-	-
DSFkNN	80	16	3	5	-	1

It can be observed on Table 33 that the majority of models in all three DSF variants tend to have either 1 or 2 layers. The frequency of models with more layers decreases as the number of layers increases, and very few models had more than 4 layers.

Overall, the insights gained from these experiments highlight the dynamic nature of the DSF algorithm and its variants, showcasing their ability to adjust the number of layers based on the specific dataset being processed. This adaptability reflects the algorithms' potential to optimize performance and capture complex relationships within the data.

6.4.6 Computational Runtimes

Compared to the RSF variants, all three DSF variants exhibit significantly larger runtimes. This is primarily due to the need to train multiple random forests per layer and the presence of multiple layers in the DSF model. Consequently, the

runtime of the DSF variants is influenced by the dataset size and the number of layers in the model.

As previously mentioned, the datasets derived from the ELSA database are not very large, resulting in relatively fast DSF runtimes, ranging from about 7 to 10 days to run a nested cross-validation for each DSF variant, for each dataset. In contrast, each experiment on the DSF variants consumed a very large amount of time when analysing the SHARE dataset, consisting of over 100,000 instances, leading to experiment durations of approximately 11 weeks for the nested cross-validation of each DSF variant. On the other hand, the haemodialysis dataset yielded the fastest experiment results, with a nested cross-validation for each DSF variant completing within 5 days. Please note, again that, the experiments were conducted using the Myrtle computer cluster hosted by the University of Kent. The cluster had with 80 CPU threads, which were distributed among the department's PhD students and researchers.

6.4.7 Top-ranked features for survival prediction

Neither the RSF nor the DSF algorithm produces a model which is directly interpretable since their learned models have too many survival trees. However, in order to provide human users with some interpretation of the learned models, we can use a feature importance measure to identify the most important features in a RSF/DSF model learned from the data, i.e., the features that most influence the predictions of the model. This feature-importance analysis can highlight general trends in the learned models and can be useful to better understand the relationships between some features and the target variable – i.e., the time passed until the diagnosis of age-related disease.

Hence, we report next the most important features in the survival models learned by the winning algorithm for each dataset, i.e., either RSF-constant-hazard-mean or DSF-KNN-mean, whichever was the algorithm with the better

predictive performance for each dataset, according to the results in Table 31. To identify the most important features for each dataset, we first computed the importance of each feature in the learned RSF/DSF model, using the well-known "permutation feature importance" measure (Altmann et al., 2010). This measure essentially quantifies the decrease in the C-index of the learned RSF/DSF model when a single feature has its values randomly shuffled.

More precisely, there are five steps involved in the permutation feature importance procedure. First, after a survival model has been trained, its C-index is calculated on the test set to measure the model's performance, as a baseline. Second, the procedure iteratively shuffles the values of a feature in the test set, one feature at a time, while keeping the values of other features constant. Third, the C-index value is re-calculated for each feature, in order to measure that feature's importance score. I.e., for each feature, the procedure calculates the difference between the baseline C-index value and the new C-index value obtained when that feature's values are shuffled in the test set: the larger the decrease in C-index value, the higher the importance of the feature. Forth, the shuffling of each feature can be repeated for multiple iterations, and the procedure computes the average importance score over those iterations, in order to get more robust results (instead of relying on a single random shuffling). In this work, the number of random shuffle iterations is 100. Fifth, the features are ranked by their importance scores. The permutation importance measure was computed by ELI5, a Python package built on top of scikit-learn (`https://github.com/eli5-org/eli5/blob/master/docs/source/overview.rst`).

We report three sets of most important features, one for the ELSA datasets and the other two for the SHARE dataset and the haemodialysis dataset, as follows. First, recall that the 9 ELSA datasets share the same set of 44 predictive features – those datasets differ in their target variables (age-related diseases). Hence, we identify the top-ranked features across the RSF/DSF models learned for those 9

Table 34: The 8 features which appear most often in the sets of top-10 features in the RSF/DSF models learned from the ELSA datasets

Feature	count	Alzheimer	Angina	HeartAtt	Psychiatric	Stroke	Diabetes	Cancer	Arthritis	Any-disease (ELSA)
mmgsn_me (grip strength)	8	V	V	V	V	V		V	V	V
confage (age)	6	V	V	V	V	V		V		
hdl (blood test)	6	V		V	V	V	V		V	
mmrroc (chair-rise test)	5	V	V	V		V		V		
wtval (weight)	5	V					V	V	V	V
scako (alcohol use frequency)	4		V		V	V	V			
smokerstat (smoking status)	4		V	V	V		V			
indsex (gender)	4			V	V				V	V

Table 35: The 4 most important features in the RSF model learned from the SHARE dataset

Rank	Feature	Description	Pred. Error
1	age	Age at interview (in years)	0.0878 ± 0.0286
2	mobilityind	Mobility index (high: has difficulties)	0.0284 ± 0.0203
3	bmi	Body mass index	0.0202 ± 0.0104
4	lgmuscle	Large muscle index (high: has difficulties)	0.0176 ± 0.0140
5	casp	CASP: quality of life and well-being index	0.0021 ± 0.0035

Table 36: The 4 most important features in the RSF model learned from the haemodialysis dataset

Rank	Feature	Description	Pred. Error
1	AgeFD	Age at first dialysis	0.0537 ± 0.0714
2	CharlsonScore	a measure of co-morbidity (incidence of multiple diseases)	0.0124 ± 0.0116
3	Myelomatosis_light _chain_deposit_disease	Myelomatosis/light chain deposit disease (binary)	0.0052 ± 0.0104
4	CHOL	Cholesterol level (HDL and LDL)	0.0025 ± 0.0076
5	UREADIFF	difference between urea after the dialysis	0.0020 ± 0.0070

datasets as a whole, which allows us to identify the most predictive features for multiple age-related diseases at the same time. This is useful to study the ageing process as a whole, from a more systemic perspective, rather than studying just one disease at a time. Second, in the case of the SHARE and haemodialysis datasets, since they contain uniquely distinct feature sets, we report their top-ranked features separately.

Table 34 shows the 8 most influential features overall, in the best RSF/DSF models learned from the 9 ELSA datasets. To identify these features, we first ranked the features in decreasing order of the permutation feature importance in each learned RSF/DSF model (i.e. for each dataset). Then, we computed the frequency of occurrence of each feature in the sets of top-10 features for those 9 datasets, and ranked the features in decreasing order of that frequency. That frequency is shown in the column "count" in Table 34, and the following columns show precisely for which datasets (i.e., diseases) the feature was among the top-10 features in the learned RSF/DSF model. This table includes all features whose count value is greater than or equal to 4 (out of the 9 ELSA datasets), and the 8 features satisfying this condition are shown in the first column of the table. In the columns with dataset names, for each cell, a tick symbol V indicates that the feature in the corresponding row is among the top-10 features for the dataset in the corresponding column.

The 8 most influential features shown in Table 34 (for the best RSF/DSF models learned from the ELSA datasets) can be described as follows (Clemens et al., 2019): *mmgsn_me* is the grip strength (Kg) of the non-dominant hand, $confage$ is the subject's age when the data were collected, *hdl* is the blood HDL (High-Density Lipoprotein) level, *mmrroc* is the outcome of chair-rise tests, *wtval* is the subject's valid weight (Kg), *scako* measures how often the subject had an alcoholic drink during the last 12 months, *smokerstat* is the present or past smoker status, and *indsex* is the gender.

In addition, Tables 35 and 36 report the 5 top-ranked features in the best RSF model learned from the SHARE and haemodialysis datasets, respectively (note that for both these datasets, the best RSF model outperformed the best DSF model). We report only the 5 top-ranked features in the best models for these two datasets due to their relatively small number of features, 15 and 38 features for the SHARE and Haemodialysis datasets, respectively.

As expected, the age variable takes the top spot (rank 1) in the best RSF model for both the SHARE and the Haemodialysis datasets, and is the runner-up (rank 2) in the best RSF/DSF models for the 9 ELSA datasets as a whole. Interestingly, several of the top-ranked features for these datasets are not standard biomarkers of specific diseases, but rather reflect the level of the frailty of individuals, like *mmgsn_me* and *mmrroc* for ELSA datasets and *mobilityind* for the SHARE dataset. Out of the several blood test results used as features in the ELSA datasets, only the HDL ("good cholesterol") level is among the top-8 features in Table 34; and the related variable CHOL (Cholesterol level) is also among the top-ranked features in Table 36 (for the Haemodialysis dataset).

6.5 Conclusion

This Chapter proposed new variants of the Deep Survival Forest (DSF) algorithm, a modification of the Deep Forest (DF) algorithm, which itself extends the Random Forest (RF) algorithm for classification or regression tasks. The DF algorithm draws from deep learning principles, especially the use of multiple learning layers. In a standard DF model, a Random Forest is trained at each layer, with predictions passed on to the next layer as additional features, much like a deep neural network. DSF extends this concept to survival analysis by replacing the base RF algorithm with the Random Survival Forest (RSF) algorithm, learning a stack of RSF models at each layer.

To assess these proposed DSF variants, we conducted a comparative analysis against the standard DSF and the standard Cox Proportional Hazards (PH) regression. Four types of experiments, using 11 real-world biomedical datasets, assessed the predictive accuracy (C-index values) of these methods.

In the first experiment, the standard DSF was compared with the standard RSF. The results demonstrated that the standard DSF consistently achieved significantly higher accuracies, validating it as a state-of-the-art survival analysis method since it outperformed the RSF, a well-recognized and effective survival analysis method.

The second experiment contrasted the two proposed DSF variants against the standard DSF. The proposed DSF with KNN-mean exhibited the highest predictive accuracy across all datasets. However, no statistically significant difference was observed among the three DSF variants.

The third experiment compared the best performing proposed DSF variant, with KNN-mean, against the widely used Cox PH regression. DSF with KNN-mean outperformed Cox PH regression in 9 out of 11 datasets, a statistically significant difference.

In the final experiment, we compared the best performing DSF variant from Chapter 6 (DSF with KNN-mean) with the best performing RSF variant from Chapter 5 (RSF with constant-hazard-mean). Both these variants outperformed the standard RSF. DSF with KNN-mean achieved higher predictive accuracy in 6 out of 11 datasets, while RSF with constant-hazard-mean performed better in the remaining 5 datasets. However, the difference was not statistically significant.

It should be noted that DSF with KNN-mean, and other DSF variants, pose a significant disadvantage due to their high computational cost. The layered nature of DSF models, each requiring the training of several RSF models, makes them the most computationally intensive methods evaluated in this thesis.

Chapter 7

Conclusions and Future Research

Survival analysis is a difficult research topic yet a promising one for employing powerful machine learning algorithms. Although the main characteristics of survival-analysis tasks look alike that of regression tasks (where machine learning methods have been very successful), survival analysis presents an interesting difficulty, i.e., data censorship. Unlike missing values, data censoring enables a wide range of data analysis techniques, taking into account the partial information about survival times associated with censored data. Therefore, broadly speaking, the goal is to utilise incomplete yet helpful information in censored data to build effective machine learning models for survival analysis.

In terms of machine learning algorithms, this thesis has focused on decision-tree-based ensembles, proposing new variants of random forests and random survival forests for survival analysis problems with censored data. This also included new variants of deep survival forests, which are based on some principles of deep learning and random survival forests.

The proposed algorithm variants have been evaluated on 11 real-world biomedical datasets, consisting mainly of datasets created from human ageing studies, where the target variable (to be predicted) is the time passed until the diagnosis of some age-related disease. The main motivation for focusing on this type of

dataset is due to the growing need for research on age-related diseases, given that old age has been shown to be the greatest risk factor for a wide variety of diseases and that the proportion of old people among the world population is continuing to increase.

In addition, most of the datasets used in the experiments have a relatively large proportion of censored data (i.e., a small proportion of uncensored instances), because a small proportion of participants were diagnosed with the age-related disease of interest by the time the data were collected. This is especially true for the datasets derived from the ELSA survey; where overall, the percentage of censored instances (subjects) is above 80% and up to 99%. This makes them particularly challenging datasets for survival analysis methods.

In summary, this research has developed new variants of supervised machine learning algorithms (based on ensembles of decision trees) for survival-analysis tasks. In addition, this thesis reported the results of computational experiments which compared the predictive accuracy of the proposed algorithm variants against the accuracy of several standard survival-analysis methods; overall the proposed algorithm variants have been shown to be competitive to (sometimes significantly more accurate than) standard methods.

The remainder of this chapter is divided into two sections. First, the next section summarises the thesis' contributions and the obtained computational results about the evaluation of the proposed algorithm variants. Then, the following section suggests research directions for future work.

7.1 Summary of Contributions

This section summarises the main contributions of this thesis, focusing on the dataset creation and the three types of new random forest variants proposed for survival analysis with censored data. These will be discussed in three separate

sub-sections, each reviewing the main rationale for the proposed method and summarising its computational results.

7.1.1 Dataset Creation

This subsection summarises this thesis' first contribution, which was the creation of the 11 survival analysis datasets used in the experiments reported in the thesis. In ten of these datasets, derived from the ELSLA and SHARE studies, the target variable represents the time until an individual is diagnosed with an age-related disease. However, in the haemodialysis dataset, the target variable represents the time until the death of an individual.

The dataset creation process involved data cleansing and the specification of a procedure for creating the target variables from the variables available in the original databases, as described in Chapter 3.

7.1.2 New Variants of Random Forests for Survival Analysis

This subsection summarises the description and computational results of the Random Target-Imputation Forests (RTIF) and the K-Nearest-Neighbours-Imputation Random Forests (KNN-RF) algorithms, which were described in Chapter 4.

In this context, we proposed two new variants of the random forest (RF) algorithm based on the imputation of censored target variables. These two proposed RF variants consist of modifying the procedure for creating subsets of training data to be used for learning the decision trees in a RF – which is basically the first step in the execution of a RF algorithm. This involved replacing the censored value of a target variable by another target value (an estimated survival time) which is then treated as an uncensored target value, so that all other components of the RF algorithm can be used without modification. Hence, these proposed

RF variants have the advantage of simplicity, involving just a small modification of standard RF algorithms for regression, to allow such algorithms to cope with survival-analysis tasks. The thesis proposes two variants of RF algorithms for this data transformation, as follows.

The first variant is based on replacing a censored target value by a randomly generated target value within instance-specific lower and upper bounds.

The second proposed RF variant is based on using the well-known K-Nearest Neighbour (KNN) algorithm to estimate the uncensored target value of each censored instance. This is a more sophisticated approach, it replaces the random generation of target variable values with a deterministic imputation method based on the actual target values from the uncensored subjects which are the nearest neighbours of the current censored subject (whose target variable value needs to be imputed). This approach uses the available known data as a "heuristic" for estimating survival times for the purpose of training the decision trees in a RF model.

The predictive performances of the two proposed RF variants were compared against the predictive performances of four other methods: a standard RF for regression, a RF with Inverse Probability of Censoring (IPC), the popular Cox regression with the Proportional Hazard (PH) assumption, and a standard Random Survival Forest (RSF) method, which is probably the most popular and successful survival analysis method in the machine learning literature.

The predictive performance of all these methods was evaluated by computing their C-index, which is probably the most popular performance measure in the survival analysis literature, in experiments with 11 real-world biomedical datasets. The results were analysed with statistical tests of significance. Four types of experiments were performed, all using the same 11 datasets, but with different experiments comparing the predictive performances of different sets of methods. The results of these experiments and statistical analyses can be summarised as

follows.

In the first experiment, RTIF obtained overall higher predictive accuracy than the standard RF regressor and RF with IPC weights. The results were statistically significant when comparing RTIF against each of the other two methods.

In the second experiment, KNN-RF also obtained overall higher predictive accuracy than the standard RF regressor and RF with IPC weights. The results were statistically significant when comparing KNN-RF against the RF regressor, but there was no significant difference between the results of KNN-RF and RF with IPC weights.

In the third experiment, the standard Random Survival Forest (RSF) obtained overall higher predictive accuracy than the two proposed RF variants (RTIF and KNN-RF). The results were statistically significant, i.e. RSF significantly outperformed both RTIF and KNN-RF.

Finally, in the fourth experiment, both RTIF and KNN-RF obtained overall a higher predictive accuracy than the standard and very popular Cox Proportional Hazards (PH) regression, with RTIF obtained overall the best ranking regarding predictive accuracy. However, the differences in predictive accuracies between RITF and each of the other two methods were not statistically significant.

7.1.3 New Variants of Random Survival Forests

This subsection summarises the description and results of several variants of Random Survival Forests (RSFs) proposed in Chapter 5.

Recall that the RSF algorithm learns an ensemble of survival trees, which are decision trees adapted to survival analysis problems. Hence, unlike the classical RF algorithm, the RSF algorithm employs some statistical techniques which enable it to cope with censored data. RSF is the most popular type of RF algorithm for survival analysis in the area of machine learning, and it has been shown to outperform several methods in survival analysis (Li et al., 2022; Zhang et al., 2022;

Snider and McBean, 2022; Miao et al., 2018a; Weeraddana et al., 2020; Gul et al., 2020).

The second contribution of this thesis was to propose a number of variants of the RSF algorithm. The proposed RSF variants focus on modifying two major components of the standard RSF algorithm: (a) the procedure used for selecting the feature to be used for splitting the data at each tree node (the node-splitting criterion); and (b) the procedure used for computing the value predicted by each leaf node (the leaf-node-prediction criterion).

Regarding the modification of the node-splitting criterion of RSFs, this thesis proposed to replace the standard Log-rank test by the Wilcoxon and Tarone-Ware tests. The main idea is that the original Log-rank node-splitting criterion assigns the same importance to all failure times, whilst the Wilcoxon and Tarone-Ware node-splitting criteria emphasize earlier failure times.

Regarding the modification of the leaf-node-prediction criterion of RSFs, this thesis proposed to modify the standard criterion of the Cumulative Hazard Function by a more direct and simpler estimate of the survival time for each subject, directly based on estimating the mean of the target variable over all instances in a leaf node, in a way that takes into account the presence of censored data. More specifically, four variants of the leaf-node-prediction criterion of RSFs were proposed, as follows:

1. **The Naive-mean criterion**: The survival time predicted at each leaf node is simply the mean over the target variable values of all instances in that leaf node, regardless of the instances' censorship status (censored or uncensored). This is a naive method used as a baseline only, which underestimates the true survival time at each node.
2. **The constant-hazard-mean criterion**: Making the strong assumption of constant hazard rate in the data, the mean survival time predicted at a leaf node is calculated as the summation of the target variable values of all

instances in that node, censored and uncensored included, divided by the number of uncensored instances. This procedure overestimates the survival time at each node.

3. **The weight-age-mean criterion (age-based weights)**: This is an extension of the previous criterion where, when computing the mean survival time predicted at a leaf node, censored subjects have their target variable values weighted, where the weights for older subjects are smaller than the weights for younger subjects, reflecting the fact that older subjects (at the time of censorship) are expected to have smaller survival times. As a result, the aforementioned overestimation of predicted survival time (associated with the constant-hazard-mean criterion) is reduced.
4. **K-NN mean**: In this criterion, before calculating the mean survival time at a leaf node, the unknown survival times of censored instances are estimated based on K-NN's predictions. Essentially, for each censored instance, the K-NN algorithm finds the K nearest uncensored neighbours of the current censored instance, and the value of the target variable for the current censored instance is replaced by the mean of the target variable values (survival times) among those K neighbours. Once this replacement is done for all originally censored instances in the current leaf node, the predicted survival time at that leaf node is simply the mean over all target variable values at that node.

The results of these proposed RSF variants were compared among themselves and against the standard RSF and the standard Cox Proportional Hazards (PH) regression, in experiments evaluating the predictive accuracy (C-index values) of the methods on 11 real-world biomedical datasets (described in Chapter 3). This involved four types of experiments, with different sets of methods being compared, and the results of these experiments can be summarised, as follows.

The first experiment compared the results of the three RSF variants with different node-splitting criteria, i.e. with the Log-rank (default criterion), Wilcoxon and Tarone-Ware criteria. In this experiment, all three RSF variants used the same standard leaf-node-prediction criterion for RSF, namely the Cumulative Hazard Function (CHF). Overall, the RSF variant with the default Log-rank test obtained the best predictive accuracy results (highest C-index values), but there was no statistically significant difference among the results of these three RSF variants.

The second experiment compared the results of the aforementioned five RSF variants with different leaf-node-prediction criteria. All these RSF variants used the same node-splitting criterion, the default Log-rank test, which obtained the best results in the first experiment. Overall, RSF with the "constant-hazard-mean" criterion obtained the highest predictive accuracy (C-index values) across the 11 datasets. When comparing the results of RSF constant-hazard-mean against the results of each of the other four RSF variants, RSF constant-hazard-mean was found to be significantly better than three of the other four variants – the only exception was that there was no significant difference between the RSF constant-hazard-mean and RSF weight-age-mean variants.

The third experiment compared three RSF variants with different node-splitting criteria, again Log-rank, Wilcoxon and Tarone-Ware criteria, but this time all RSF variants used "constant-hazard-mean" as the leaf-node-prediction criterion, which was the best criterion with the best results in the second experiment. Overall, the best results were again obtained by the RSF variant with Log-rank. Again, there was no statistically significant difference between the three RSF variants in this experiment. Hence, the results of this third experiment are similar to the results of the first experiment, both confirming that the default Log-rank criterion was the best out of the three evaluated node-splitting criteria.

Finally, the fourth experiment compared the best RSF variant across all previous experiments, namely RSF with Log-rank test and constant-hazard-mean predictions, against the very popular Cox Proportional Hazards (PH) regression. That RSF variant achieved higher predictive accuracy than the accuracy of Cox PH in 10 of the 11 datasets, and this result was statistically significant.

7.1.4 New Variants of Deep Survival Forests

This subsection summarises the description and results of the new variants of the Deep Survival Forest (DSF) algorithm proposed in Chapter 6.

The DSF algorithm is an extension of the Deep Forest (DF) algorithm, which is in turn an extension of the RF algorithm designed for regression or classification, rather than survival analysis. The DF algorithm is based on some principles of deep learning, particularly the use of multiple layers of learning. In the standard DF algorithm, a set of RFs is trained in each layer and the predictions of one layer are propagated to the next layer as additional features (analogous to the layers of a deep neural network). That is, a DF model is essentially a stack of several RF models, each of which is learned using the original RF algorithm. The DSF algorithm extends the DF algorithm by adapting it to survival analysis, by simply replacing the base RF algorithm with the RSF algorithm. Hence, the DSF algorithm learns a stack of several RSF models, each learned using the standard RSF algorithm.

The third contribution of this thesis was to propose two variants of the DSF algorithm. The proposed DSF algorithm modifications arrange RSF models into layers in the same manner as the standard DSF method does. The difference is that the two proposed DSF variants use two of the RSF variants proposed in Chapter 5, namely RSF constant-hazard-mean and RSF KNN-Mean (instead of standard RSF), as base learners in each layer.

The results of the proposed DSF variants were compared against the standard

DSF and the standard Cox Proportional Hazards (PH) regression, in experiments which evaluated the predictive accuracy (C-index values) of the methods on 11 real-world biomedical datasets (described in Chapter 3). This involved four types of experiments, with different sets of methods being compared across the experiments. The results of these experiments can be summarised as follows.

In the first experiment, the standard DSF was compared against the standard RSF. The results have shown that standard DSF obtained statistically significantly higher accuracies than standard RSF. This supports the claim that standard DSF is a state-of-the-art survival analysis method, since it clearly outperformed standard RSF, which is by itself a very strong survival analysis method, often outperforming other methods in the literature, as mentioned earlier.

In the second experiment, the two proposed variants of DSF (with leaf-node-prediction criteria proposed in this thesis) were compared against standard DSF. Overall, the highest predictive accuracy (C-index) results were obtained by the proposed DSF with KNN-mean. However, the differences in the results of the three DSF variants were not statistically significant.

In the third experiment, the best proposed DSF variant, with KNN-mean, was compared against the very popular Cox PH regression. The results have shown that DSF with KNN-mean obtained higher predictive accuracies in 9 of the 11 datasets, and the difference in predictive accuracy between these two methods was statistically significant.

In the fourth experiment, DSF with KNN-mean, which obtained the best results among the DSF variants evaluated in Chapter 6, was compared against RSF with constant-hazard-mean, which obtained the best results among the DSF variants evaluated in chapter 5. Note that both these algorithm variants obtained better results than the standard RSF, which in turn was the best method evaluated in Chapter 4. Hence, the results of this fourth experiment reported in Chapter 6 are useful to determine which was the best survival analysis method overall,

across all methods evaluated in the thesis. The results have shown that DSF with KNN-mean obtained higher predictive accuracy (C-index values) in 6 of the 11 datasets, whilst RSF with constant-hazard-mean obtained higher accuracy in the other 5 datasets. The difference in these results was not statistically significant.

It should be noted, however, that DSF with KNN-mean, as well as the other DSF variants (including standard DSF), have the disadvantage of being by far the most computationally expensive of the several types of methods evaluated in this thesis. This is due to a DSF model typically having multiple layers, where each layer requires the training of several RSF models.

7.2 Research Directions for Future Work

This section suggests five research directions for future work, as follows. First, proposing other new variants of the Random Survival Forest (RSF) algorithm. Second, proposing other new variants of the Deep Survival Forest (DSF) algorithm. Third, performing additional experiments to evaluate the proposed algorithm variants. Fourth, proposing a new Automated Machine Learning (Auto-ML) system for survival analysis. Finally, discovering new knowledge or patterns about age-related diseases. These research directions are discussed in the next five subsections.

7.2.1 Proposing other new variants of the Random Survival Forest (RSF) algorithm

Although several different variants of the RSF algorithm have already been proposed in the thesis, it would still be interesting to design and evaluate other, potentially more powerful variants of the algorithm – possibly using as a basis a more advanced version of this type of algorithm in the literature.

Taking the Oblique Random Survival Forest (ORSF) algorithm (Jaeger et al.,

2019) as an example, this algorithm learns an ensemble of survival trees with oblique data splits where features were regularised by Cox proportional hazard models before applying the Log-rank test. Therefore, two options for modifying this algorithm are as follows: (1) the Log-rank test could be replaced with other node-splitting-criteria, such as the other criteria used in this thesis or a different criterion like the AUC-based node-splitting criteria proposed in (Eifler, 2014); and (2) the leaf-node-prediction criterion of ORSF could be replaced with other criteria, such as the other criteria proposed in the thesis.

In addition, as a more sophisticated type of leaf-node-prediction criterion, it would be interesting to investigate the use of model trees (Witten et al., 2005), where the target value predicted by each leaf node is computed by a linear regression model, learned for the data belonging to that leaf node, after replacing censored target values by an estimate (like the one calculated by the K-NN-mean criterion). The use of such linear models in survival trees' leaf nodes could be investigated not only as a variant of the standard RSF algorithm, but also as a variant of the ORSF algorithm.

7.2.2 Proposing other new variants of the Deep Survival Forest (DSF) algorithm

Recall that the standard DSF algorithm learns a model consisting of multiple layers (inspired by the multiple layers of deep neural networks), where each layer consists of several RSF models, learned by the standard RSF algorithm. Hence, a simple but potentially effective approach for designing a more powerful DSF variant consists of replacing the base algorithm that is used for learning the RSF models in each layer. More specifically, the standard RSF algorithm used for learning in each layer could be replaced by another, potentially more effective RSF variant.

This thesis has already proposed (in Chapter 6) two DSF variants where the

base RSF algorithm was replaced by two corresponding RSF variants proposed in Chapter 5 of the thesis. It should be noted that, out of the several new RSF variants proposed in Chapter 5, only two were selected to be used as a basis for the two new DSF variants proposed in Chapter 6. This was mainly due to a limitation in the time available to run all experiments with DSF variants since the DSF algorithm (and its variants) are very time-consuming. Hence, in future work, it would be interesting to propose and evaluate other new DSF variants, based on other RSF variants proposed in Chapter 5; or going further, new DSF variants based on other potentially more powerful types of RSF variants not evaluated in this thesis, like the aforementioned Oblique RSF.

7.2.3 Performing additional experiments to evaluate the proposed algorithm variants

In this thesis, the proposed variants of RF, RSF and DSF algorithms were compared against each other and against two other types of methods: the standard versions of those algorithms, and other well-known methods for survival analysis (including the very popular Cox PH regression).

However, one limitation of the experiments is that the proposed RF, RSF, and DSF variants were not compared against other variants of those algorithms proposed in the literature – e.g. (Weeraddana et al., 2020; Wang and Li, 2017; Miao et al., 2018b; Eifler, 2014; Wright, Dankowski and Ziegler, 2017; Wang and Liu, 2018; Wongvibulsin, Wu and Zeger, 2019; Jaeger et al., 2019; Utkin et al., 2019b; Tollenaar and Van Der Heijden, 2019; Gul et al., 2020). Hence, future research could be directed at performing this kind of comparison, in order to further evaluate the effectiveness of the proposed algorithm variants.

In addition, future research could include extending the experiments to include more real-world survival analysis datasets, beyond the 11 real-world datasets used in this research.

7.2.4 Proposing a new Automated Machine Learning (Auto-ML) system for survival analysis

Automated Machine Learning (Auto-ML) is a sub-area of machine learning that involves automatically selecting the best machine learning algorithm (out of a pre-defined set of algorithms) and its best hyper-parameter settings for a given input dataset (Waring, Lindvall and Umeton, 2020; Liu, Lu and Lu, 2021). Therefore, Auto-ML avoids users spending a lot of time and effort with ad-hoc experiments trying user-specified machine learning algorithms and different configurations (hyper-parameter settings) of those algorithms.

The majority of Auto-ML systems address a standard supervised learning task (typically classification), so those systems' base algorithms cannot directly cope with censored data. Hence, an interesting direction for future research would be to develop an Auto-ML system for survival analysis tasks with censored data. In this case, the space of candidate algorithms could include e.g. many variants of RSF, DSF and Cox regression algorithms (as well as other types of survival analysis methods), and then the Auto-ML system would automatically select the best of those algorithm variants and its best configuration for any given input survival analysis dataset.

This would be a more systematic and principled approach for exploring a large number of different survival analysis methods as a whole, in a single experiment; as opposed to the more ad-hoc approach followed in this thesis, where many smaller experiments (each comparing a few methods) were performed, and their results had to be individually analysed.

This author is aware of only two works on Auto-ML systems that address survival analysis tasks, coping with censored data, as follows. The AutoScore-Survival system was specifically designed to cope with time-to-event, right-censored data (Xie et al., 2022). However, from an Auto-ML perspective, it is a relatively

simple system where the used survival analysis methods are essentially a standard RSF and standard Cox regression. Therefore, there is plenty of opportunity for extending the set of candidate survival analysis methods in the AutoScore-Survival system with other variants of RSF and Cox regression, as well as other more advanced methods like DSF. Actually, the authors of (Xie et al., 2022) admit (in the Discussion section of their article) that "this is the initial development of AutoScore-Survival ... Future development should extend the framework with advanced algorithms".

Another relevant Auto-ML system is Just Add Data Bio (JADBio), which was designed for biomarker discovery in biomedical applications, i.e. identifying a relatively small set of features that can be used as biomarker or biosignatures for predicting a target variable (Tsamardinos et al., 2022). This article claims that the system can cope with right-censored data, but the article focused on reporting the results of the system in several biomedical case studies, rather than a precise machine learning-oriented description of the system. Hence, it is not clear in the article which candidate survival analysis methods can be selected by the system and which of their hyper-parameter settings can be optimised.

7.2.5 Discovering new knowledge or patterns about age-related diseases

Finally, the survival analysis datasets of age-related diseases created in this thesis represent an interesting and important application domain; considering that, as discussed earlier, the proportion of elderly people in the world population is increasing (WHO, 2022). Hopefully, the application of machine learning methods to such datasets can lead to new knowledge, patterns or insights about which predictive variables have a greater influence in the development of age-related diseases (as the type of target variable being predicted).

This thesis has made an initial attempt at identifying such most important

predictive variables in each of the created datasets, by ranking the features in the best learned RSF or DSF models in decreasing order of feature importance and reporting the top-ranked features in each dataset. However, this initial analysis of feature importance was limited to using just one standard feature importance measure, which is often used in the context of random forests.

Hence, a natural direction for future research would be to perform a more extensive feature importance analysis by using several different feature importance measures proposed in the literature, and comparing their results. For instance, it would be interesting to evaluate to what extent different feature importance measures produce similar feature ranks, and try to identify a subset of features which are consistently among the top-ranked features across all or nearly all of the feature importance measures used in an experiment. This could lead to the identification of a more robust set of features that most strongly predict the future diagnosis of some age-related diseases.

More broadly speaking, this could potentially improve our understanding of how complex health states affect the process of ageing and perhaps open up new treatment paths for extending a healthy lifespan (i.e. the period of life spent in good health, without suffering the heavy burden of age-related diseases). After all, we want to maintain our physical and mental capability to perform the activities that we cherish at old age, rather than living idly in the hospital.

Bibliography

(2006). Machine learning in bioinformatics: A brief survey and recommendations for practitioners. *Computers in Biology and Medicine*, 36, pp. 1104–1125.

Aalen, O. (1978). Nonparametric inference for a family of counting processes. *The Annals of Statistics*, pp. 701–726.

Akai, H. et al. (2018a). Predicting prognosis of resected hepatocellular carcinoma by radiomics analysis with random survival forest. *Diagnostic and interventional imaging*, 99(10), pp. 643–651.

Akai, H. et al. (2018b). Predicting prognosis of resected hepatocellular carcinoma by radiomics analysis with random survival forest. *Diagnostic and Interventional Imaging*, 99, pp. 643–651.

Altmann, A., Toloşi, L., Sander, O. and Lengauer, T. (2010). Permutation importance: a corrected feature importance measure. *Bioinformatics*, 26(10), pp. 1340–1347.

Blanco, R., Inza, I., Merino, M., Quiroga, J. and Larrañaga, P. (2005). Feature selection in Bayesian classifiers for the prognosis of survival of cirrhotic patients treated with TIPS. *Journal of Biomedical Informatics*, 38(5), pp. 376–388.

Börsch-Supan, A. et al. (2013). Data resource profile: The survey of health, ageing and retirement in europe (share). *International Journal of Epidemiology*, 42(4), pp. 992–1001.

Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2), pp. 123–140.

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), pp. 5–32, `/dx.doi.org/10.1023{%}2FA{%}3A1010933404324`.

Bruning, J. L. and Kintz, B. L. (1987). *Computational handbook of statistics*. Scott, Foresman & Co.

Chang, S. W., Abdul-Kareem, S., Merican, A. F. and Zain, R. B. (2013). Oral cancer prognosis based on clinicopathologic and genomic markers using a hybrid of feature selection and machine learning methods. *BMC Bioinformatics*, 14(1), p. 170.

Chen, K. et al. (2020). Comparative analysis of surface water quality prediction performance and identification of key water parameters using different machine learning models based on big data. *Water research*, 171, p. 115454.

Chen, Y. C., Ke, W. C. and Chiu, H. W. (2014). Risk classification of cancer survival using ANN with gene expression data from multiple laboratories. *Computers in Biology and Medicine*, 48(1), pp. 1–7.

Clark, T. G., Bradburn, M. J., Love, S. B. and Altman, D. G. (2003). Survival Analysis Part I: Basic concepts and first analyses. *British Journal of Cancer*, 89(2), pp. 232–238.

Clemens, S. et al. (2019). English Longitudinal Study of Ageing: Waves 0-8 https://www.elsa-project.ac.uk/.

Cox, D. R. (1972). Regression models and life-tables. *Journal of the Royal Statistical Society: Series B (Methodological)*, 34(2), pp. 187–202.

Delen, D., Walker, G. and Kadam, A. (2005). Predicting breast cancer survivability: A comparison of three data mining methods. *Artificial Intelligence in Medicine*, 34(2), pp. 113–127.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7, pp. 1–30.

Desmedt, C. et al. (2007). Strong time dependence of the 76-gene prognostic signature for node-negative breast cancer patients in the transbig multicenter independent validation series. *Clinical cancer research*, 13(11), pp. 3207–3214.

Dietrich, S. et al. (2016). Random survival forest in practice: a method for modelling complex metabolomics data in time to event analysis. *International journal of epidemiology*, 45(5), pp. 1406–1420.

Eifler, F. (2014). *Introduction of AUC-based splitting criteria to random survival forests*. Ph.D. thesis.

Fabris, F., de Magalhães, J. P. and Freitas, A. A. (2017). A review of supervised machine learning applied to ageing research. *Biogerontology*, 18(2), pp. 171–188.

Faraggi, D. and Simon, R. (1995). A neural network model for survival data. *Statistics in medicine*, 14(1), pp. 73–82.

Fernández-Delgado, M., Cernadas, E., Barro, S. and Amorim, D. (2014). Do we need hundreds of classifiers to solve real world classification problems? *The journal of machine learning research*, 15(1), pp. 3133–3181.

Friedman, J. H. (1997). On bias, variance, 0/1—loss, and the curse-of-dimensionality. *Data mining and knowledge discovery*, 1(1), pp. 55–77.

Friedman, M. (1940). A comparison of alternative tests of significance for the problem of m rankings. *Annals of Mathematical Statistics*, 11, pp. 86–92.

Ganaie, M. A., Hu, M. et al. (2021). Ensemble deep learning: A review. *arXiv preprint arXiv:210402395*.

George, J., Elliott, R. A. and Stewart, D. C. (2008). A systematic review of interventions to improve medication taking in elderly patients prescribed multiple medications. *Drugs & aging*, 25(4), pp. 307–324.

Geurts, P., Ernst, D. and Wehenkel, L. (2006). Extremely randomized trees. *Machine learning*, 63(1), pp. 3–42.

Ghorbani, A., Abid, A. and Zou, J. (2019). Interpretation of neural networks is fragile. In *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, pp. 3681–3688.

Goodfellow, I., Bengio, Y. and Courville, A. (2016). *Deep learning*. MIT press.

Gruber, S., Hunkler, C. and Stuck, S. (2014). Generating easyshare: guidelines, structure, content and programming. Tech. rep., SHARE Working Paper Series 17-2014. Munich.

Gul, N. et al. (2020). Optimal survival trees ensemble. *arXiv preprint arXiv:200509043*.

Hamidi, O., Poorolajal, J., Farhadian, M. and Tapak, L. (2016). Identifying important risk factors for survival in kidney graft failure patients using random survival forests. *Iranian journal of public health*, 45(1), p. 27.

Harrell, F. E., Lee, K. L. and Mark, D. B. (1996). Tutorial in Biostatistics Multivariable Prognostic Models: Issues in Developing Models, Evaluating Assumptions and Adequacy,and Measuring and Reducing Errors. *Statistics in Medicine*, 15(4), pp. 361–387.

Holm, S. (1979). A simple sequentially rejective multiple test procedure. *Scandinavian journal of statistics*, pp. 65–70.

Hothorn, T., Buhlmann, P., Dudoit, S., Molinaro, A. and Van Der Laan, M. J. (2006). Survival ensembles. *Biostatistics*, 7(3), pp. 355–373.

Hu, X., Chu, L., Pei, J., Liu, W. and Bian, J. (2021). Model complexity of deep learning: A survey. *Knowledge and Information Systems*, 63(10), pp. 2585–2619.

Ibrahim, J. G., Chen, M.-H., Sinha, D., Ibrahim, J. and Chen, M. (2001). *Bayesian survival analysis*, vol. 2. Springer.

Iman, R. L. and Davenport, J. M. (1980). Approximations of the critical region of the friedman statistic. *Communications in Statistics - Theory and Methods*, A9, pp. 571–595.

Ishwaran, H. and Kogalur, U. B. (2007). Random Survival Forests for R. *New Functions for Multivariate Analysis*, 7(2), pp. 25–31.

Ishwaran, H., Kogalur, U. B., Blackstone, E. H. and Lauer, M. S. (2008). Random survival forests. *Annals of Applied Statistics*, 2(3), pp. 841–860.

Jaeger, B. C. et al. (2019). Oblique random survival forests. *Annals of Applied Statistics*, 13(3), pp. 1847–1883.

Japkowicz, N. and Shah, M. (2011). *Evaluating learning algorithms: a classification perspective*. Cambridge University Press.

Kalbfleisch, J. D. and Prentice, R. L. (2011). *The statistical analysis of failure time data*. John Wiley & Sons.

Kaplan, E. L. and Meier, P. (1958). Nonparametric Estimation from Incomplete Observations. *Journal of the American Statistical Association*, 53(282), pp. 457–481.

Kleinbaum, D. G. and Klein, M. (2012). *Survival Analysis: A Self-Learning Text, Third Edition*, 700 pages, Springer.

Kourou, K., Exarchos, T. P., Exarchos, K. P., Karamouzis, M. V. and Fotiadis, D. I. (2015). Machine learning applications in cancer prognosis and prediction. *Computational and Structural Biotechnology Journal*, 13, pp. 8–17.

Leary, E. B. et al. (2020). Association of rapid eye movement sleep with mortality in middle-aged and older adults. *JAMA neurology*, 77(10), pp. 1241–1251.

LeCun, Y., Bengio, Y. and Hinton, G. (2015). Deep learning. *nature*, 521(7553), pp. 436–444.

Li, H. et al. (2022). Development and validation of a new multiparametric random survival forest predictive model for breast cancer recurrence with a potential benefit to individual outcomes. *Cancer Management and Research*, 14, p. 909.

Lin, Y. and Jeon, Y. (2006). Random forests and adaptive nearest neighbors. *Journal of the American Statistical Association*, 101(474), pp. 578–590.

Liu, G., Lu, D. and Lu, J. (2021). Pharm-automl: An open-source, end-to-end automated machine learning package for clinical outcome prediction. *CPT: pharmacometrics & systems pharmacology*, 10(5), pp. 478–488.

Lynch, C. M. et al. (2017). Prediction of lung cancer patient survival via supervised machine learning classification techniques. *International Journal of Medical Informatics*, 108(April 2016), pp. 1–8.

Miao, F., Cai, Y.-P., Zhang, Y.-X., Fan, X.-M. and Li, Y. (2018a). Predictive modeling of hospital mortality for patients with heart failure by using an improved random survival forest. *IEEE Access*, 6, pp. 7244–7253.

Miao, F., Cai, Y. P., Zhang, Y. X., Fan, X. M. and Li, Y. (2018b). Predictive modeling of hospital mortality for patients with heart failure by using an improved random survival forest. *IEEE Access*, 6, pp. 7244–7253.

Mogensen, U. B., Ishwaran, H. and Gerds, T. A. (2012). Evaluating random forests for survival analysis using prediction error curves. *Journal of Statistical Software*, 50, p. 1.

Molinaro, A. M., Dudoit, S. and Van Der Laan, M. J. (2004). Tree-based multivariate regression and density estimation with right-censored data. *Journal of Multivariate Analysis*, 90(1 SPEC. ISS.), pp. 154–177.

Nakatsu, G. et al. (2018). Alterations in enteric virome are associated with colorectal cancer and survival outcomes. *Gastroenterology*, 155(2), pp. 529–541.

Nelson, W. (1972). Theory and Applications of Hazard Plotting for Censored Failure Data. *Technometrics*, 14(4), pp. 945–966.

Panahiazar, M., Taslimitehrani, V., Pereira, N. and Pathak, J. (2015). Using EHRs and Machine Learning for Heart Failure Survival Analysis. *Studies in Health Technology and Informatics*, 216, pp. 40–44.

Pang, H., George, S. L., Hui, K. and Tong, T. (2012). Gene selection using iterative feature elimination random forests for survival outcomes. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 9(5), pp. 1422–1431.

Pedregosa, F., Varoquaux, G. et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, pp. 2825–2830.

Pölsterl, S. (2020). scikit-survival: A library for time-to-event analysis built on top of scikit-learn. *Journal of Machine Learning Research*, 21(212), pp. 1–6.

Pölsterl, S., Navab, N. and Katouzian, A. (2015). Fast training of support vector machines for survival analysis. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Springer, pp. 243–259.

Pölsterl, S., Navab, N. and Katouzian, A. (2015). Fast training of support vector machines for survival analysis. In *In: Lecture Notes in Computer Science*, vol. 9285, Springer, Cham, pp. 243–259, `arXiv:1207.6324`.

Pomsuwan, T. (2017). *Feature Selection for the Classification of Longitudinal Human Ageing Data*. Master by research thesis, Master by Research Thesis. University of Kent.

Probst, P., Wright, M. N. and Boulesteix, A. L. (2019). Hyperparameters and tuning strategies for random forest. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 9(3), pp. 1–15.

Quinlan, J. (1993). C4. 5: programs for machine learning. *San Mateo, CA: Morgan Kaufmann*.

Rahman, S. A. et al. (2021). The augis survival predictor: prediction of long-term and conditional survival after esophagectomy using random survival forests. *Annals of Surgery*.

Roadknight, C., Suryanarayanan, D., Aickelin, U., Scholefield, J. and Durrant, L. (2015). An ensemble of machine learning and anti-learning methods for predicting tumour patient survival rates. *Proceedings of the 2015 IEEE International Conference on Data Science and Advanced Analytics, DSAA 2015*, pp. 1–7.

Robins, J. M. and Rotnitzky, A. (1992). Recovery of information and adjustment for dependent censoring using surrogate markers. In *AIDS Epidemiology*, Springer, pp. 297–331.

Rokach, L. and Maimon, O. (2014). *Data Mining with Decision Trees: theory and applications*, *Series in Machine Perception and Artificial Intelligence*, vol. 81. World Scientific, 2nd edn.

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5), pp. 206–215.

Segal, M. R. (1988). Regression Trees for Censored Data. *Biometrics*, 44(1), p. 35.

Selvin, S. (2008). *Survival analysis for epidemiologic and medical research*. Cambridge University Press.

Shrestha, A. and Mahmood, A. (2019). Review of deep learning algorithms and architectures. *IEEE access*, 7, pp. 53040–53065.

Simon, N., Friedman, J., Hastie, T. and Tibshirani, R. (2011). Regularization paths for cox's proportional hazards model via coordinate descent. *Journal of statistical software*, 39(5), p. 1.

Singh, A., Thakur, N. and Sharma, A. (2016). A review of supervised machine learning algorithms. In *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, Ieee, pp. 1310–1315.

Snider, B. and McBean, E. A. (2022). Assessing the impact of pipe rehabilitation on decreasing watermain break rates using random survival forest models. *Journal of Water Resources Planning and Management*, 148(8), p. 04022045.

Spooner, A. et al. (2020). A comparison of machine learning methods for survival analysis of high-dimensional clinical data for dementia prediction. *Scientific reports*, 10(1), pp. 1–10.

Štajduhar, I., Dalbelo-Bašić, B. and Bogunović, N. (2009). Impact of censoring on learning Bayesian networks in survival modelling. *Artificial Intelligence in Medicine*, 47(3), pp. 199–217.

Tollenaar, N. and Van Der Heijden, P. G. (2019). Optimizing predictive performance of criminal recidivism models using registration data with binary and survival outcomes. *PLoS ONE*, 14(3), p. e0213245.

Torgo, L. (1997). Functional models for regression tree leaves. In *ICML*, Citeseer, 1990, pp. 385–393.

Touw, W. G., Bayjanov, J. R. and al, e. (2013). Data mining in the life science swith random forest. *Briefings in Bioinformatics*, 14(3), pp. 315–326.

Tsamardinos, I. et al. (2022). Just add data: automated predictive modeling for knowledge discovery and feature selection. *NPJ precision oncology*, 6(1), pp. 1–17.

UK Renal Registry (2020). 22nd annual report.

UK Renal Registry (UKRR) (2022). Uk renal registry 24th annual report. Tech. rep., Murray Research Archive, Bristol, UK, url https://ukkidney.org/audit-research/annual-report .

Uno, H., Cai, T., Pencina, M. J., D'Agostino, R. B. and Wei, L.-J. (2011). On the c-statistics for evaluating overall adequacy of risk prediction procedures with censored survival data. *Statistics in medicine*, 30(10), pp. 1105–1117.

Urquhart, R., Johnston, G., Abdolell, M. and Porter, G. A. (2015). Patterns of health care utilization preceding a colorectal cancer diagnosis are strong predictors of dying quickly following diagnosis. *BMC palliative care*, 14(1), p. 2.

Utkin, L., Konstantinov, A., Meldo, A., Ryabinin, M. and Chukanov, V. (2019a). A deep forest improvement by using weighted schemes. In *2019 24th Conference of Open Innovations Association (FRUCT)*, IEEE, pp. 451–456.

Utkin, L., Konstantinov, A., Meldo, A., Sokolova, V. and Coolen, F. (2021). The deep survival forest and elastic-net-cox cascade models as extensions of the deep forest. In *Proceedings of International Scientific Conference on Telecommunications, Computing and Control*, Springer, pp. 205–217.

Utkin, L. V. et al. (2019b). A weighted random survival forest. *Knowledge-Based Systems*, 177, pp. 136–144.

Utkin, L. V., Konstantinov, A., Lukashin, A. A. and Muliukha, V. A. (2020). An adaptive weighted deep survival forest. In *2020 XXIII International Conference on Soft Computing and Measurements (SCM)*, IEEE, pp. 198–201.

Vock, D. M., Wolfson, J. et al. (2016). Adapting machine learning techniques to censored time-to-event health record data. *Journal of Biomedical Informatics*, 61, pp. 119–31.

Wang, H. and Li, G. (2017). A selective review on random survival forests for high dimensional data. *Quantitative Bio-science*, 36(2), p. 85.

Wang, P., Li, Y. and Reddy, C. K. (2019). Machine learning for survival analysis: A survey. *ACM Computing Surveys (CSUR)*, 51(6), pp. 1–36.

Wang, Q. and Dinse, G. E. (2011). Linear regression analysis of survival data with missing censoring indicators. *Lifetime Data Analysis*, 17(2), pp. 256–279.

Wang, W. and Liu, W. (2018). Integration of gene interaction information into a reweighted random survival forest approach for accurate survival prediction and survival biomarker discovery. *Scientific Reports*, 8(1).

Wang, Y. and Witten, I. (1996). *Induction of model trees for predicting continuous classes*. Ph.D. thesis, University of Waikato.

Wang, Y., Su, H., Zhang, B. and Hu, X. (2018). Interpret neural networks by identifying critical data routing paths. In *proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8906–8914.

Waring, J., Lindvall, C. and Umeton, R. (2020). Automated machine learning: Review of the state-of-the-art and opportunities for healthcare. *Artificial intelligence in medicine*, 104, p. 101822.

Weeraddana, D., MallawaArachchi, S., Warnakula, T., Li, Z. and Wang, Y. (2020). Long-term pipeline failure prediction using nonparametric survival analysis. *arXiv preprint arXiv:201108671*.

Weinstein, B. G., Marconi, S., Bohlman, S. A., Zare, A. and White, E. P. (2019). Geographic generalization in airborne rgb deep learning tree detection. *bioRxiv*.

WHO (2022). Ageing and health.

Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics Bulletin*, 1, p. 80.

Witten, I. H., Frank, E., Hall, M. A., Pal, C. J. and DATA, M. (2005). Practical machine learning tools and techniques. In *Data Mining*, vol. 2.

Wongvibulsin, S., Wu, K. C. and Zeger, S. L. (2019). Clinical risk prediction with random forests for survival, longitudinal, and multivariate (rf-slam) data analysis. *BMC Medical Research Methodology*, 20.

Wright, M. N., Dankowski, T. and Ziegler, A. (2017). Unbiased split variable selection for random survival forests using maximally selected rank statistics. *Statistics in Medicine*, 36, pp. 1272–1284.

Xie, F. et al. (2022). Autoscore-survival: Developing interpretable machine learning-based time-to-event scores with right-censored survival data. *Journal of Biomedical Informatics*, 125, p. 103959.

Yamashita, R., Nishio, M., Do, R. K. G. and Togashi, K. (2018). Convolutional neural networks: an overview and application in radiology. *Insights into imaging*, 9(4), pp. 611–629.

Yang, S. and Prentice, R. (2010). Improved logrank-type tests for survival data using adaptive weights. *Biometrics*, 66(1), pp. 30–8.

Zacharaki, E. I., Morita, N. et al. (2012). Survival analysis of patients with high-grade gliomas based on data mining of imaging variables. *American Journal of Neuroradiology*, 33(6), pp. 1065–1071.

Zhang, L. et al. (2022). Prediction of prognosis in elderly patients with sepsis based on machine learning (random survival forest). *BMC Emergency Medicine*, 22(1), pp. 1–10.

Zhang, Y., Tiňo, P., Leonardis, A. and Tang, K. (2021). A survey on neural network interpretability. *IEEE Transactions on Emerging Topics in Computational Intelligence*.

Zhou, Z.-H. (2012). *Ensemble methods: foundations and algorithms*. CRC press.

Zhou, Z.-H. and Feng, J. (2019). Deep forest. *National Science Review*, 6(1), pp. 74–86.

Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, 67(2), pp. 301–320.

Appendix A

The values of the features' coefficients in the learned Cox regression models

In this Appendix, there are 11 tables reporting the values of the features' coefficients (the β values) in the Cox regression models learned from each of the 11 datasets. These β values were obtained by training the Cox regression algorithm on the full dataset (instead of using cross-validation), in order to maximise the quality of the learned model. In each table (i.e. for each dataset), the features are shown in decreasing order of the absolute (ignoring sign) value of the coefficient β. That is, the most important features in each model are placed at the top rows of the corresponding table. Note that, by considering absolute feature values, the importance of a feature depends only on the magnitude of its coefficient, without distinguishing between positive and negative β signs, but the signs are shown to allow a more precise interpretation of the feature's effect on the target variable. The meaning of each feature in these tables is defined in Chapter 3.

Table 37: The coefficients of the Cox regression model for the Alzheimer target variable of the ELSA dataset

Feature	coefficient
rtin=1	-7.9327087
mmcrre=2.0	-7.9269699
hdl=3	-7.5502405
sysval=4	-6.7253336
hasurg	-6.6699189
hdl=4	-6.5004141
mmssre=2	-6.173749
htfvc=3	-6.1061719
mmlsre=2.0	5.91917147
hba1c=3	-5.6734029
mmgsd_me=4	-5.4876862
htfev=3	-5.421895
hscrp=1	-5.2041246
hipval=4	-5.166779
rtin=2	-5.0725379
diaval=4	-4.9608085
apoe=2	-4.881852
cfib=3	-4.7877512
htpf=4	-4.6653746
bmival=4	-4.4747892
fglu=2	-4.4334273
fglu=3	-4.2450812
rtin=4	-4.1420562
htval=3	3.99831941
htval=2	3.96905735
hscrp=2	-3.8446216
trig=2	-3.6523373
hba1c=4	-3.6280704
apoe=3	-3.6153941
mmgsn_me=4	-3.6037633
htval=1	3.41851446
chol=4	-3.1074139
hgb=2	3.09944777
htval=4	-2.9749137
hgb=1	2.9423102
cfib=4	-2.8208819
wstval=3	2.70239253
hgb=4	2.6973123
rtin=3	-2.5868053
wtval=3	-2.3231511
mmftre2=4.0	-2.1372757
hgb=3	2.0656981
mapval=4	-2.0518362
whval=4	1.97348444
mapval=3	1.90356337
chol=3	1.85821828
ldl=2	-1.6949972
trig=3	-1.5811611
mapval=2	1.56359026

Feature	coefficient
ldl=4	-1.5459865
wtval=2	-1.5190695
scako=6.0	-1.441642
chol=2	1.40908569
hastro	1.39258523
hscrp=4	-1.3901362
mmstre=2.0	1.3058999
hba1c=2	1.26451507
sysval=2	-1.2043922
trig=4	-1.174429
mmftre2=3.0	-0.9900997
mmgsn_me=2	-0.9770861
scako=5.0	-0.9468894
ldl=3	-0.9287365
bmival=2	-0.9148555
pulval=1	0.89609719
mmstre=3.0	0.88298203
inhaler	-0.8611083
diaval=1	0.84351594
fglu=4	-0.8178241
mmgsn_me=1	-0.7790026
scako=2.0	-0.770892
mmrroc=4.0	0.7669385
mmftre2=5.0	0.75395418
ldl=1	-0.7423104
bmival=3	0.71759976

Feature	coefficient
mmlsre=3.0	-0.6419641
scako=8.0	-0.6242398
cfib=1	0.6031585
chol=1	0.59425249
fit	0.57362675
bmival=1	-0.5591869
wstval=1	0.55598391
mmrroc=2.0	0.55315667
scako=7.0	-0.5427747
fglu=1	-0.5345483
mmlore=2.0	0.5282357
htfev=1	-0.5250644
mmgsd_me=3	-0.5206794
wstval=4	-0.5192351
wtval=4	-0.5123828
mmcrre=3.0	0.50075705
mmlore=3.0	0.47939025
cfib=2	-0.4768621
wtval=1	-0.4694903
pulval=4	-0.4590249
mmftre2=2.0	0.41488052
hipval=1	0.36554053
whval=3	-0.3596172
hdl=2	-0.3592826
pulval=3	-0.3535901
clotb	0.3436563

Feature	coefficient
scako=4.0	-0.3272378
eyesurg	0.3230047
hdl=1	-0.3222967
htfvc=4	-0.3189674
smokerstat=3.0	0.3045343
pulval=2	0.28616469
diaval=2	0.28445655
htfev=4	-0.2771801
htfvc=1	0.27545266
sysval=1	-0.2442869
htfvc=2	-0.2384819
whval=2	-0.2367132
indsex	0.23118003
smokerstat=4.0	0.23056008
mmrroc=5.0	-0.22729
mmgsd_me=2	-0.2053699
hipval=3	0.20105205
mmrroc=3.0	0.19458657
htfev=2	-0.1897344
htpf=1	0.18758467
whval=1	-0.182184
mapval=1	0.17683598
scako=3.0	-0.1717864
trig=1	0.15773053
smokerstat=1.0	-0.1319435
mmgsn_me=3	0.13159401

Feature	coefficient
diaval=3	-0.1062664
confage	0.09681828
smokerstat=2.0	0.07851031
hipval=2	0.07322581
mmssre=3	-0.0700315
htpf=2	0.06466189
hba1c=1	-0.0605851
chestin	0.05166471
apoe=1	0.04983529
sysval=3	-0.0493828
mmgsd_me=1	-0.04847
htpf=3	-0.0435565
hscrp=3	-0.027277
wstval=2	-0.0072674
apoe=4	-0.002075

Table 38: The coefficients of the Cox regression for the Angina target variable of the ELSA dataset

Feature	coefficient
mmssre=2	-7.620346
htfvc=3	-7.1403567
rtin=2	-7.1167932
bmival=4	-6.731847
trig=2	-6.4477572
fglu=2	-6.4371299
htpf=4	-6.3214846
hscrp=1	-6.2475208
cfib=3	-6.2201957
trig=3	-5.9401698
apoe=3	-5.8026
hipval=4	-5.7681639
hba1c=3	-5.5334062
rtin=4	-5.225452
hba1c=4	-5.1101654
hgb=4	5.03883471
chol=4	-4.845758
trig=4	-4.6408961
htval=1	4.63675503
htval=3	4.63029121
rtin=3	-4.4902854
hgb=2	4.44301451
htval=2	4.37723035
hgb=3	4.33633379
hscrp=2	-4.2802617
cfib=4	-4.0545397
mapval=4	4.00127889
pulval=4	-3.9492577
hgb=1	3.78951284
htval=4	3.78432663
fglu=3	-3.7299687
htfvc=4	-3.5243833
diaval=4	-2.9702354
hastro	1.93370877
ldl=4	-1.7010552
mmftre2=5.0	-1.6344135
fglu=4	-1.4764844
mmgsd_me=2	-1.2084239
hdl=3	-1.159237
hasurg	1.07192906
mmgsd_me=4	-1.0507147
htfev=3	1.03901868
hba1c=2	1.01350818
mmlsre=3.0	1.00514769
hscrp=4	-0.9859174
wstval=3	-0.9616217
mmcrre=3.0	-0.9422017
fit	0.9341515
mmgsd_me=3	-0.9326254

Feature	coefficient
chol=3	0.89026713
hipval=3	0.87269346
sysval=3	0.84875961
apoe=2	0.84469026
mmgsd_me=1	-0.8235417
hdl=4	0.77159829
wtval=2	0.70552436
wtval=4	-0.6885426
trig=1	0.67089842
mmrroc=4.0	-0.6482446
whval=3	-0.6410551
scako=7.0	0.63500336
hipval=2	-0.6250662
mmcrre=2.0	0.59599421
wtval=3	0.58864211
htfvc=2	-0.5771085
smokerstat=1.0	0.55698044
chol=1	-0.5449089
smokerstat=2.0	0.53651735
hdl=2	-0.5345504
mapval=2	-0.5342824
ldl=3	-0.5246043
mmlsre=2.0	0.4879062
mmrroc=5.0	0.47754914
mmstre=2.0	-0.4770752
bmival=2	0.47691588

Feature	coefficient
bmival=3	-0.4701682
mapval=3	-0.4644559
scako=8.0	0.44793942
pulval=2	0.4452195
mmssre=3	-0.4324309
scako=6.0	0.43176391
sysval=4	-0.4108623
diaval=2	0.40193172
whval=1	-0.3734499
mmstre=3.0	-0.3690946
htpf=1	0.36597593
mmftre2=3.0	0.36508321
mmgsn_me=3	0.34382161
smokerstat=4.0	0.3426991
whval=2	-0.341516
chestin	0.31892058
cfib=2	-0.3186197
hba1c=1	0.30818925
wtval=1	0.30209513
inhaler	0.30190336
apoe=1	-0.2961453
wstval=2	-0.291414
htpf=2	0.29052164
mmrroc=3.0	-0.2793078
mapval=1	-0.2706839
htfev=1	0.27005148

Feature	coefficient
diaval=1	0.26152745
bmival=1	0.2550899
mmlore=3.0	0.25254342
mmlore=2.0	0.24022951
pulval=1	0.23828597
scako=4.0	-0.2259461
mmftre2=2.0	-0.2116881
hscrp=3	-0.2036734
mmrroc=2.0	0.19471245
wstval=4	-0.1945655
htfvc=1	-0.1649451
sysval=1	0.1643518
sysval=2	0.15961334
scako=5.0	-0.1461821
mmgsn_me=1	0.14354431
chol=2	-0.1396803
rtin=1	0.1346034
fglu=1	-0.1325183
hdl=1	-0.1307568
indsex	0.12679221
diaval=3	-0.1254126
htpf=3	0.12027138
htfev=2	0.09184959
hipval=1	0.08965017
mmgsn_me=2	-0.0888209
scako=2.0	-0.0755155

Feature	coefficient
scako=3.0	0.07493693
clotb	0.0739859
cfib=1	-0.0738325
pulval=3	0.0706568
smokerstat=3.0	0.0595946
mmgsn_me=4	0.04641292
mmftre2=4.0	0.04386294
confage	0.04365893
ldl=2	0.03441834
whval=4	-0.0317313
htfev=4	-0.0275974
wstval=1	0.00883225
eyesurg	-0.0054089
apoe=4	-0.0039168
ldl=1	0.00390816

Table 39: The coefficients of the Cox regression model for the Any-disease target variable of the ELSA dataset

Feature	coefficient
hscrp=4	10.175242
cfib=4	-7.3247284
apoe=4	-7.281966
trig=4	-7.2781148
hscrp=3	-6.8486819
pulval=4	-6.7941804
hipval=4	-6.6905569
hscrp=2	-6.5621927
htfvc=4	2.99980789
htfev=4	-1.7311636
fglu=2	1.5847906
hba1c=4	1.54303669
fglu=4	1.45577924
hba1c=3	1.31199089
hba1c=2	1.26538034
wtval=4	1.17794009
hgb=4	-1.162146
hdl=4	-0.9732054
ldl=4	-0.9645058
mmcrre=2.0	0.93114375
mapval=4	0.92818633
hastro	0.79893356
hgb=3	-0.7336361
hgb=2	-0.7140102
wstval=4	0.66526073
mmssre=2	0.64854145
whval=4	0.62370727
htfev=3	-0.6100218
mmrroc=2.0	0.57416159
bmival=4	0.57178443
mmstre=2.0	-0.5652856
apoe=2	0.55170861
wtval=3	0.53362028
eyesurg	0.51447618
mmssre=3	-0.5041489
hgb=1	-0.4715526
rtin=4	0.47151622
mmrroc=5.0	0.46881736
diaval=4	-0.4608419
wstval=3	0.45654897
hscrp=1	0.45007459
ldl=3	-0.4387779
mmftre2=4.0	0.41113188
htfvc=3	0.40356518
rtin=1	0.38195567
whval=1	0.36830648
fglu=1	0.36557705
sysval=3	0.36449683
pulval=3	0.35656706

Feature	coefficient
bmival=3	0.34954556
wtval=1	0.34436998
wtval=2	0.33874202
mmlsre=3.0	0.32174383
cfib=3	0.31530804
mmftre2=3.0	0.2993868
mmrroc=3.0	-0.2739087
htfvc=2	-0.2693475
wstval=2	0.26257594
mmgsn_me=4	0.25553167
sysval=4	0.25097335
whval=2	0.23583648
mmcrre=3.0	-0.2325463
mapval=3	-0.2278776
mmgsd_me=1	-0.2261859
chol=3	0.22341488
indsex	-0.2130369
hdl=2	-0.2065131
mmgsn_me=2	-0.190998
smokerstat=1.0	-0.1896403
hasurg	-0.1845854
mmgsd_me=4	-0.178294
htval=4	0.17554883
scako=7.0	0.17298339
chol=1	-0.1623666
apoe=3	-0.158563

Feature	coefficient
chol=4	0.1576282
scako=5.0	-0.1569162
trig=1	-0.1539875
hipval=1	0.1457227
whval=3	0.14095382
mmlore=3.0	-0.1381909
wstval=1	0.1322957
bmival=1	-0.13134
mmftre2=2.0	-0.1248535
hba1c=1	0.11687656
mmgsd_me=3	-0.1168035
mmgsd_me=2	-0.1050843
ldl=2	-0.1050524
chestin	-0.096234
bmival=2	0.08745771
htpf=4	-0.0796537
htval=1	0.07671792
diaval=3	-0.076626
chol=2	-0.0711572
scako=4.0	-0.0700388
hipval=3	-0.0695995
mmgsn_me=1	-0.069526
fit	-0.0681629
mmlsre=2.0	-0.0679914
smokerstat=3.0	0.06696189
mapval=1	-0.0643409

Feature	coefficient
mmgsn_me=3	-0.0643217
mmrroc=4.0	-0.0639756
rtin=2	0.06322902
htpf=3	-0.06257
hipval=2	0.06227592
hdl=1	-0.0537697
mmlore=2.0	0.0512987
diaval=2	0.05020946
pulval=1	0.04914189
htval=2	0.04818401
scako=8.0	0.04747473
mmstre=3.0	0.04430711
htval=3	0.0439132
mapval=2	0.04174562
inhaler	-0.0389644
scako=2.0	0.03109668
sysval=1	0.03028577
smokerstat=2.0	0.02962991
scako=6.0	0.02467921
cfib=1	0.02231149
cfib=2	0.02216864
confage	0.02104021
pulval=2	0.02027646
scako=3.0	-0.0178802
smokerstat=4.0	-0.0168148
htfev=2	0.01454268

Feature	coefficient
clotb	0.01332637
htfvc=1	0.01330201
mmftre2=5.0	-0.0091113
trig=2	-0.0074305
htpf=2	-0.0073617
htpf=1	-0.007052
sysval=2	-0.0064363
htfev=1	0.00602529
ldl=1	0.00597547
diaval=1	0.00494482
apoe=1	0.00297846
hdl=3	0.00201211
trig=3	0
fglu=3	0
rtin=3	0

Table 40: The coefficients of the Cox regression the for the Arthritis target variable of the ELSA dataset

Feature	coefficient
htfev=4	-14.011158
mapval=4	-8.1797771
htfvc=4	8.08704706
mmcrre=3.0	-7.8565215
cfib=4	-7.8127789
apoe=3	-7.4660541
hscrp=2	-7.2844382
pulval=4	-7.1434892
hba1c=4	-6.8114378
hscrp=3	-6.7371326
hgb=1	6.6755572
fglu=3	-6.4100873
hgb=2	6.25034009
hgb=3	6.23954961
fglu=4	-5.8644085
rtin=2	-5.7373329
hgb=4	5.68789999
hscrp=4	4.95366918
trig=4	-3.8413637
apoe=4	-3.7991784
wtval=4	2.16076795
rtin=3	2.01274027
sysval=4	1.53842205
mmssre=2	1.32861306
fglu=2	1.29246418
mmcrre=2.0	1.0756104
hasurg	-0.969319
cfib=3	0.85914092
trig=2	0.79987798
hdl=4	-0.7223741
mmrroc=2.0	0.64731177
ldl=4	0.64138755
rtin=4	0.63071497
wtval=3	0.60473112
hipval=4	0.59323011
htpf=4	0.55222879
mmgsd_me=4	-0.5110416
hipval=2	0.51021682
clotb	0.4993916
pulval=3	-0.4700368
mmrroc=4.0	-0.4634406
hba1c=2	0.45832405
mmlsre=3.0	-0.4549191
mmlore=3.0	0.444865
fglu=1	-0.4098736
rtin=1	0.40924104
sysval=3	0.40098342
wtval=2	0.39134129
bmival=4	0.37192635

Feature	coefficient
indsex	-0.3640852
mmgsd_me=3	-0.3526073
mmstre=2.0	-0.3437439
chestin	-0.3393918
wtval=1	0.33480863
sysval=1	0.33402958
fit	0.32594653
trig=1	-0.3217318
diaval=3	-0.2885614
wstval=4	-0.2852297
htval=2	0.27165459
hipval=3	-0.2569725
hastro	-0.2568866
htfev=2	0.25604496
mmgsn_me=2	-0.2532575
htval=1	0.25115472
mapval=3	0.24376804
mmlsre=2.0	-0.2290625
hscrp=1	-0.2190311
sysval=2	0.20876631
eyesurg	-0.1982571
apoe=2	0.19627947
wstval=2	0.18973998
mapval=1	-0.1852998
hipval=1	0.18148885
scako=4.0	-0.1679249

Feature	coefficient
chol=3	0.166488
ldl=1	0.16610444
mmgsd_me=1	-0.1344279
mmssre=3	0.12320835
htfev=3	0.11779203
smokerstat=2.0	0.11522651
scako=5.0	-0.114624
hdl=1	-0.1132758
whval=4	-0.1130608
smokerstat=3.0	-0.107939
scako=3.0	0.10677824
ldl=3	-0.1034086
scako=7.0	0.10154984
chol=2	0.10138919
htfvc=3	0.09991637
bmival=3	-0.0975439
scako=2.0	-0.0944119
htpf=1	0.09107622
apoe=1	0.09071346
mmstre=3.0	-0.0879797
diaval=4	0.08533211
mmrroc=3.0	-0.0849126
mmlore=2.0	-0.0843824
pulval=2	0.08314892
mmftre2=4.0	0.08264179
htfvc=2	-0.0809872

Feature	coefficient
cfib=2	0.07928901
htfev=1	0.07811997
htpf=2	-0.0776816
mmftre2=3.0	0.07236465
diaval=1	0.06986004
hdl=3	-0.0684205
diaval=2	0.06698506
mmgsn_me=3	0.06647406
mmgsd_me=2	-0.066292
smokerstat=1.0	0.06598008
bmival=2	-0.0650508
scako=8.0	-0.0594636
mmgsn_me=1	-0.0588609
wstval=3	0.0573271
mmrroc=5.0	-0.0527987
chol=1	0.04978863
inhaler	0.04646118
htpf=3	-0.0447267
whval=1	0.04020254
htfvc=1	-0.0399875
hba1c=1	0.0384582
whval=2	0.03773256
mmftre2=5.0	0.03755734
cfib=1	0.03656825
htval=4	-0.0338042
smokerstat=4.0	0.03361622

Feature	coefficient
scako=6.0	0.032209
hdl=2	-0.0288729
whval=3	-0.0278249
mmgsn_me=4	-0.0238836
pulval=1	-0.0190082
chol=4	-0.0185081
hba1c=3	0.0159767
htval=3	0.01486348
wstval=1	0.00867243
bmival=1	0.00625589
ldl=2	-0.0056927
mmftre2=2.0	0.0027789
mapval=2	-0.002081
confage	0.0005729
trig=3	0

Table 41: The coefficients of the Cox regression model for the Cancer target variable of the ELSA dataset

Feature	coefficient
hipval=4	-8.2452876
ldl=4	-8.0274891
wstval=4	-7.8748098
apoe=2	-7.6091816
pulval=4	7.51583836
fglu=2	-7.0420457
mapval=4	-7.0326264
apoe=3	-6.7910174
apoe=4	-5.849462
htfvc=4	-5.7972357
cfib=4	-5.7802624
rtin=4	-5.6571747
sysval=4	-5.5716617
htval=3	4.76206732
htval=2	4.73406011
htval=4	4.72051316
htval=1	4.70435873
fglu=4	4.2701582
htfev=4	-3.796319
hscrp=3	3.59630246
hscrp=4	-3.4155066
fglu=3	2.89951692
hba1c=4	2.30162259
rtin=3	2.22673977
mmftre2=5.0	-1.7911834
wtval=4	1.75855511
chol=4	1.75091546
hscrp=2	1.73232654
hgb=2	-1.7285295
hgb=3	-1.5807891
hgb=4	-1.5448762
wstval=3	1.1975817
hgb=1	-1.1394598
hdl=3	1.11419337
hba1c=2	-1.0175386
bmival=4	1.00447221
trig=4	-0.9476069
hba1c=3	-0.9272944
mmrroc=2.0	-0.8789287
htpf=4	-0.7899401
mmcrre=3.0	-0.7780309
mmrroc=5.0	0.74347694
mmgsn_me=4	0.74321071
trig=2	-0.697113
cfib=3	0.6560014
wtval=3	-0.6352387
htfev=3	0.5565223
mmgsd_me=4	-0.5218293
mmcrre=2.0	-0.4906893

Feature	coefficient
eyesurg	-0.4882548
mmgsn_me=2	0.48746474
sysval=3	0.46345803
pulval=2	-0.4520699
pulval=3	-0.4434157
whval=3	-0.4235174
hasurg	0.40636416
ldl=1	-0.3823337
wstval=2	0.37908728
mmrroc=4.0	0.37000613
htfvc=3	0.36957345
whval=2	-0.3605642
scako=7.0	0.33972706
cfib=2	0.33034145
mmssre=3	-0.3266951
hdl=4	0.32389203
hscrp=1	0.32068575
smokerstat=1.0	-0.2979384
inhaler	-0.2966227
chol=1	0.2950233
mmssre=2	-0.2899554
trig=3	-0.2845705
rtin=2	0.27305883
mmlore=3.0	0.27029039
diaval=4	0.25014016
smokerstat=4.0	0.24618251

Feature	coefficient
htfev=1	-0.2458878
smokerstat=2.0	0.23861071
mmlsre=3.0	-0.237581
mmftre2=4.0	-0.2348818
hdl=1	0.22296946
fit	-0.2217365
whval=4	0.21660553
htfvc=2	-0.2150219
pulval=1	-0.2124412
scako=5.0	0.21162905
clotb	-0.2102702
mapval=1	0.18787162
hipval=1	-0.1873355
smokerstat=3.0	0.17371119
hdl=2	-0.1633887
mmlsre=2.0	-0.1631944
diaval=3	-0.1600679
whval=1	-0.1529563
mmrroc=3.0	0.15101774
trig=1	0.1499507
wstval=1	0.14645337
sysval=1	-0.1332539
scako=4.0	0.13284452
scako=6.0	0.13117542
mmlore=2.0	0.13036918
mmgsn_me=1	0.12874554

Feature	coefficient
mmgsd_me=3	0.12686364
scako=3.0	0.1177874
mmstre=3.0	0.11718944
mmgsd_me=2	0.11654141
hipval=2	-0.1158105
scako=8.0	-0.1093979
hastro	-0.1087937
mapval=2	0.10786997
htpf=1	-0.1051121
hipval=3	0.10504098
mapval=3	-0.1001408
diaval=2	-0.0986155
apoe=1	0.09251814
htfev=2	-0.0923905
bmival=3	0.09174691
fglu=1	0.0900725
htpf=2	-0.0851641
chol=2	-0.0818859
mmstre=2.0	0.06674482
rtin=1	-0.0663849
indsex	-0.0650776
diaval=1	-0.0643342
ldl=2	-0.057042
mmftre2=2.0	-0.0534456
hba1c=1	-0.0524968
bmival=2	0.04627599

Feature	coefficient
chol=3	0.04531045
wtval=2	0.04337225
htpf=3	-0.0415451
sysval=2	0.04056033
confage	0.03724313
ldl=3	-0.0307623
bmival=1	-0.0240992
chestin	0.01909158
mmftre2=3.0	-0.0189348
scako=2.0	0.01759936
mmgsd_me=1	-0.0169962
htfvc=1	0.01570281
wtval=1	0.01026843
cfib=1	-0.0083243
mmgsn_me=3	-0.0068433

Table 42: The coefficients of the Cox regression model for the Diabetes target variable of the ELSA dataset

Feature	coefficient
trig=4	-8.4931402
hdl=3	-8.0991794
eyesurg	-7.854386
hipval=4	-7.8396572
trig=3	-7.690475
hgb=1	7.07996624
hgb=4	6.85248945
hgb=3	6.80771278
hgb=2	6.60101737
wstval=4	-6.593431
ldl=4	-6.5862892
hdl=4	-6.2458492
apoe=4	-5.2239055
rtin=3	-5.1899831
cfib=4	-4.9369648
htval=2	4.9128543
htval=1	4.89941328
htval=4	4.73285337
htval=3	4.57440509
hba1c=4	4.06503237
pulval=4	4.01378987
fglu=4	3.40820086
hba1c=2	3.40302538
mapval=4	-3.3640815
hba1c=3	3.32044016
apoe=3	3.01561979
hscrp=2	2.71929499
fglu=2	2.31688282
hscrp=3	-1.9463122
hscrp=4	-1.8951141
sysval=3	1.89357793
chol=4	1.88604993
cfib=3	-1.781083
wtval=4	1.70732962
trig=2	-1.6594976
hba1c=1	1.63927083
rtin=4	1.58526022
diaval=4	1.57375159
fglu=1	1.55982253
sysval=4	1.5279631
apoe=2	1.39682298
htfev=3	-1.3364883
sysval=2	1.27944635
bmival=4	1.26555284
mapval=3	-1.2334491
rtin=2	1.16014236
wstval=2	1.13067969
htfev=4	1.11260447
hdl=2	-1.0574251

Feature	coefficient
bmival=2	1.05445125
mmlsre=3.0	0.94336832
sysval=1	0.93531919
htfev=2	-0.9235308
fit	-0.9215452
mmgsd_me=4	-0.8683379
wstval=1	0.81069918
hastro	-0.7556557
bmival=3	0.75342691
hasurg	-0.703779
wstval=3	0.69930543
scako=2.0	-0.6992957
diaval=3	0.69042985
mapval=2	-0.6831388
chol=2	0.6778633
htfvc=4	0.67166654
mmgsn_me=4	0.6510526
mmgsd_me=2	-0.6339463
mmssre=2	0.60968188
mmcrre=3.0	0.59748233
ldl=3	-0.5882059
hipval=3	0.55801955
indsex	0.55661944
whval=4	0.5537927
htfvc=3	0.55322262
clotb	0.51084417

Feature	coefficient
wtval=3	0.50170863
chol=1	0.50081276
mmftre2=5.0	0.49942109
mapval=1	-0.4922883
mmrroc=3.0	-0.4553108
bmival=1	0.45047159
pulval=3	0.44534771
hscrp=1	0.42320127
chol=3	0.38276927
mmcrre=2.0	-0.3732762
diaval=2	0.36078623
mmgsn_me=2	-0.3512996
mmgsn_me=3	-0.3499895
wtval=1	0.34874214
mmgsn_me=1	-0.3369265
mmgsd_me=1	-0.333085
scako=3.0	-0.332479
trig=1	0.33071026
chestin	-0.3274855
htfev=1	-0.3273794
mmrroc=4.0	-0.3112836
mmgsd_me=3	-0.3075974
wtval=2	0.30187368
scako=7.0	0.2931687
whval=3	0.27603413
hdl=1	-0.2667309

Feature	coefficient
mmftre2=2.0	-0.2630407
pulval=2	0.25576537
hipval=2	-0.2491735
pulval=1	0.24634066
mmstre=3.0	0.24582036
diaval=1	0.23664407
scako=4.0	-0.2322434
apoe=1	0.23211082
ldl=2	-0.2285895
htpf=1	-0.225706
mmrroc=2.0	0.21929646
rtin=1	0.21299376
htpf=2	-0.2064246
smokerstat=1.0	-0.1785303
htpf=3	-0.1735736
mmrroc=5.0	-0.1727088
mmlsre=2.0	-0.1702601
mmftre2=3.0	0.15994581
mmssre=3	-0.1491508
inhaler	0.12162553
smokerstat=2.0	-0.1202445
hipval=1	-0.1155659
cfib=2	-0.1150687
whval=2	0.1057846
smokerstat=4.0	0.10561814
mmlore=2.0	0.07763565

Feature	coefficient
cfib=1	0.07756603
mmstre=2.0	0.07518869
smokerstat=3.0	-0.0626358
htfvc=1	0.05758673
scako=8.0	-0.0560216
htfvc=2	0.03746714
mmftre2=4.0	-0.0370658
confage	-0.0282241
scako=5.0	-0.0259621
scako=6.0	-0.0226694
ldl=1	0.01553887
htpf=4	0.01268887
whval=1	-0.008542
mmlore=3.0	-0.0029223
fglu=3	0

Table 43: The coefficients of the Cox regression model for the Heart Attack target variable of the ELSA dataset

Feature	coefficient
hscrp=3	9.55510165
htfev=3	-8.4327836
hscrp=1	-7.8960106
mmrroc=2.0	-7.8818519
wstval=4	7.85608125
mmcrre=3.0	-7.6775028
chol=4	-7.5584524
fglu=2	-7.5365534
diaval=4	-6.8283568
hdl=4	-6.7647871
trig=2	-6.5869858
hipval=4	-6.5691024
rtin=2	-6.4446354
htval=1	6.10457203
hba1c=3	-6.0665986
htval=4	6.04597009
bmival=4	-5.8387027
hba1c=4	-5.8071108
htval=3	5.76008219
htval=2	5.7174984
wtval=4	5.54550077
cfib=3	-5.3708198
apoe=3	-5.2597361
trig=3	-5.1395446
rtin=3	-5.0774648
fglu=3	-5.0257985
htfev=4	-4.9931383
sysval=4	-4.7659409
mapval=4	-3.7793902
cfib=4	-3.5890539
trig=4	-3.5790824
whval=4	-3.2395789
hscrp=2	-3.0153586
rtin=4	2.66482481
hgb=1	-2.4066381
htfvc=4	-2.3901666
sysval=3	1.94912874
hgb=3	-1.8558188
hgb=2	-1.8372993
htfvc=3	1.5565591
mmlsre=3.0	1.49791011
mmlore=3.0	-1.4927359
apoe=2	1.43442966
ldl=3	1.33079821
mmgsn_me=2	-1.2919595
smokerstat=3.0	-1.2487655
mmcrre=2.0	1.2000419
hipval=3	-1.1447645
wtval=3	1.12959894

Feature	coefficient
hasurg	1.11215748
hgb=4	-1.0955626
fglu=4	-1.0095984
sysval=2	0.99681685
whval=2	-0.9682671
sysval=1	0.96176938
ldl=2	0.93527204
scako=2.0	-0.9301825
mmgsd_me=4	0.90342811
mmrroc=5.0	0.87827826
whval=3	-0.8751973
mapval=2	0.87129307
mmgsn_me=1	-0.8355116
whval=1	-0.8263152
fit	0.82375971
indsex	0.80499951
hba1c=2	0.79698972
chol=2	-0.7734565
mmssre=3	0.77099869
hscrp=4	-0.7472192
htfev=2	-0.741358
mmgsd_me=3	0.69270686
mmftre2=3.0	-0.6922797
mmlsre=2.0	0.68482136
ldl=1	0.66882821
mmgsn_me=4	-0.6445939

Feature	coefficient
mmftre2=5.0	-0.6366131
mmgsn_me=3	-0.6240749
diaval=3	-0.6133503
diaval=2	-0.6122113
chol=1	-0.6110988
chol=3	-0.6090703
mmrroc=4.0	-0.5993254
mmstre=2.0	0.58853233
scako=5.0	-0.5641538
pulval=4	-0.5552109
mmgsd_me=2	0.55416864
bmival=3	0.54193292
pulval=3	-0.5063226
apoe=1	0.49917708
bmival=2	0.49628306
hdl=3	0.49609891
htpf=4	0.49264634
wstval=3	-0.4909923
mmssre=2	0.47428221
ldl=4	-0.4709071
hipval=1	0.4515722
hdl=2	-0.433986
smokerstat=4.0	0.42542163
cfib=2	0.41302628
rtin=1	-0.379957
htfvc=2	-0.3780664

Feature	coefficient
mmftre2=4.0	-0.3760799
trig=1	-0.3737206
hdl=1	-0.3735558
pulval=1	0.36843602
hba1c=1	0.36047713
scako=7.0	-0.360204
mmgsd_me=1	0.34519259
mapval=1	0.32433482
eyesurg	0.28459904
smokerstat=1.0	-0.2658444
htpf=3	0.24292218
mapval=3	0.21387381
hastro	-0.209175
clotb	0.19485712
smokerstat=2.0	-0.1866691
htpf=1	-0.1827455
chestin	0.17113616
scako=8.0	0.164426
diaval=1	-0.1598116
wstval=2	-0.1567578
fglu=1	-0.1440142
htfvc=1	-0.1327861
htpf=2	0.13097423
wtval=1	-0.1308862
wtval=2	0.12774399
scako=6.0	0.0979871

Feature	coefficient
mmftre2=2.0	-0.094681
mmrroc=3.0	0.08706727
bmival=1	0.07200919
scako=3.0	-0.0675788
mmstre=3.0	-0.0570043
cfib=1	-0.0529962
pulval=2	-0.0412388
confage	0.03475355
hipval=2	0.02034183
inhaler	-0.0178579
wstval=1	-0.0134858
scako=4.0	-0.0114631
mmlore=2.0	-0.0107384
htfev=1	0.00592468
apoe=4	-0.000208

Table 44: The coefficients of the Cox regression model for the Psychiatric disorder target variable of the ELSA dataset

Feature	coefficient
htfev=3	-14.688214
htfev=4	-12.451733
htfvc=4	10.2327462
hasurg	-7.7098603
htfvc=3	-7.3978565
mmgsn_me=4	-6.9239873
hba1c=3	-6.8434996
mmgsd_me=4	-6.7745884
whval=4	-6.7383442
hdl=4	-6.651327
hba1c=4	-6.4584527
trig=3	-6.4533377
pulval=4	-6.4454495
fglu=2	-6.3539214
cfib=3	-6.1753572
wtval=4	-5.984573
hscrp=1	-5.9695818
trig=4	-5.9618291
hgb=4	5.95195657
hgb=1	5.9206951
hgb=3	5.75760223
hgb=2	5.73195221
hipval=4	-5.7057825
chol=4	-5.4273047
rtin=4	-5.0537688
cfib=4	-4.9122836
rtin=2	-4.7720886
fglu=3	-4.7471458
rtin=3	-4.3535287
fglu=4	-3.8317822
hscrp=2	-3.5098877
sysval=4	2.87732896
wstval=4	-2.3172506
apoe=3	2.19333365
pulval=3	-2.0875423
mmlsre=3.0	1.58608179
hastro	1.44746455
ldl=3	-1.4473424
mapval=3	-1.2977384
apoe=2	1.24272269
sysval=3	1.23554147
wstval=3	-1.2311962
pulval=2	-1.1696701
bmival=4	1.11613279
hscrp=4	-0.9905284
mmcrre=2.0	0.93826703
mmrroc=5.0	0.92325603
trig=1	0.86037404
chol=3	0.85259774

Feature	coefficient
mmgsd_me=3	0.82320393
ldl=1	0.77339227
mmgsn_me=3	-0.7718106
trig=2	-0.7672726
scako=2.0	-0.7366359
mapval=1	-0.7317698
apoe=1	-0.7164139
ldl=4	-0.7110988
pulval=1	-0.7099274
eyesurg	0.64938263
apoe=4	-0.6479222
diaval=4	0.62493468
mapval=4	-0.6132614
hba1c=2	0.61298221
htval=3	0.57900334
sysval=2	0.51062896
mapval=2	-0.5010168
scako=5.0	-0.468974
htval=1	0.46557047
ldl=2	0.44135811
cfib=2	-0.4191519
bmival=3	0.39868035
smokerstat=2.0	0.38106166
whval=1	-0.3768646
mmssre=3	-0.3743048
rtin=1	-0.3724284

Feature	coefficient
hipval=2	-0.3685294
indsex	-0.3645478
mmssre=2	0.35262586
whval=3	-0.3510723
diaval=1	0.34957404
clotb	0.34368643
hdl=3	-0.3426565
whval=2	-0.3407074
hdl=2	-0.3398684
mmrroc=2.0	0.33532988
htfev=1	-0.3334911
smokerstat=3.0	0.32591308
scako=4.0	-0.3188103
htval=4	0.31295204
mmftre2=5.0	0.30867723
htpf=3	-0.2873622
wtval=1	0.28341527
mmgsd_me=1	0.2834072
hdl=1	-0.2832113
diaval=3	0.27168051
fglu=1	0.26282379
mmlsre=2.0	0.23145593
mmftre2=3.0	-0.2291707
htpf=4	-0.2263165
inhaler	0.21941085
mmlore=3.0	0.2107462

Feature	coefficient
mmftre2=4.0	0.20990814
bmival=1	-0.203669
sysval=1	0.20352343
mmcrre=3.0	0.20313842
htfev=2	-0.1993733
mmstre=3.0	0.19283612
wstval=1	-0.1885475
scako=7.0	0.18418051
mmgsn_me=2	-0.1667051
mmstre=2.0	0.16487559
chol=1	-0.1536729
hscrp=3	-0.1461413
wstval=2	-0.1195775
chestin	-0.1018395
smokerstat=1.0	0.10112534
htfvc=2	0.10098114
wtval=2	0.09958401
htval=2	0.09910937
mmlore=2.0	-0.091012
fit	0.08500947
mmrroc=4.0	0.08384553
smokerstat=4.0	0.08263621
scako=6.0	0.07590401
chol=2	0.07464007
mmrroc=3.0	0.06743386
hipval=1	0.06493724

Feature	coefficient
cfib=1	-0.0648313
htfvc=1	0.05469485
diaval=2	0.0508063
htpf=2	-0.0410051
wtval=3	0.02886178
scako=8.0	-0.0282733
mmftre2=2.0	0.02599605
mmgsd_me=2	0.02318575
scako=3.0	-0.0224732
htpf=1	-0.0208927
bmival=2	0.01836079
confage	-0.0157655
mmgsn_me=1	0.01202925
hba1c=1	0.01045744
hipval=3	-0.0103812

Table 45: The coefficients of the Cox regression model for the Stroke target variable of the ELSA dataset

Feature	coefficient
htfev=4	-13.901515
rtin=2	-11.395533
hscrp=4	-8.5136698
htfvc=4	8.01989058
mmrroc=2.0	-7.8908482
mmgsn_me=4	-7.7858115
hscrp=1	-7.0186781
ldl=4	-6.7979294
hdl=4	-6.7220709
apoe=4	-6.5717586
apoe=3	-6.4010841
trig=3	-6.1750347
htfev=3	-6.142761
cfib=3	-6.1153124
hba1c=4	-6.1140794
pulval=4	-5.7838049
bmival=4	-5.6542283
hipval=4	-5.3886602
rtin=4	-5.3577766
fglu=3	-5.3057645
rtin=3	-5.1253689
hscrp=2	-4.5892514
fglu=4	-4.3177877
trig=4	-4.2247604
cfib=4	3.59998765
mmlsre=3.0	2.40507919
mmssre=2	2.12069561
htpf=4	2.01091091
wstval=4	-1.9257757
hgb=3	-1.7566493
hgb=1	-1.7202655
chol=4	1.64798371
htval=4	-1.5419884
hgb=2	-1.485796
hgb=4	-1.4108951
hdl=3	-1.3783941
sysval=4	1.23620219
sysval=3	1.20504716
htval=1	-1.192686
mmgsn_me=2	-1.0967337
hipval=3	-1.0425371
mmstre=2.0	1.00550495
mmstre=3.0	1.00280284
htval=3	-0.9535941
chol=3	-0.9481754
wtval=4	-0.9399198
htval=2	-0.9276249
mmftre2=3.0	-0.920535
wtval=3	-0.8518606

Feature	coefficient
diaval=4	-0.8122189
ldl=3	-0.8111655
chol=1	-0.7658644
fglu=2	0.72900521
mmgsn_me=1	-0.6766683
trig=2	0.66291011
sysval=2	0.65703385
mmlsre=2.0	0.63614
mmrroc=5.0	0.61868571
bmival=3	0.60685624
htfvc=3	0.60539429
mmgsn_me=3	-0.5951974
diaval=1	-0.5945777
mmgsd_me=3	0.59301957
diaval=2	-0.5792313
mmgsd_me=2	0.55756268
bmival=2	0.54195863
mapval=4	0.51117283
mmssre=3	0.46873191
mapval=2	0.45523885
mmftre2=5.0	-0.4530132
mmgsd_me=4	0.43589715
hipval=2	-0.4234598
sysval=1	0.42183012
pulval=3	-0.4127403
mmftre2=4.0	-0.402873

Feature	coefficient
smokerstat=4.0	0.39110889
wstval=2	-0.3690123
fglu=1	0.36821341
eyesurg	0.36647427
whval=4	0.36343036
mapval=1	0.35755115
scako=4.0	-0.3522506
trig=1	0.33779746
smokerstat=3.0	0.33257032
hdl=1	-0.3305968
cfib=2	0.32814076
htfev=2	-0.326723
mmgsd_me=1	0.32511917
scako=6.0	-0.3167792
wstval=3	0.2982155
ldl=1	0.2955625
clotb	0.29514041
hscrp=3	-0.2910935
hasurg	-0.280883
chol=2	-0.2798368
hdl=2	-0.2680231
mmlore=2.0	0.26685822
cfib=1	0.26072993
mmlore=3.0	0.25685592
diaval=3	-0.2543801
wstval=1	-0.2484339

Feature	coefficient
mmcrre=3.0	-0.2465687
apoe=1	-0.2416889
scako=2.0	-0.2327551
htfvc=2	-0.2269477
scako=8.0	0.2194743
mmrroc=3.0	0.19675418
hba1c=2	-0.190899
indsex	0.17935618
whval=2	0.17688573
pulval=1	0.16587928
smokerstat=2.0	0.15088176
bmival=1	0.14422689
ldl=2	-0.1311953
scako=3.0	0.12237501
htpf=3	0.12053097
wtval=2	-0.1188908
mmcrre=2.0	-0.1134774
whval=3	0.10816614
hba1c=3	-0.1054497
whval=1	-0.1015949
rtin=1	0.10063758
hipval=1	-0.0989686
hba1c=1	-0.0978006
apoe=2	-0.0963757
htpf=1	-0.0873529
mapval=3	-0.0863727

Feature	coefficient
scako=7.0	0.06944343
pulval=2	0.06873757
inhaler	-0.0543166
hastro	0.0504157
mmrroc=4.0	0.0441866
fit	-0.0430341
confage	0.03830532
wtval=1	0.03749159
mmftre2=2.0	-0.0353639
htfvc=1	-0.0325378
htpf=2	0.03200552
htfev=1	0.02177943
scako=5.0	-0.0154971
chestin	0.01290678
smokerstat=1.0	-0.0012738

Table 46: The coefficients of the Cox regression model for the Any-disease target variable of the SHARE dataset

Feature	coefficient
maxgrip=4	-0.4425834
bmi=2	0.33366131
mobilityind=3.0	0.22161766
lgmuscle=4.0	0.21851409
bmi=1	0.20884418
mobilityind=2.0	0.2001368
lgmuscle=3.0	0.18909584
mobilityind=1.0	0.1821891
mobilityind=4.0	0.17157139
lgmuscle=2.0	0.17133513
iadlza=5.0	-0.1710163
grossmotor=4.0	-0.1687743
lgmuscle=1.0	0.16102185
grossmotor=3.0	-0.1546868
bmi=3	0.15173584
adla=5.0	0.14595222
iadlza=3.0	-0.1387141
casp=1	0.13300752
iadlza=4.0	-0.1267138
casp=2	0.1048609
grossmotor=2.0	-0.1012363
ever_smoked	0.09443105
br015_=4.0	0.09108701
bmi=4	0.08393774
casp=3	0.07570523
adla=4.0	0.07547842
maxgrip=3	-0.0599051
iadlza=2.0	-0.0598788
br015_=3.0	0.05931362
br015_=2.0	0.05299952
br010_mod=2.0	0.05141445
female	0.04839139
grossmotor=1.0	-0.0467667
smoking	-0.0457124
br010_mod=7.0	0.0395388
adla=3.0	0.03700037
br010_mod=3.0	0.03324835
br010_mod=6.0	0.02861607
iadlza=1.0	-0.0231415
adla=1.0	0.02106356
finemotor=1.0	0.0189295
age	0.01892203
br010_mod=5.0	0.0176155
finemotor=2.0	0.01565171
maxgrip=1	-0.015019
finemotor=3.0	0.0135096
br010_mod=4.0	0.01329486
adla=2.0	0.01074344
casp=4	-0.0081846

Feature	coefficient
maxgrip=2	0.00804222
ldl=3	-0.8111655
chol=1	-0.7658644
fglu=2	0.72900521
mmgsn_me=1	-0.6766683
trig=2	0.66291011
sysval=2	0.65703385
mmlsre=2.0	0.63614
mmrroc=5.0	0.61868571
bmival=3	0.60685624
htfvc=3	0.60539429
mmgsn_me=3	-0.5951974
diaval=1	-0.5945777
mmgsd_me=3	0.59301957
diaval=2	-0.5792313
mmgsd_me=2	0.55756268
bmival=2	0.54195863
mapval=4	0.51117283
mmssre=3	0.46873191
mapval=2	0.45523885
mmftre2=5.0	-0.4530132
mmgsd_me=4	0.43589715
hipval=2	-0.4234598
sysval=1	0.42183012
pulval=3	-0.4127403
mmftre2=4.0	-0.402873

Feature	coefficient
smokerstat=4.0	0.39110889
wstval=2	-0.3690123
fglu=1	0.36821341
eyesurg	0.36647427
whval=4	0.36343036
mapval=1	0.35755115
scako=4.0	-0.3522506
trig=1	0.33779746
smokerstat=3.0	0.33257032
hdl=1	-0.3305968
cfib=2	0.32814076
htfev=2	-0.326723
mmgsd_me=1	0.32511917
scako=6.0	-0.3167792
wstval=3	0.2982155
ldl=1	0.2955625
clotb	0.29514041
hscrp=3	-0.2910935
hasurg	-0.280883
chol=2	-0.2798368
hdl=2	-0.2680231
mmlore=2.0	0.26685822
cfib=1	0.26072993
mmlore=3.0	0.25685592
diaval=3	-0.2543801
wstval=1	-0.2484339

Feature	coefficient
mmcrre=3.0	-0.2465687
apoe=1	-0.2416889
scako=2.0	-0.2327551
htfvc=2	-0.2269477
scako=8.0	0.2194743
mmrroc=3.0	0.19675418
hba1c=2	-0.190899
indsex	0.17935618
whval=2	0.17688573
pulval=1	0.16587928
smokerstat=2.0	0.15088176
bmival=1	0.14422689
ldl=2	-0.1311953
scako=3.0	0.12237501
htpf=3	0.12053097
wtval=2	-0.1188908
mmcrre=2.0	-0.1134774
whval=3	0.10816614
hba1c=3	-0.1054497
whval=1	-0.1015949
rtin=1	0.10063758
hipval=1	-0.0989686
hba1c=1	-0.0978006
apoe=2	-0.0963757
htpf=1	-0.0873529
mapval=3	-0.0863727

Feature	coefficient
scako=7.0	0.06944343
pulval=2	0.06873757
inhaler	-0.0543166
hastro	0.0504157
mmrroc=4.0	0.0441866
fit	-0.0430341
confage	0.03830532
wtval=1	0.03749159
mmftre2=2.0	-0.0353639
htfvc=1	-0.0325378
htpf=2	0.03200552
htfev=1	0.02177943
scako=5.0	-0.0154971
chestin	0.01290678
smokerstat=1.0	-0.0012738

Table 47: The coefficients of Cox regression for the Haemodialysis dataset

Feature	coefficient
CREA=4	-9.6307633
Transferrin=1	-7.7892497
FERR=3	-7.609931
Transferrin=2	-6.8974805
PTH=4	-6.5590497
FERR=4	-6.3571868
MAMC=4	-5.9472031
CRP=4	-5.4476195
CHOL=4	3.44590315
FERR=1	3.14853846
PTH=2	2.28176325
HBa1C=4	2.19746821
Urea=4	2.10409372
CharlsonScore=4	1.54636034
ALB=4	1.47890027
HB=4	-1.4361246
CRP=2	-1.3301905
PO4=3	1.3247969
TRAN=4	1.26607276
TRAN=3	1.21302843
Myeloma_Amyloid	-1.0615495
MAC=3	-1.0555043
MAC=4	-1.0298259
CRP=3	0.91444318
CharlsonScore=3	0.82655811
Chol_HDL=4	0.81208059
PostBMI=3	-0.8112887
CA=3	-0.802152
B12=3	-0.7690683
CHOL=3	-0.7434584
CA=4	-0.6468105
TRAN=2	0.60091144
ALB=2	0.58863054
HDL=4	0.58617728
TSF=1	-0.5329514
CREA=2	-0.495134
CHOL=2	0.49200419
PostBMI=4	-0.4690498
MAMC=3	0.45946969
ALB=3	0.4563885
TIN	0.45390577
CharlsonScore=2	0.42006248
estMetabolicRate=3	0.39803601
ALB=1	0.38997504
GripStrength=4	-0.3849983
B12=4	0.35230222
MAC=2	-0.347721
CREA=3	0.34516018
PostBMI=1	-0.3370802
FallsRisk=3.0	-0.331693

Feature	coefficient
MAMC=1	0.3305329
MAC=1	-0.294227
Transferrin=4	-0.2881953
TSF=3	-0.282225
Diabetic+Nephropathy	-0.2716045
Renal_Vascular_Disease	0.26915399
PTH=1	0.26331246
Chol_HDL=2	0.26252815
CRP=1	-0.2590165
MAMC=2	0.25529349
HBa1C=3	0.25052759
PostBMI=2	-0.2257672
Acute_HD	-0.2250627
CREA=1	-0.2194862
GripStrength=1	-0.2179637
HDL=2	-0.2049712
TxpStat	-0.1909532
ESR=2	0.18870192
CharlsonScore=1	0.18073845
K=4	-0.1798042
HBa1C=1	-0.1773377
TRAN=1	-0.1750703
HB=3	-0.174307
PTH=3	0.17323733
HCO3=4	0.17104585
estMetabolicRate=4	-0.1648629

Feature	coefficient
CHOL=1	-0.1601672
B12=2	0.15683723
Structural_Renal_Disease	0.14592599
ESR=1	-0.1116703
CA=1	-0.1105436
Primary_GN	0.10984405
Diabetes	0.09469222
GripStrength=3	0.08906317
ESR=4	0.08901845
TSF=2	0.08175234
HBa1C=2	0.08163238
PO4=1	-0.0792953
HDL=3	0.07660908
EPO=3.0	0.07510639
HB=2	-0.072854
HB=1	-0.0590355
K=3	0.0566431
K=1	0.05640102
HDL=1	0.055072
estMetabolicRate=2	-0.0518765
Urea=1	0.04710159
PO4=4	0.04008387
AgeFD	0.03920714
EPO=2.0	-0.0364014
K=2	-0.0356546
CA=2	-0.0319856

Feature	coefficient
estMetabolicRate=1	0.0277908
Chol_HDL=3	0.02723692
Chol_HDL=1	-0.0264698
GripStrength=2	-0.026221
TSF=4	-0.0198872
B12=1	0.01396124
FallsRisk=2.0	0.00810364
PO4=2	-0.0052998
ESR=3	0.00506127
Urea=3	0
Transferrin=3	0
FERR=2	0
HCO3=1	0
HCO3=2	0
HCO3=3	0
Urea=2	0
whval=3	0.10816614
hba1c=3	-0.1054497
whval=1	-0.1015949
rtin=1	0.10063758
hipval=1	-0.0989686
hba1c=1	-0.0978006
apoe=2	-0.0963757
htpf=1	-0.0873529
mapval=3	-0.0863727
scako=7.0	0.06944343

Feature	coefficient
pulval=2	0.06873757
inhaler	-0.0543166
hastro	0.0504157
mmrroc=4.0	0.0441866
fit	-0.0430341
confage	0.03830532
wtval=1	0.03749159
mmftre2=2.0	-0.0353639
htfvc=1	-0.0325378
htpf=2	0.03200552
htfev=1	0.02177943
scako=5.0	-0.0154971
chestin	0.01290678
smokerstat=1.0	-0.0012738