University of Alberta

LEARNING ACCURATE REGRESSORS FOR PREDICTING SURVIVAL TIMES OF INDIVIDUAL CANCER PATIENTS

by

Hsiu-Chin Lin

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science

Spring 2011
Edmonton, Alberta

Examining Committee

Russell Greiner, Computing Science

Vickie Baracos, Oncology

Joerg Sander, Computing Science

Abstract

Survival prediction is the task of predicting the length of time that an individual patient will survive; accurate predictions can give doctors better guidelines on selecting treatments and planning futures. This differs from the standard *survival analysis*, which focuses on *population-based* studies and tries to discover the prognostic factors and/or analyze the median survival times of different groups of patients.

The objective of our work, *survival prediction*, is different: to find the most accurate model for predicting the survival times for each *individual* patient. We view this as a regression problem, where we try to map the features for each patient to his/her survival time. As the relationship between features and survival time is still not understood, we consider various ways to *learn* these models from historical patient records. This is challenging in medical/clinical data due to the presence of irrelevant features, outliers, and missing class labels. This dissertation describes our approach for overcoming these, and other challenges, producing techniques that can predict survival times.

We focus our experiments on a data set of 2402 patients, including 1260 censored patients (i.e., whose survival time is not known). Our approach consists of two major steps. In the first step, we apply various grouping methods to divide the data set into smaller populations. In the second step, we apply different regression models to each sub-group we obtained from the first step. Our experiments show that the *linear regression*, the *support vector regression*, and the *gating regression* are effective: each predictor can obtain an *average cross validated relative absolute error* lower than 0.54 (where the *average relative absolute error* of a regressor is $E[\frac{|t-p|}{p}]$ where t is the true survival time and p is our prediction for each patient). We also use our regressors to classify each patient into "long survivor" versus "short survivor" where the classification boundary is the median survival time of the entire population; here, we show that several regressors can achieve at least 70% accuracy. These experimental results verify that we can effectively predict patients' survival times with a combination of statistical and machine learning approaches.

Table of Contents

1 Introduction **1**
- 1.1 The Problem . . . 2
- 1.2 Motivation . . . 2
- 1.3 Challenges . . . 3
- 1.4 Contributions . . . 4
- 1.5 Outline . . . 4

2 Background **6**
- 2.1 Survival Analysis . . . 7
  - 2.1.1 Medical Prognosis . . . 8
- 2.2 Survival Data . . . 8
  - 2.2.1 Censored Observations . . . 9
  - 2.2.2 Formulation of Survival Data . . . 9
- 2.3 Modelling Survival Distribution . . . 10
  - 2.3.1 Kaplan-Meier Estimator . . . 10
  - 2.3.2 Cox Proportional Hazard Model . . . 12
  - 2.3.3 Logistic Regression . . . 14
  - 2.3.4 Evaluation and Validation . . . 15
- 2.4 Related Work . . . 20
  - 2.4.1 Previous Work in Survival Analysis . . . 21
  - 2.4.2 Previous Research in Artificial Intelligence . . . 22
  - 2.4.3 Previous Work in Machine Learning . . . 22

3 Evaluation **25**
- 3.1 Cross-Validation . . . 26
- 3.2 Evaluating Predictions . . . 27
  - 3.2.1 L1 and L2 Error . . . 27
  - 3.2.2 Relative Absolute Error . . . 28
  - 3.2.3 Concordance Index . . . 29
  - 3.2.4 Correlation Coefficient . . . 30
  - 3.2.5 Combinations and Interpretations . . . 31
- 3.3 Visualization . . . 33

4 Methodology **35**
- 4.1 Censoring . . . 39
  - 4.1.1 Approximation of Survival Time . . . 39
  - 4.1.2 Weighting Censoring . . . 41
- 4.2 Regression . . . 42
  - 4.2.1 Regression Algorithms . . . 43
  - 4.2.2 Regression Algorithms for Censored data . . . 50
  - 4.2.3 Gating Regression . . . 52
- 4.3 Grouping . . . 55
  - 4.3.1 Classification and Regression Tree . . . 55
  - 4.3.2 Clustering . . . 60
- 4.4 Outliers . . . 64
  - 4.4.1 Mahalanobis Distance . . . 65
  - 4.4.2 Minimum Covariance Determinant Estimator . . . 66
- 4.5 Preprocessing . . . 70
  - 4.5.1 Feature Representation . . . 70
  - 4.5.2 Feature Selection . . . 71
  - 4.5.3 Log-Space Transformation . . . 73

5 Experiments **74**
5.1 Experimental Setups . . . 75
5.1.1 Data Set . . . 75
5.1.2 Methods . . . 75
5.1.3 Evaluation . . . 76
5.2 Experimental Results . . . 77
5.2.1 Baseline . . . 77
5.2.2 Handcrafted Tree . . . 78
5.2.3 Learning Algorithms . . . 79
5.2.4 Handling Censored Data . . . 80
5.2.5 Grouping . . . 81
5.2.6 Combination of CART and Regressions . . . 82
5.2.7 Outliers Detection . . . 82
5.2.8 Log-space Transformation . . . 83
5.2.9 Classification . . . 85
5.3 Discussion . . . 86
5.3.1 Processing Features . . . 87
5.3.2 Segregating Patients . . . 88
5.3.3 Learning Predictors . . . 89
5.3.4 Handling Censoring . . . 89

6 Conclusion **91**

Bibliography **93**

A Data Set **96**
A.1 Attribute Descriptions . . . 97
A.2 Histogram . . . 99

B Detailed Experimental Results **100**
B.1 Experiment Methods . . . 101
B.2 Detailed Experimental Results . . . 102
B.3 Best Results . . . 112

List of Tables

2.1 The formulations and notations of survival data . . . 10
2.2 An example of the feature set . . . 10
2.3 An example of a data set . . . 11
2.4 An example of using the *Kaplan-Meier estimator* . . . 11
2.5 An example of result from the *Logistic Regression* . . . 15
2.6 An example of using the *concordance index* . . . 18
2.7 The definition of *confusion matrix* . . . 18
2.8 An example of the *confusion matrix* . . . 19

3.1 An example of using the *L1*, the *L2*, and the *relative absolute error* . . . 29
3.2 Examples of extreme cases in calculating the *relative absolute error* . . . 29
3.3 An example of using *concordance index* . . . 30

4.1 An example of approximating event times by adding a constant . . . 40
4.2 An example of approximating event times by averaging over uncensored patients in the *risk set* . . . 40
4.3 An example of approximating event times by averaging over the *risk set* . . . 41
4.4 An example of using candidate error functions . . . 59
4.5 An example of experimental results with and without outliers . . . 65
4.6 An example of using the *Mahalanobis estimator* . . . 66
4.7 An example of using the *minimum covariance determinant estimator* . . . 67
4.8 An example of pre-processed and post-processed survival data . . . 70

5.1 Experimental results on the *baseline* . . . 78
5.2 Experimental results on a handcrafted tree using the *Kaplan-Meier estimator* . . . 79
5.3 Experimental results on conventional learning algorithms . . . 79
5.4 Experimental results on handling censored data . . . 81
5.5 Experimental results on grouping methods . . . 82
5.6 Experimental results on combining *CART* with *log-rank statistics* and regressions . . . 83
5.7 Experimental results on regression methods, after eliminating 3% outliers . . . 83
5.8 Experimental results on regression methods in logarithmic space . . . 84
5.9 P-values of pair-t test . . . 84
5.10 Experimental results on classification . . . 86
5.11 Top 10 models . . . 87
5.12 Top 5 models with the lowest $L1_{p<12}$ . . . 88

A.1 Here is the caption in block . . . 97
A.2 Here is the caption in block . . . 98
A.3 Histograms of features . . . 99

B.1 A List of methods for predicting survival times . . . 101
B.2 Here is the caption in block . . . 102
B.3 Here is the caption in block . . . 105
B.4 Here is the caption in block . . . 110
B.5 Top 10 models . . . 112
B.7 Here is the caption in block . . . 113
B.6 Visualizations of predicted survival time versus actual survival time of the final model 118
B.8 Here is the caption in block . . . 119

List of Figures

2.1 An example of a *Kaplan-Meier curve* extracted from [42] 12
2.2 An example of visualizing the performance of a prognostic factor 16
2.3 An example of *ROC curves* and *AUC* extracted from [16] 21
2.4 An example of a *probabilistic graphical model* 23

3.1 An example of *3-fold cross-validation* extracted from [11] 27
3.2 Visualizations of two predictors with the same *concordance index* but different *average relative absolute error* 31
3.3 Visualizations of two predictors with the same *average relative absolute error* but different *concordance index* 32
3.4 An illustration of 95% *confidence interval* 33
3.5 A visualization of actual survival times versus estimated survival times 34

4.1 A framework of survival predictions 38
4.2 A visualization of the *support vector regression* with soft margin, extracted from [55]. 44
4.3 An example of tree splitting 47
4.4 An example of using the *regression trees* 50
4.5 A visualization of the loss function of the *SVRc* (extracted from [28]) 52
4.6 The *gating regression* algorithm 53
4.7 An example of using the *log-rank statistics* as the splitting criterion 57
4.8 An example of a mixture model 61
4.9 An illustration of *EM Clustering* extracted from [31] 62
4.10 An example of *EM Clustering* 63
4.11 A visualization of an outlier in a 2-dimensional space 64
4.12 A visualization of an outlier in a 2-dimensional space 67

5.1 A Visualization of true survival times versus predicted survival times 78
5.2 A Visualization of true survival times versus predicted survival times 80
5.3 Examples of decision trees split by the *CART* with the *log-rank statistics* as the splitting criterion 82
5.4 A Visualization of true survival times versus predicted survival times 85

Chapter 1

Introduction

1.1 The Problem

Imagine if you are diagnosed with a terminal illness. Here, it is clearly useful to know how much time you have left to spend with your loving families and friends.

Survival Prediction is the task of predicting the length of time that a patient will survive. This task is difficult in general due to the complicated relationship between genetic, biological, and environmental factors in the human body. Furthermore, no one knows which features affect survivability.

Most research on this topic has focused on *population-based* studies, which try to discover the prognostic factors and/or analyze the probability distributions over the survival times of different groups of patients. This problem has been studied within the field of statistics for decades, and there exist some standard methodologies to estimate survival distributions, such as the Kaplan-Meier estimator, the Cox Proportional Hazard Model, and the logistic regression model. (Details can be found in Section 2.3.) While these models are powerful, they are not designed to make predictions for *individuals*. Knowing only that a patient belongs to a specified distribution, the best we can do is return the mean/median survival time. However, as this analysis applies to a large group, it usually has huge variance (e.g., 19 months $\pm$ 30 months). Recalling the question we asked above, we would prefer a more detailed response – i.e., one with tighter variance.

This motivates our work, which has this objective: to find the most accurate model for predicting the survival times for each *individual* patient. We view this as a regression problem, where we try to map the features for each patient to his/her survival time. As the relationship between features and survival time is still not understood, we consider various ways to *learn* these models from existing historical patients. This is challenging in medical/clinical data due to the presence of irrelevant features, outliers, and missing class labels. In this thesis, we describe our approach and framework for handling these, and other challenges, producing techniques that can group patients, and regression methods for predicting survival times.

1.2 Motivation

Thousands of people suffer or die from cancers each day. Medical doctors still cannot make accurate prognosis since the relationship between health conditions and survivability is still unknown. When patients consult doctors regarding their survival time, the doctors make predictions based on their medical knowledge and previous observations. On the other hand, past experiences are not always reliable – e. g., prognoses from different doctors are often inconsistent [7]. A system that can use medical observations to produce accurate survival times has immediate application in real world. We list several below.

Most cancer patients have to decide whether they want to receive treatments, such as chemotherapy or radiation therapy. However, these treatments are invasive and carry the potential for severe side effects, indeed, many patients die from the therapies rather than the cancer itself [18]. Patients

with mild outlooks may not need to undergo aggressive treatments. Patients with severe outlooks may find it more prudent to focus their energies on getting their affairs in order and spending time with their families rather than subjecting themselves to painful treatments. In all cases, accurate prognosis can give doctors better guidelines on selecting treatments and planning futures. Providing patients with an accurate prognosis allows them to make informed choices about whether to accept painful or risky treatments.

There are also other ways that an effective prognostic system will be able to improve the quality of health care. The relationship between health conditions and survivability is still unknown. If we can show which features are more dominant in determining the survival time, it could help the medical doctors understand why certain phenomenon happens.

1.3 Challenges

Survival prediction is challenging for several reasons. This is difficult in general due to the complicated relationship between genetic, biological, and environmental factors in the human body, especially as it is not clear which variable affects survivability. For each patient, we can collect many factors from the medical records, such as his/her age, gender, blood test results, weight loss, etc. However, irrelevant features solely contribute more noise to the data and degrade the quality of the regression model.

Also, working with survival data is challenging as the information is often incomplete. This is especially problematic when the class labels (survival times) are missing, because supervised learning algorithms rely on these labels for training a predictor. The survival data that we analyze in this work contains 2402 patients, including 1260 censored — i.e., patients whose survival time is unknown. (Details of missing class labels can be found in Section 2.1.) However, we cannot simply eliminate these 1260 patients from our experiment since they contribute important information .

The third challenge in processing medical data is the the presence of *outliers*— that is, patients who are extremely different from the rest of the populations. Among all stage IV lung cancer patients in our data set, the median survival time is 11.20 ± 12.89 months, but 4 out of 389 patients have survived more than 5 years (more than 3 standard deviations away from the mean). Additionally, patients could die from unexpected external factors that are not correlated to their diseases. Imagine a group of patients, whose medical conditions are similar, normally survive more than 2 years, but one patient passed away from a car accident 1 week after his/her diagnosis. We certainly cannot anticipate this patient's death because it is caused by an accident. Outliers often exert problematic influence on the parameters and should to be excluded before training the model.

Another challenge is that patients are heterogeneous, with different survival patterns for different subgroups of patients. For instance, the variable "cigarette smoking" may be important for lung cancer patient but perhaps not as critical as for colon-rectum cancer patients. In order to overcome this non-linear relations between features and survival time, we attempt to model each risk group

separately. Unfortunately, it is not known which group of patients share the same survival pattern. Therefore, in this thesis, a major task is to design an appropriate method that can segregate patients with different survival distributions.

1.4 Contributions

My thesis claim is that we are able to learn a model from historical patient data that can effectively predict survival times for novel patients. We build this model from a data set of patients' historical records, including personal attributes, diagnostic assessments, and blood test results. We develop a framework for processing censored information, segregating populations, and predicting survival time for each individual patients. We define the measurements to evaluate the quality of survival predictions, an assessment that verifies that a model built upon a combination of machine learning and statistical methodologies can make decent survival predictions.

We consider survival prediction as a regression problem and base our solution on a combination of unsupervised and supervised learning. One issue we need to address is the missing class labels in the training samples; here, we propose several methodologies to approximate the class labels and a weighting strategy to lower the significance of these incomplete instances. Our approach to this problem involves two phases, the *learning* phase and the *performance* phase. The learning phase has two steps. In the first step, we apply various grouping methods to segregate patients into smaller populations. In the second step, we apply different regression models to each sub-group we obtained from the first step. Then, we pick the most accurate combination as our final model. In the performance phase, we can predict a specific value for each novel patient by using the final model we produce from the learning phase.

1.5 Outline

We briefly provide some background information in the next chapter. Section 2.1 introduces the basic concept and terminologies of survival analysis. Section 2.2 describes the definition and the formulation of survival data. Section 2.3 shows several classical models and evaluation methods that have been used in *survival analysis*. We will discuss how this differs from our *survival prediction* task. Although there is no standard methodology to predict survival, Section 2.4 will briefly review some historical and related work in prognosis using statistical models, artificial intelligence, and machine learning.

In Chapter 3, we describe the validation and evaluation methods of our framework. We describe the notion of cross-validation in Section 3.1. The evaluation methods for uncensored and censored data are explained in Section 3.2.

Chapter 4 explains our framework and methodologies in detail. Section 4.1 introduces our ideas of handling censoring observations. We propose several techniques to approximate survival times for

censored patients, and we discuss the details in Section 4.1.1. Our attempts to weight the censored observations are described in Section 4.1.2.

In Section 4.2, we review the methodologies and background theories of regression models. In Section 4.2.1, we describe the general ideas of the *linear regression*, the *support vector regression*, and the *regression trees*. In Section 4.2.2, we illustrate some variations of regression models that are customized to handle censored observations. As mentioned above, our approach first divides patients into smaller population; in the last section, we describe the ideas of automatically select a good regressor for each population.

Section 4.3 describes our methodologies for grouping patients, which are based on two primary methods, tree and clustering. Section 4.3.1 describes our ideas of applying the *classification and regression trees* to discriminate patients, and Section 4.3.2 describes the *clustering* methods. Details of splitting, pruning, and parameters setup can be found in this chapter.

In Section 4.4, we discuss some issues of outliers and our methods on eliminating them from the data set. Section 4.4.1 provides a detailed explanation on *Mahalanobis distance*, and Section 4.4.2 provides information on applying the *minimum covariance estimator* in outlier detection.

Section 4.5 describes our methods for manipulating data. As our data is a mixture of discrete and continuous variables, Section 4.5.1 explains our reasons and methods for expanding the discrete features. In Section 4.5.2, we describe ways to eliminate irrelevant features using *subset selection by backward wrapper* and *variable ranking by mutual information*. In Section 4.5.3, we discuss the technique of log-space transformation.

Chapter 5 summarizes the experiment setup and outcomes. The result of censored data handling, outlier detections, grouping methods, and regression algorithms can be found in Section 5.2, and detailed results are provided in Appendix B. Finally, we conclude our work in Chapter 6.

Chapter 2

Background

In this chapter, we briefly describe the basic concept and related work on survival analysis. Section 2.1 explains some frequently used terminologies and equations on medical prognosis and survival analysis. Section 2.2 describes the definition and the formulation of survival data. Section 2.3 discusses common research methods in survival analysis, and Section 2.4 reviews some relevant work in prognosis using survival analysis, artificial intelligence, and machine learning approaches.

2.1 Survival Analysis

The goal of survival analysis is to examine and model the time of an event of interest, which has been broadly applied in medicine, reliability study, financial insurance, etc. For instance, in clinical research, some typical events are death or recurrence of a phenomenon. Notice that survival analysis is different from survival prediction in that survival analysis models the *probability distribution* of this phenomenon within a population, which is typically used to discover the *prognostic factors* (i.e., factors that affect survivability) or estimate the *median survival time* of this population. In contrast, the goal of survival prediction is to accurately predict the *remaining time* to this phenomenon for each **individual** within a population.

Let T be a non-negative random variable that represents survival time. In medical prognosis, for instance, T generally refers to how much longer a patient will remain alive, after the diagnosis of the disease. The probability of T can be specified in three ways: the survival function, the density function, and the hazard function. Here is a list of terminologies and frequently used functions.

The *survival function* is the probability that an event will occur later than a specified time t. The survival function is a non-increasing function, which is defined as

$$S(t) = Pr(T > t) = 1 - Pr(T \leq t) \tag{2.1}$$

$$\text{where } S(0) = 1 \text{ and } S(\infty) = 0$$

The cumulative density function of T is defined as

$$F(t) = Pr(T \leq t) = 1 - S(t)$$

By definition, the probability density function of T can be calculated by

$$f(t) = \frac{dF(t)}{dt} = -\frac{dS(t)}{dt}$$

The hazard function of T measures the instantaneous decline in survival at $T = t$, given that the patient has survived until t.

$$h(t) = Pr(t \leq T < t + dt | T \geq t) = \frac{f(t)}{S(t)}$$

The cumulative hazard function is related to the survival function by

$$H(t) = \int_0^t h(u)du = -\ln S(t)$$

$$S(t) = e^{-H(t)} \tag{2.2}$$

After acquiring the survival function of a population, many medical researchers are interested in looking at the *median survival time*. This median survival time divides the samples into two equal halves, so that half the patients die before this time and the other half survive longer than this time. Some follow a variant: take the median survival time within the *95% confidence interval*— i.e., after eliminating the most extreme five-percent of patients [39].

2.1.1 Medical Prognosis

Medical prognosis studies the challenge of estimating the recurrence of disease and predicting survivability of patients [15]. Much medical and clinical reasoning takes the *hypothetico-deductive* method as a procedure in scientific research. A hypothesis is a statement or explanation about some medical observations that is either true or false, and the hypothetical-deductive approach is the procedure of testing the hypothesis to deduce a consequence of the explanation. For example, consider the assumption that females have longer survival time than males among lung cancer patients. To verify this assumption, experiments are carried out to test the difference on survival distributions of females and males. The result is usually verified with some statistical methods; if the statistical result is significant, then this hypothesis is supported.

A *randomized clinical trial* (RCT) is a common method to verify a hypothesis through experiment. This process randomly chooses patients belonging to various predefined groups (e.g., men vs. Women or men under 25 vs. Women over 50) to receive treatments. We record the relevant characteristics of each patient, such as age, gender, etc. After completing the experiments (often taking years), researchers ask whether there is a statistical difference among different groups; such result may provide useful information in designing procedures for the treatment and the development of drugs [40].

As described above, studies of medical prognosis normally begin with one or more assumptions and involve performing experiments to support some given hypotheses. However, it is usually not proven that these assumptions are relevant and unbiased. Besides, it is relatively harder to analyze high dimensional data set with hypothesis (e.g., consider micro-array data that includes thousands of features). For these reasons, our approach is to develop models solely from a data set of patients' records without any prior grouping in our work.

2.2 Survival Data

In this section, we will introduce the notion of censoring and our formulation of survival data. We will provide a simple example of survival data, which will be used to demonstrate examples through out this paper.

2.2.1 Censored Observations

A major difference between survival data and a standard machine learning data is that some patients may not have event-times, that is, we do not know when these patients experience the event. Such omissions are their label — that is called *censoring* [10]. Type I censoring occurs when the patients are still alive when the data is collected, so we only know the time of their last visit. Censoring can also happen when a patient failed to follow up after a certain time, and this is called Type II censoring. In either case, the censored value is a lower bound on the patient's actual survival time.

In survival analysis, censored data can provide valuable information even though the actual event time is unknown [51]. Missing survival times is dissimilar to missing values in the features because the censored time provides the lower bound on the actual survival time. Simply eliminating censored observations or treating them as uncensored samples (i.e., pretend that the censored time is the time of the actual event) would bias the predictor and so should be avoided. For instance, consider a group of patients that were studied over a fixed period of 3 months, and imagine that 10% of patients died within 3 months while the remaining 90% survived at the end of study. If we simply omit all censored observations, the remaining data would suggest that all patients will die within this fixed period of 3 months.

Given the difficulties of obtaining and collecting information from patients' records, the number of patients in the data set is often insufficient. Although the actual survival times of censored data are unknown, Shivaswamy et al. suggests that we should utilize a censored observation when there is complete information [51]. For example, assume that our goal is to predict whether a patient can survive more than 2 years versus less than 2 years. It is correct to include all censored patients who survive at least 2 years since we know that they belong to the categories of long survivors. (If a censored patient whose censor time is shorter than 2 years, we choose to discard this sample since we are not sure whether or not this patient can survive longer than 2 years.)

2.2.2 Formulation of Survival Data

Given a set of survival data $D = (X, T)$, where X represents the feature values (characteristics of patients) and T represents the class label (survival time or censored time), our goal is to predict a survival time P for each patient. Table 2.1 is a list of symbols/variables that we use through out this paper.

Table 2.2 and Table 2.3 shows an example of survival data. Throughout this paper, we will use this imaginary data set to demonstrate examples of different models and algorithms. Table 2.2 lists some features of this example data, which are either nominal (e.g., Yes/No, Censored/Uncensored, etc) or numerical (i.e., real numbers). Table 2.3 is an imaginary data set of 6 patients.

Symbols	Description
D	Survival data set
n	Number of instances (patients)
m	Number of features
A	m-dimensional vector of features in D
X	$n \times m$ feature matrix of D
X_i	m-dimensional vector of feature values of the i^{th} patient in D
X^j	n-dimensional vector of feature values of the j^{th} feature of n patients in D
x_i^j	Feature value of the j^{th} feature of the i^{th} patient in D
T	n-dimensional vector of class labels (survival time or censored time) of D
t_i	Class label of the i^{th} patient in D and $t_i \in \mathbb{R}$
C	n-dimensional vector of censored flags of n patients in D
c_i	Censored flag of the i^{th} patient in D and $c_i \in \{0, 1\}$
P	n-dimensional vector of predicted survival times of n patients in D
p_i	Predicted survival time for the i^{th} patient in D and $p_i \in \mathbb{R}$
R_i	Risk set at time t_i such that $R_i = \{\text{Patient}_j \in D \vert t_j \geq t_i\}$

Table 2.1: The formulations and notations of survival data

Feature Name	Variable Name	Data Type	Range
Patient Name	Patient_i	String	Unique Identifier
Gender	X^{gender}	Nominal	0: Male, 1: Female
Age	X^{age}	Numerical	
Lung Cancer	X^{lung}	Nominal	0: No, 1: Yes
Pancreas Cancer	$X^{pancreas}$	Nominal	0: No, 1: Yes
Performance Status	X^{ps}	Nominal	1, 2, 3, 4
Albumin	$X^{albumin}$	Numerical	
$\vdots$	$\vdots$	$\vdots$	$\vdots$
Time (Event or Censored)	t_i	Numerical	
Censored Flag	c_i	Nominal	0: Uncensored, 1: Censored

Table 2.2: An example of the feature set

2.3 Modelling Survival Distribution

Survival analysis traditionally focuses on analyzing prognostic factor and/or modelling the survival distribution of a population [52]. Here is an imaginary scenario: we would like to know if cigarette smoking is a good prognostic factor for lung cancer. We study the populations of lung cancer patients with versus without cigarette smoking, and we try to prove or disprove that people who smoke and have lung cancer are more likely to die earlier than those who do not. There are several well-established methods on modelling such *survival distribution*, which we briefly describe them in the following section. We conclude this section by relating these approaches to our goal.

2.3.1 Kaplan-Meier Estimator

Kaplan-Meier analysis [27] is one of the most widely used tools for analyzing survival distributions of different populations. The basic idea of Kaplan-Meier is to examine the proportion of events that occur at each distinct time point. Assuming there are r distinct event times, $(t_1, t_2, ..., t_r)$, and R_i is

Patient$_i$	X^{gender}	X^{age}	X^{lung}	$X^{pancreas}$	X^{ps}	$X^{albumin}$	...	t_i	c_i
Patient$_0$	0	80	0	1	1	33.5	...	1	0
Patient$_1$	1	70	0	1	3	30.7	...	8	0
Patient$_2$	1	65	1	0	4	25.2	...	10	1
Patient$_3$	0	25	1	0	1	50.3	...	13	0
Patient$_4$	1	65	0	1	2	27.4	...	18	1
Patient$_5$	1	60	0	1	1	26.7	...	120	0

Table 2.3: An example of a data set

the *risk set* , which is the set of patients who are still alive at time t_i, let

$$n_i = |R_i| \text{ be the size of the risk set at time } t_i$$

$$d_i \text{ be the number of events that occur at time } t_i$$

$$\alpha_i = P(T > t_i | T > t_{i-1}) = \frac{n_i - d_i}{n_i}$$

More precisely, α_i is the probability of survival in $[t_{i-1}, t_i)$ given that a person has survived until at least t_{i-1}. The Kaplan-Meier estimator of the probability of survival at time t_i is calculated by:

$$S(t_i) = S(t_{i-1}) \times \alpha_i = \prod_{t_j < t_i} \alpha_j$$

For the first 5 patients in our imaginary data set in Table 2.3, there are 3 events in this data set, at time t =1, 8, and 13. (Censored observations are not considered as events.) At each distinct event time t, we can calculate the number of events, the size of the risk set, and the probability of survival, as shown in Table 2.4. Figure 2.1 plots the probability distribution of the survival function we calculated in the second table.

t_i	n_i	d_i	α_i	$S(t_i)$
0	5	0	5 / 5	1.0
1	5	1	4 / 5	1.0 $\times$ 4/5 = 0.8
8	4	1	3 / 4	0.8 $\times$ 3/4 = 0.6
13	2	1	1 / 2	0.6 $\times$ 1/2 = 0.3

Table 2.4: An example of using the *Kaplan-Meier estimator*

Kaplan-Meier is one of the standard methods in survival analysis. One advantage is that this model effectively incorporates censored observations since the estimator only requires the information on the size of the risk set and number of events at each distinct time. Notice that the final product of Kaplan-Meier estimator is a survival *distribution* not a survival *time*, which is different from our primary goal. Since this model is one of the most common tool in survival analysis, we will later try this for our prediction task: after placing a patient into a particular Kaplan-Meier distribution, we will then use the mean/median survival time from this KM distribution as a predicted survival time for that patient.

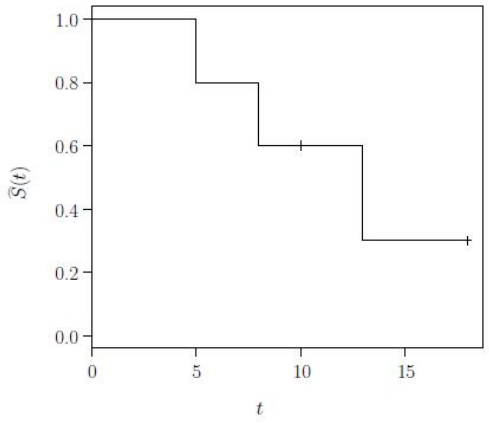


Figure 2.1: An example of a *Kaplan-Meier curve* extracted from [42]

2.3.2 Cox Proportional Hazard Model

Cox Proportional Hazard Model [9] is a multivariate regression method in survival analysis that incorporates the ideas of the Kaplan-Meier estimator and the effect of features. A *proportional hazard* model assumes that the hazard (see Section 2.1 for definition) between different risk groups are proportional, and all features are time independent. This model consists of (1) a function that models the effect of features, and (2) a *baseline hazard*, an arbitrary and unspecified function that models the risk over time. In general, any hazard function can be written as

$$h(t|X) = h_0(t)g(X)$$

where $h_0(t)$ is the baseline hazard function

X is a vector of features

$g(X)$ is a function that models the effect of features

Since each instance is proportional to the baseline, the *hazard ratio* (HR), or the ratio between a risk group and the baseline, can be calculated by:

$$HR = \frac{h(t|X)}{h_0(t)} = g(X)$$

One of the benefits of the proportional model is that we can leave the baseline function unspecified if we are interested in the the hazard ratio between the two groups.

Cox proportional hazard model is a type of proportional hazard model, which assumes that the effect of features is an exponential function of a linear combination of the features. More precisely, this model defines $g(X) = e^{\beta(X-X_0)}$ where β is a vector of parameters for the feature and X_0 is a vector of features of the baseline. The hazard function of a Cox proportional hazard model can be

written as:

$$h(t|X) = h_0(t)e^{\beta(X-X_0)}$$

We will demonstrate how to estimate the hazard function of a patient using the same imaginary scenario from the last section (see Table 2.3). Assuming that for each patient, we are only given his/her age $= X_i^{age}$, gender $= X_i^{gender}$ (0 indicates Male and 1 indicates Female), and event time t_i. Let's use Patient$_0$ as the baseline in this example and assume that we know the hazard function of Patient$_0$ — that is, the baseline hazard function $h_0(t)$, which models the instantaneous risk of Patient$_0$ over time, as previously described in Section 2.1. (Recall that the baseline hazard function is an arbitrary and unspecified function. In this example, our aim is to show the process of calculating the hazard function of a patient from the baseline hazard function, and so we will leave this function unspecified.) Assuming that we had fit a Cox model and obtained two parameters: $\beta^{age} = 0.01$ for the age feature and $\beta^{gender} = 1$ for the gender feature, we can estimate the hazard function of Patient$_1$ by

$$\begin{aligned} h_1(t) &= h_0(t)e^{[\beta^{age}(X_1^{age}-X_0^{age})+\beta^{gender}(X_1^{gender}-X_0^{gender})]} \\ &= h_0(t)e^{[(0.01)(70-80)+(1)(1-0)]} \\ &\approx h_0(t) \times 2.4596 \end{aligned}$$

The above result shows that Patient$_1$ is approximately two to three times more risky (i.e., more likely to have an event) than Patient$_0$. Notice that it does not matter which patient was selected as the baseline, since the hazard function only measures the ratios between the patients. It is common to choose the average of each feature μ_X as the baseline and then normalize the features of each patient. Letting $X' = X - \mu_X$ be a vector of normalized features, we can rewrite the hazard function of the Cox model as:

$$h(t|X) = h_0(t)e^{\beta X'}$$

The parameter β can be estimated using partial likelihood maximization, and the baseline hazard function can be estimated by Kalbfleisch-Prentice Estimator [25]. Detail of calculations can be found in the work of [25] and is omitted here. Once we have the baseline hazard function, we can estimate the baseline survival function $S_0(t)$ (by Equation 2.2). Then, the survival function for each individual patient is

$$S(t) = S_0(t)^{e^{\beta X'}}$$

Similar to the KM estimator, the Cox proportional hazard model is one of the standard methodologies in survival analysis, but it is not designed to produce survival time. In our work, we attempt to evaluate the effectiveness of Cox model for individual prediction by using the median survival time generated from a Cox model as the predicted survival time for each patient.

2.3.3 Logistic Regression

Logistic Regression is a popular methodology for predicting the posterior probabilities of an individual belonging to some classes, when there are two or more classes [35]. In survival analysis, if we are only concerned with the likelihood of an event, we can apply logistic regression to estimate the probability of this event. For example, to predict whether or not a patient will experience a recurrence, the logistic regression is employed to estimate the probability of "recurrence" (positive class) versus "no recurrence" (negative class). Let

X be a vector of features of the patient

β be the vector of coefficient for positive class

β^0 be the intercept or noise variable for positive class

$C = Positive$ denotes that the patient belongs to the positive class

$C = Negative$ denotes that the patient belongs to the negative class

The probability that the patient belongs to the positive class and the negative class are estimated by:

$$Pr(C = Positive|X = X_i) = \frac{1}{1 + e^{\beta^0 + \beta x}}$$
$$Pr(C = Negative|X = X_i) = 1 - Pr(C = Positive|X = X_i)$$

Suppose now we are interested to know the likelihood of 1 year survival of each patient in the data set from Table 2.3. Without loss of generality, we can define $Long\_Survival$ (survive more than 1 year) as the positive class and $Short\_Survival$ as the negative class (survive less than 1 year). Imagine that we fit a logistic regression model and obtain three parameters $b^0 = -0.5$, $b^{age} = 0.01$, and $b^{gender} = 1$. The process of estimating the probability that Patient$_0$ belongs to the longer survival group versus the shorter survival group is shown below. Table 2.5 shows the predicted probabilities for the first five patients in the data set in Table 2.3.

$$\begin{aligned} Pr(Long\_Survival|X = X_0) &= \frac{1}{1 + e^{[\beta^0 + \beta^{age} X_0^{age} + \beta^{gender} X_0^{gender}]}} \\ &= \frac{1}{1 + e^{[(-0.5) + (0.01)(80) + (1)(0)]}} \\ &= \frac{1}{1 + e^{0.1}} \\ &\approx 0.43 \\ Pr(Short\_Survival|X = X_0) &= 1 - P(Long\_Survival|X = X_0) \\ &\approx 1 - 0.43 \\ &\approx 0.57 \end{aligned}$$

Patient	Time (Month)	Event	$Pr(Long\_Survival)$	$Pr(Short\_Survival)$
Patient_0	1	Death	0.43	0.53
Patient_1	8	Death	0.23	0.77
Patient_2	10	Censor	0.24	0.76
Patient_3	13	Death	0.56	0.44
Patient_4	18	Censor	0.24	0.76
Patient_5	120	Death	0.15	0.85

Table 2.5: An example of result from the *Logistic Regression*

If there are more than two classes, we need to estimate the posterior probability that a patient belongs to *each* class. Let

X be a vector of features of the patient

β_k be the vector of coefficient for class k

β_k^0 be the intercept or noise variable for class k

$C = k$ denotes that the patient belongs to class k

$$Pr(C = k|X = X_i) = \frac{e^{\beta_k^0 + \beta_k X_i}}{1 + \sum_{l=1}^{K-1} e^{\beta_l^0 + \beta_l X_i}}, \text{ for } k = 1, .., K-1$$
$$Pr(C = K|X = X_i) = \frac{1}{1 + \sum_{l=1}^{K-1} e^{\beta_l^0 + \beta_l X_i}}$$

Logistic Regression is another dominant method in survival analysis; unfortunately, this model is not sufficient for predicting individual survival times. Logistic Regression produce the *probability* that a patient belongs to a class but not the *remaining time* of survival. Again, although it is different from our primary objective, we will test its effectiveness in classifying patients into long survivors and short survivors.

2.3.4 Evaluation and Validation

Much research on survival analysis focuses on finding the prognostic factors that can segregate patients and/or produce a survival distribution for a population. The effectiveness of a prognostic factor and the predictability of a survival function is usually evaluated via statistical methods and/or graphical visualization. We discuss some general methods here.

Evaluating Prognostic Factor

How well a feature can discriminate populations is usually measured by some statistical test such as the *p-value* from the *log-rank* statistics, which uses an hypothesis-test to compare the difference between two survival distributions. For instance, to determine whether gender is an effective feature on discriminating patients, one could test the difference between the survival distributions of men and women. If their survival distributions are significantly different, gender is considered an effective prognostic factor.

Here is the basic procedure of log-rank statistical test. Suppose we are given two populations, G_1 and G_2. For each patient, we are given his/her survival time (if the patient is already dead) or censored time (if the patient is still alive or has left the study prior to death). This statistic will measure the likelihood that G_1 and G_2 are identical (the null hypothesis).

Let t_i be distinct event times in either group

d_{1i} and d_{2i} be the number of observed events of group 1 and group 2 at t_i

$d_i = d_{1i} + d_{2i}$ be the number of observed events at t_i

n_{1i} and n_{2i} be size of the risk set of group 1 and group 2 at t_i

$n_i = n_{1i} + n_{2i}$ be size of the risk set at t_i

$E_{ij} = d_i \dfrac{d_{1i}}{d_{1i} + d_{2i}}$ be the expected value of group i

$V_j = \dfrac{d_i(n_{1j}/n_i)(1 - n_{1j}/n_i)(n_i - d_i)}{n_j - 1}$ be the empirical variance

The *Z score* of the log-rank statistic is calculated by

$$Z = \frac{\sum_{i=1}^{j} d_{1i} E_{1j}}{\sqrt{\sum_{i=1}^{j} V_j}}$$

Finally, the *p-value* is obtained by applying the chi-square statistic to the Z score shown above. A large *p-value* (close to 1) indicates that these two populations are similar, while smaller *p-value* (close to 0) suggests they are not. Typically, a *p-value* < 0.05 is considered significant. That is, this value allows us to claim that the null hypothesis does not hold here — i.e., the two populations are not drawn from the same distribution.

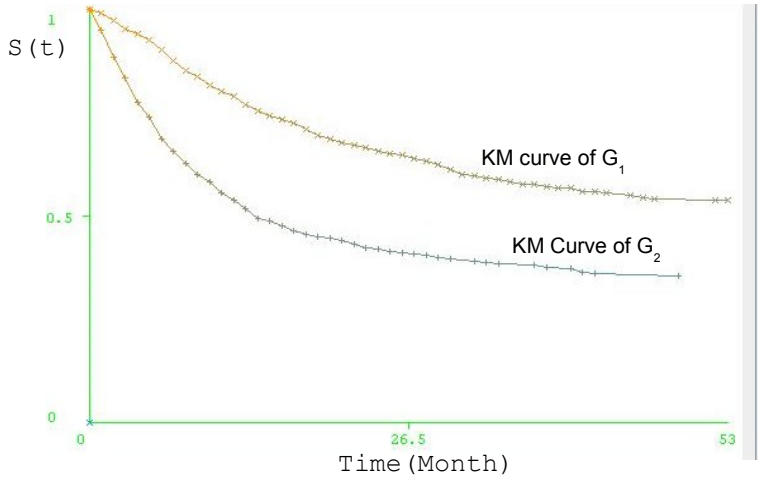


Figure 2.2: An example of visualizing the performance of a prognostic factor

Another typical way to evaluate the performance of a prognostic factor is to visualize at the Kaplan-Meier curves of the different sub-populations (differing only in the value of this single fac-

tor) in the same plot [27]. For example, Figure 2.2 shows the Kaplan-Meier curves of two risk groups such that these two groups do not have the same survival distributions.

The p-value and the graphical visualization are two common tools in evaluating the effectiveness of prognostic factor. Notice this is different with our primary interest of predicting individual survival times. However, the *p-value* of the log-rank statistic can help us quantify the difference between two groups of patients. In our work, we use this measurement to evaluate how well a feature can segregate patients into small sub-populations. (See Section 4.3.1 for more details on how *p-value* is used when splitting a classification and regression tree.)

Evaluating Predictability

The *concordance index(CI)* is a standard metric for evaluating the correctness of a survival distribution by measuring the proportion of all comparable patient pairs in which the predicted and actual survival time are ranked in the correct order [20]. Imagine that our goal is to predict whether or not a patient can survival until a specific time t (e.g., 1 year, 2 years, 5 years), and our model predicts the probability that a patient will live longer than t. For all possible pairs of patients i and j, let:

t_i, t_j be the survival time of patient i, j

p_i, p_j be the predicted probability of survival until a specific time t for patient i, j

A pair i and j is "comparable" if we can determine whether $t_i > t_j$ or vice versa. (If Patient$_i$ is censored at t_i and $t_i < t_j$, we cannot compare the survival time of these two patients since the actual survival time of Patient$_i$ could be longer than t_j.) For all comparable pairs of i and j, the relationship between them is defined (disjointly) as:

$$\textit{Concordant} \text{ if } \begin{cases} t_i > t_j \text{ and } p_i > p_j \text{ or} \\ t_i < t_j \text{ and } p_i < p_j \end{cases}$$

$$\text{If not } \textit{Concordant} \text{ , then } \textit{Tie} \text{ if } \begin{cases} t_i > t_j \text{ and } p_i = p_j \text{ or} \\ t_i < t_j \text{ and } p_i = p_j \end{cases}$$

$$\text{If not } \textit{Concordant} \text{ nor } \textit{Tie} \text{ , then } \textit{Discordant} \text{ if } \begin{cases} t_i > t_j \text{ and } p_i < p_j \text{ or} \\ t_i < t_j \text{ and } p_i > p_j \end{cases}$$

The *concordance index* is then calculated by:

$$CI = \frac{\text{number of concordance} + 0.5 \times \text{number of ties}}{\text{number of comparable pairs}}$$

Table 2.6 shows the result from Logistic Regression model (Table 2.5) and a few examples of determining concordant relationship between pairs of patients. In order to calculate the *concordance index* , we need know the number of concordant pairs among all pairs of comparable patients. Notice that Patient$_2$ and Patient$_3$ are not comparable since Patient$_2$ is censored and may survive longer than Patient$_3$, and we cannot determine their concordant relationship. Here, of the 15 pairs of patients, 11 pairs are comparable.

Patient	Time (Month)	Event	$Pr(Long\_Survival)$
Patient_0	1	Death	0.43
Patient_1	8	Death	0.23
Patient_2	10	Censor	0.24
Patient_3	13	Death	0.56
Patient_4	18	Censor	0.24
Patient_5	120	Death	0.15

$\Downarrow$

Patient_i	Patient_j	t_i	t_j	p_i	p_j	Result
Patient_0	Patient_1	1	8	0.43	0.23	Discordant
Patient_0	Patient_2	1	>10	0.43	0.24	Discordant
Patient_0	Patient_3	1	13	0.43	0.56	Concordant
Patient_0	Patient_4	1	>18	0.43	0.24	Discordant
Patient_0	Patient_5	1	120	0.43	0.15	Discordant
Patient_1	Patient_2	8	>10	0.23	0.24	Concordant
Patient_1	Patient_3	8	13	0.23	0.56	Concordant
$\vdots$						

Table 2.6: An example of using the *concordance index*

The *CI* is then the ratio between number of concordant pairs to the number of all possible pairs. This number range from 0 to 1, with 0.5 indicates no correlation, and 1 indicates perfect prediction. A difference of 0.02 or larger is typically considered to be significant. In the above example, the *CI* is $\frac{11}{15} \approx 0.73$.

In survival analysis, the Concordance Index is typically applied to evaluate models that produce probability distributions (such as the proportional hazard models) or probability of different survival classes (such as Logistic Regression models). In our work, we define Concordance Index differently than the example above since our models produce survival times instead of survival probabilities. We will reintroduce this idea again in Section 3.2.3.

Besides *concordance index* , it is also typical to measure *accuracy*, *sensitivity*, *specificity*. *receiver operating characteristic curve* (ROC) and/or the *area under a receiver operating characteristic curve* (AUC). Assume that the actual outcomes and the predicted outcomes are either positive or negative (e.g. survive longer than 1 years), we can visualize the relationships from a *confusion matrix*, which is defined in Table 2.7.

		Actual Outcome	
		Positive	Negative
Predicted Outcome	Positive	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)
		Sensitivity	Specificity

Table 2.7: The definition of *confusion matrix*

Table 2.8 shows an example of predicted results from Logistic Regression model and confusion matrix from these predictions. Recall that $Long\_Survival$ represents the group of patients who survive longer than 1 year (the positive class) and $Short\_Survival$ represents the group of patients who survive shorter than 1 year (the negative class). $Pr(Long\_Survival)$ is the predicted probability that a patient belongs to the positive class (i.e., survive longer than 1 year) and $Pr(Short\_Survival)$ is the predicted probability that the patient belongs to the negative class (i.e., survive less than 1 year). If $Pr(Long\_Survival) > Pr(Long\_Survival)$, the patient is more likely to be in the positive class, or the *predicted class* for this patient is $Long\_Survival$. From the results of predictions in the first table, we can determine each component of a confusion matrix as shown in the second table.

Patient$_i$	$Pr(Long\_Survival)$	$Pr(Short\_Survival)$	Actual Class	Predicted Class
Patient$_0$	0.43	0.57	$Short\_Survival$	$Short\_Survival$
Patient$_1$	0.23	0.77	$Short\_Survival$	$Short\_Survival$
Patient$_2$	0.24	0.76	$Short\_Survival$	$Short\_Survival$
Patient$_3$	0.56	0.44	$Long\_Survival$	$Long\_Survival$
Patient$_4$	0.24	0.76	$Long\_Survival$	$Short\_Survival$
Patient$_5$	0.15	0.85	$Long\_Survival$	$Short\_Survival$

$\Downarrow$

		Actual Outcome	
		Positive	Negative
Predicted Outcome	Positive	TP: 1	FP: 0
	Negative	FN: 2	TN: 3
		Sensitivity	Specificity

Table 2.8: An example of the *confusion matrix*

The most naive measurement is the *accuracy*, which is the percentage of correct predictions. *Sensitivity* measures the percentage of true positive that are correctly predicted as positive, and *specificity* measures the percentage of negatives that are correctly predicted as negative, that is, use:

$$\textit{accuracy} = \frac{\text{TP + TN}}{\text{TP+TN+FP+FN}}$$
$$\textit{sensitivity} = \frac{\text{TP}}{\text{TP + FN}}$$
$$\textit{specificity} = \frac{\text{TN}}{\text{TN + FP}}$$

From the same example of confusion matrix above, we can calculate:

$$\textit{accuracy} = \frac{1 + 3}{1+3+0+2} \approx 0.67$$
$$\textit{sensitivity} = \frac{1}{1 + 2} \approx 0.33$$
$$\textit{specificity} = \frac{3}{3 + 0} = 1$$

If the class labels are real numbers (e.g., in our case, the survival times), the regression results can be estimated by determining a threshold value to create a classification boundary between two

classes. (In the example above, the classification boundary is 1 year.) Assuming that t is the actual output and p is the predicted output, where both t and p are real number, we can define a threshold value c, so that all values larger than c belong to one class and all values less than c belong to another class. *sensitivity* and *specificity* can be calculated by:

$$\textit{sensitivity} = Pr(p > c | t > c)$$
$$\textit{specificity} = Pr(p < c | t < c)$$

While these measurement are often adapted for model assessment, they are problematic if the classes are imbalanced (i.e., the majority belong to one class) which is often the case in medical data [36]. An alternative method is to inspect the *receiver operating characteristic* (ROC) [16], which measures classifier performance over the whole range of possible frequencies and tradeoff. A ROC curve has two dimension: the *sensitivity* on the x-coordinate and (*1-specificity*) on the y-coordinate. One point in an ROC dominates another if it is above and left to another (higher true positive and lower false positive). The ROC curves display the relationship of predictions and outcomes by plotting the estimates of *sensitivity* versus (*1-specificity*) for all possible threshold values. In the previous example, we used 1 year as the classification boundary. For this particular time-threshold, the classification results are *sensitivity* $= 0.5$ and *1-specificity*$= 0$, which yields a single point $(0.5, 0)$ in a two-dimensional graph.

We can define multiple classification boundaries (e.g., 1 months, 6 months, 1 year, 2 years, etc.) and repeat the same calculation for each boundary. The results will be multiple pairs of (*sensitivity*, *1-specificity*), or multiple ROC points. Plotting these points on the same figure and connecting them together will give us a ROC curve. Figure 2.3 illustrates an example of ROC curves from two classifiers A and B.

Another way to evaluate a classifier or regressor is to estimate the area under the ROC curve (AUC), which reduces multiple ROC points to a single scalar number. From Figure 2.3, we can also visualize that Classifier B has larger area under its ROC curve, and therefore is a better classifier.

Survival analysis tells us the probability distribution of survival times, from which we can compute the average or the median survival time. However, this statistic is usually with large variance since it is an average estimation for a population. Recalling our motivating example, we want to be able to anticipate how long an individual patient will survive. The methods we reviewed in this section are powerful in producing and evaluating survival distributions; nevertheless, these methods are not precise enough to predict an accurate *survival time* for *individuals*. Hence, in this work, we have to consider alternative methods for our purpose.

2.4 Related Work

While survival analysis is primarily studied by statisticians and bio-statisticians, many computer scientists have attempted to utilize artificial intelligence and machine learning techniques for medical

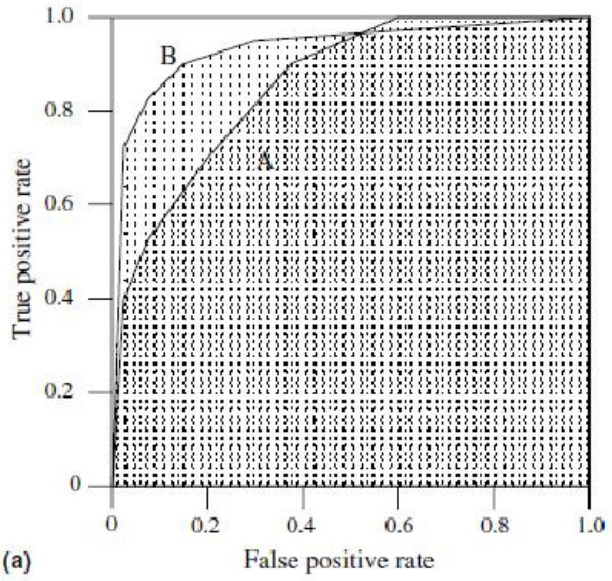


Figure 2.3: An example of *ROC curves* and *AUC* extracted from [16]

diagnosis and prognosis. Early applications of artificial intelligence in medicine focuses on modelling the knowledge of experts, but today, much research adapts machine learning algorithms to solve binary or multi-class classification problems. In this section, we review some relevant work in prognosis using survival analysis, artificial intelligence, or machine learning approach.

2.4.1 Previous Work in Survival Analysis

A large number of research projects are working on discovering the prognostic factors or determining the relationship between features and survival. Seve et al. use uni-variate Cox proportional hazard model to determine the significance of each variable to overall survival, and found that low serum albumin levels and liver metastasis (both prognostic factor have *p-value* < 0.0001) are powerful prognostic markers for carcinomas cancer patients [49]. But notice that knowing the relevant prognostic markers is not sufficient to predict individual survival times.

Predicting the probability of survival is another focal point in survival analysis. One example is the commercial APACHE III [29], a proprietary database and decision support system that estimates the probability of death in intensive care units. Luaces et al. also predict the probability of survival in intensive care unit but with different approach: the authors apply support vector machine where the model is built by optimizing the area under the ROC curve [33]. Notice that this method produces the *probability* (that the patient will survive in ICU), which is different from our goal of predicting the remaining *time* of survival.

Beer et al. use gene-expression to identify whether a patient with lung adenocarcinomas belongs to the low risk group or the high risk group [1]. This work use uni-variate analysis to find the

prognostic factor that can best identify risk groups. Log-rank statistics was applied to quantify the differences among these groups, and the top 50 genes correctly identified low and the high risk group within the hold-out set ($p = 0.024$). Again, this is not sufficient to predict individual survival times.

Another research subject in survival analysis is to develop scoring system for prognosis of specific diseases or events. For example, the Model for End-Stage Liver Disease (MELD) is used to determine the priority of liver transplant candidates [26]. This system predicts the mortality in patients with end-stage liver disease and quantified the assessment as a numerical score from 6 to 40. The quality of this scoring system is evaluated using 95% *concordance index* , and the best model achieved 0.87% concordance. Again, developing a scoring system is not our objective, but *concordance index* will be applied in our evaluation criteria.

2.4.2 Previous Research in Artificial Intelligence

Since the 1960s, several artificial intelligence projects involved in medical diagnosis or prognosis, most focusing on modelling large scale domain-specific knowledge [48]. The first successful rule-based expert system, MYCIN, was designed to identify bacteria and recommend antibiotics for treatments. The system was a simple inference engine with a knowledge base of if-then rules. After posing a sequence of yes/no questions, it then provided a list of plausible diagnoses. However, the development of this type of expert system fell out of favour due to difficulty in translating medical knowledge into logical statements as required by such rule-based models.

In the early 1980s, probability and decision theory were introduced to the AI community, lead by Pearl who started the Bayesian networks to encode the inherent uncertainty [38]. This probabilistic approach resolved the difficulties of rule-based system and became the dominant models of representing uncertainty in medical expert system [22]. Figure 2.4 shows a simple example of a probabilistic graphical model. It is known that asthma can cause cough and wheeze, and flu can cause fever and cough. Given the presence of fever, cough, and wheeze, the model can derive the probability of flu and probability of asthma. One downside of this approach is the challenge in constructing a correct structure since the causality between features and outcomes is still unknown. Even though the structure and parameters can be acquired using machine learning methods, the large number of features often makes the training process, and hence this approach, ineffective.

2.4.3 Previous Work in Machine Learning

Data-driven approach plays a major role in medical diagnosis/prognosis research today [6]. One of the advantages of machine learning approach is that most algorithms could build a model using only raw data – i.e., without prior medical or biological knowledge. Various machine learning techniques have been used in *classifying* patients into groups of different survival times; for example, predicting whether or not a patient will survive more than two year or less than two years. We briefly review some representative work here.

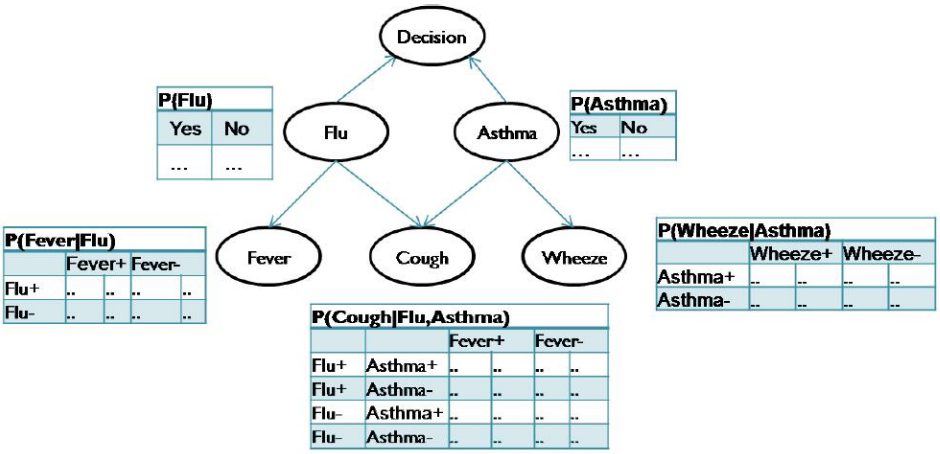


Figure 2.4: An example of a *probabilistic graphical model*

Delen et al. [12] try to predict whether a breast cancer patient will still be alive after 5 years from the date of diagnosis. Artificial neural network, decision trees, and logistic regression were applied in this project, and the performances were evaluated by their *accuracy*, *sensitivity*, and *specificity*. Their experiments show that decision tree outperforms other models with a classification *accuracy* of 93.6%. Decision tree seems like a potential solution, but this algorithm predicts a survival class rather then a survival time. In our work, we apply regression trees, a variation of decision trees, which predicts a real number for each individual; see Section 4.2.1.

Jayasurya et al. [23] employ Bayesian network and support vector machine for two-year survival prediction in lung cancer patients. Their results show that Bayesian network outperformed support vector machine with $AUC = 0.77$ versus $AUC = 0.71$ respectively. These two models are originally designed for classification purpose, which is different with our primary goal. This work has shown the effectiveness of these two algorithms; hence, we will evaluate their performance on classifying patients using the median survival time as the classification boundary.

Other applications including variations of neural networks. For example, Dybowski et al. [14] predict systemic inflammatory response syndrome and show that neural network is more accurate than logistic regression ($AUC = 0.863$ vs $AUC = 0.753$). One advantage of neural network is the ability to model nonlinear relationship, but it is often challenging to train such a model. Again, we will test the effectiveness of Neural Network for classification in our experiments.

Lee1 et al. [30] also try to predict a category for each breast cancer patient, but their work categorizes patients into three risk groups: good, intermediate, and poor. The goal of their work is to find *a linear combination of features* that can segregate the populations. (Notice that their work is different from conventional survival analysis, which tests how well a single feature can discriminate different populations.) The authors consider this task as a *multi-class classification* problem, that is, classifying patients into more than two categories. Their approach is to apply Support Vector

Machine (SVM) with Gaussian Kernel to classify patients into these three categories. Experiments are performed on a set of 253 breast cancer patients, and the performance of this model is evaluated by calculating the p-values of log-rank statistics between all pairs of categories. In their experiments, (a linear combination of) all prognostic factors have *p-value* < 0.05, and therefore these separations are significant. We applies SVM for both regression and classification, and we provide basic concept and our approach of SVM in Section 4.2.1.

Aside from predicting how long a patient can survival, medical doctors are interested in predicting the recurrence of a disease or cancer. This is crucial because the physicians can use this prediction to decide whether to perform a surgery on a patient or not. Zupan et al. consider this problem as a *binary classification* task, and they attempt to predict the probability that a patient will have recurrence within 7 years [57]. Unfortunately, the difficulty of applying machine learning to this task was due to the large amount of censoring in their testing data. (Whether the prostate cancer recurs after 7 years was unknown for 73% of the patients because the 7 years period had not been reached.) The authors used a weighted example technique to handle censored data and applied different statistical and classification models to find the probability of recurrence. They claimed that Naive Bayes and the Cox model perform better than other predictors by evaluating their *accuracy*, *sensitivity*, *specificity*, and *concordance index*. In our work, we consider similar type of weighted method to handle censored observations (see Section 4.1.2 for more details).

There are several machine learning projects that segregate patients via classification and regression trees. To determine recurrence of breast cancer patients, Ture et al. incorporate a decision tree with Kaplan-Meier estimator to segregate patients into small categories [54]. Experiment results suggest that C4.5 performs better than other models by evaluating their *sensitivity*, *specificity*, etc. We use a similar approach in this work, and we describe our solution in detail in Section 4.3.1.

Chapter 3

Evaluation

This chapter summarizes our methods on validations and evaluation. Section 3.1 discusses the reasons of incorporating verification methods and the *k-fold cross-validation*. We followed the standard machine learning approach of evaluating the quality of a predictor by the average loss over the set of independent testing cases, but we found it challenging to define the term "loss" in predicting survival times. In Section 3.2, we will explain why conventional approaches are not sufficient in survival predictions and how our models are evaluated with alternative measurements for uncensored and censored observations.

3.1 Cross-Validation

As described in section 2.3.4, much research on survival analysis has focused on finding prognostic factors of some specific events for a population. In our work, our goal is to make predictions for *new* patients — i.e., it is more crucial that our models perform well on *novel* data that was not used for training the model.

In machine learning, we typically evaluate a predictor by measuring its performance on a holdout set— that is, a set of data that is not used for training the predictor. We will use a learning algorithm $L(.)$ to build a predictor $f = L(D)$ using a given training data set D. To evaluate the quality of that predictor f, D is partitioned into a training set $D_{training}$ and a testing set $D_{testing}$, such that $D_{training}$ is for building the predictor $f' = L(D_{training})$ and $D_{testing}$ is for evaluating this learned f'. (The purpose of f' is to rate the effectiveness of $L(.)$; recall the actual f is based on all of D.) How D is divided into $D_{training}$ and $D_{testing}$ will affect the results of assessment. It is common to incorporate a *k-fold cross-validation* to obtain the best estimate of the effectiveness of the predictor [41].

Given a data set D, a learning algorithm $L(.)$, and a constant k (e.g., $k = 5$ or $k = 10$ are common choices for k), the process of the *k-fold cross-validation* is as follows:

Algorithm 1 The *k-fold cross-validation* algorithm

Input:

D: a data set
$L(.)$: a learning algorithm
k: a constant that specifies the number of divisions for cross-validation

Output:

$Err_{CV}(L, D)$: the average cross-validation error of the algorithm $L(.)$ on the data set D

1: Randomly divide D into k disjoint subsets of equal size $D = \cup\{D_1, D_2, ..., D_k\}$
2: **for** $i = 1$ to k **do**
3: $\quad D_{testing} \leftarrow D_i$
4: $\quad D_{training} \leftarrow D - D_i$
5: $\quad f_i = L(D_{training})$ be the predictor based on $D_{training}$
6: $\quad E_i \leftarrow$ evaluation result of f_i based on $D_{testing}$
7: $Err_{CV}(L, D) \leftarrow \frac{1}{k} \sum_i^k E_i$
8: **return** $Err_{CV}(L, D)$

The *k-fold cross-validation* effectively uses all data for both training and testing, without reusing

any instance for testing. After k iteration (with different D_i in each iteration), each D_i is used for both training ($k-1$ times) and testing (once). We evaluate the performance of each f_i and the average performance from these f_i (see the next section for evaluation methods).

Figure 3.1 illustrates an example of a 3-fold cross-validation. Suppose that we have a set of patients who are either in class A or class B. We begin by randomly dividing our data into 3 sets: the red set, the blue set, and the grey set. As shown in the figure, the rightmost classifier is trained using the red set and the blue set, and it is evaluated using the grey set. We repeat the same process 3 times (with different testing set), so we have 3 predictors and 3 evaluations. Finally, we can estimate the average results from these 3 experiments.

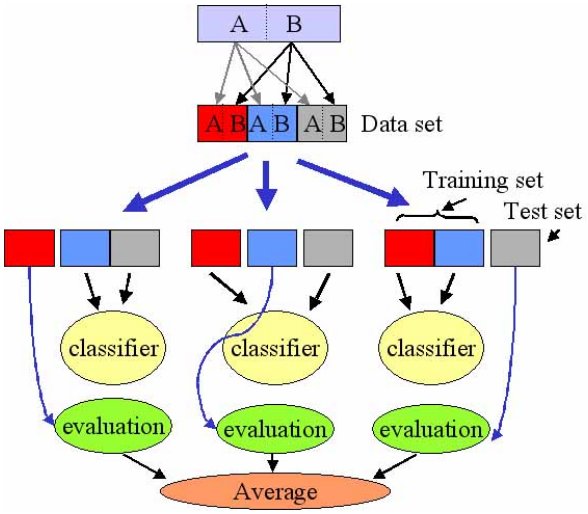


Figure 3.1: An example of *3-fold cross-validation* extracted from [11]

3.2 Evaluating Predictions

Ideally, the regression models should give a prediction p_i for each patient Patient$_i$ that is a reasonable approximation of the patient's true survival time t_i. In the following section, we will discuss our evaluation methods and how to interpret them in our experiments.

3.2.1 L1 and L2 Error

In machine learning community, typical methods for evaluating regression models include:

- The *average mean absolute error*, or the *average L1 error*, measures the magnitude of difference between the true survival time and the estimated survival time.

$$average\ L1\ error = \frac{1}{n}\sum_{i=1}^{n}|t_i - p_i|$$

- The *average root-mean-squared error*, or the *average L2 error*, measures the average root squared errors between actual survival time and the predicted survival time.

$$average\ L2\ error = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(t_i - p_i)^2}$$

Table 3.1 shows an example using the *average L1 error* and the *average L2 error* on our imaginary data in Table 2.3. Notice that censored patients are not included in this example since we do not know their actual survival times and so we cannot evaluate their predictions using L1 nor L2.

3.2.2 Relative Absolute Error

However, the performance of a predictor cannot be fairly evaluated using *L1* or *L2* error. From the example in Table 3.1, the true survival time of Patient$_0$ is 1 month, and the model predicts 6 months. Consider another case, Patient$_5$, whose true survival time is 10 years, and the model predicts 10 years + 5 months. The absolute L1 error (resp., squared L2 error) for Patient$_0$ and Patient$_5$ are both 5 months (resp. 25 months^2) respectively; while this is a rather good prediction for Patient$_5$, it is less adequate for Patient$_0$.

For this reason, we consider evaluating our model using the *average relative absolute error (RAE)*; more precisely, for each patient, we measure the difference between the true survival time t_i and the predicted survival time p_i over the magnitude of the prediction p_i. The *RAE* of the i^{th} patient is calculated by:

$$RAE_i = \frac{|t_i - p_i|}{p_i}$$

The *average relative absolute error* is then calculated by:

$$RAE = \frac{1}{n}RAE_i = \frac{1}{n}\sum_{i=1}^{n}\frac{|t_i - p_i|}{p_i}$$

Table 3.1 shows an example of different evaluation method on the same data set in Table 2.3. We predict that Patient$_5$ will survive 10 years + 5 months, which differs from Patient$_5$'s true survival time by a factor of 4%. Although Patient$_5$ and Patient$_0$ have the same L1 and L2 error, the *relative absolute error* for Patient$_0$ was 83%, while the *relative absolute error* for Patient$_5$ was 4%. The *relative absolute error* is consistent with our assessment that the prediction for Patient$_5$ is better than the prediction for Patient$_0$.

Notice that we have chosen the absolute error *relative to the predicted survival time* but not *relative to the true survival time* since the prediction is known at the time of prognosis while the actual survival time is not. We want to estimate, in average, how much our predictions may be off from the patient's actual survival time, and we consider this quantity will provide a useful information for our patients. For instance, in the example in Table 3.1, the *average relative absolute error* of our predictor is 0.5. Suppose we have a new patient, Patient$_6$, whose predicted survival time is 8

Patient	Actual Survival (Month)	Predicted Survival (Month)	L1	L2	RAE
Patient_0	1	6	5	25	0.83
Patient_1	8	4	4	16	1.00
Patient_3	13	15	2	4	0.13
Patient_5	120	125	5	25	0.04
Average	35.50	37.75	4.00	≈ 4.18	0.5

Table 3.1: An example of using the *L1*, the *L2*, and the *relative absolute error*

months. We can inform Patient_6 that he/she will probably survive 8 months, but this prediction may be different from his/her true survival time by 4 months (i.e., 50% of 8 months).

Notice that the *relative absolute error* is not a completely fair measurement. If $p_i \gg t_i$, then $RAE_i \approx 1$. In contrast, if $p_i \ll t_i$, then RAE_i will be relatively large since the absolute error is divided by a small number. Consider the example in Table 3.2. Here, as we predict Patient_0 will survive 11 month, but the actual survival time is 1 month, the *relative absolute error* for Patient_0 is 10. We predict that Patient_1 will survive 100000 months while the actual survival time is 8 months, the *relative absolute error* for Patient_1 is approximately 1. However, we consider the prediction for Patient_0 is better since the prediction for Patient_1 is unreasonable and unrealistic.

Patient	Actual Survival (Month)	Predicted Survival (Month)	RAE
Patient_0	1	11	10
Patient_1	8	100000	≈ 1

Table 3.2: Examples of extreme cases in calculating the *relative absolute error*

Although there is no standard definition on what a "bad" *RAE* is, in our work, we consider limiting the *RAE* of our models under 1. For instance, if the *RAE* of a predictor is higher than one, we do not consider using this predictor.

3.2.3 Concordance Index

The *Concordance Index (CI)*, described in Section 2.3.4, is one of the most common evaluation methods for survival models; it measures the portion of all pairs of patients whose *predicted survival probability* are correctly ordered among all "comparable" pairs of patients. In this work, we also use the *CI* to evaluate our predictors, but we measure the portion of correctly ordered *predicted survival times* instead.

$$\text{Let } t_i, t_j \text{ be the actual survival times of patient } i, j$$

$$p_i, p_j \text{ be the predicted } \textit{survival times} \text{ of patient } i, j$$

Recall that a pair i and j is "comparable" if we can determine whether $t_i > t_j$ or vise versa. (If Patient_i is censored at t_i and $t_i < t_j$, we cannot compare the survival time of these two patients

since the actual survival time of Patient$_i$ could be longer than t_j.) For all comparable pairs of i and j, the relationship between them is defined (disjointly) as:

$$\textit{Concordant} \text{ if } \begin{cases} t_i > t_j \text{ and } p_i > p_j \text{ or} \\ t_i < t_j \text{ and } p_i < p_j \end{cases}$$

$$\text{If not } \textit{Concordant}\text{, then } \textit{Tie} \text{ if } \begin{cases} t_i > t_j \text{ and } p_j - p_i < 1 \text{ or} \\ t_i < t_j \text{ and } p_i - p_j < 1 \end{cases}$$

If not *Concordant* nor *Tie*, then *Discordant*

Notice that the p_i and p_j above are different from the p_i and p_j in Section 2.3.4, as here, our regressors are generating *survival times* instead of *survival probabilities* (e.g., the results from Logistic Regression or Cox proportional hazard models). Recall that the *concordance index* is calculated by:

$$\textit{CI} = \frac{\text{number of concordant pairs} + 0.5 \times \text{number of ties}}{\text{number of comparable pairs}}$$

Similar to the previous definition, the *concordance index* ranges from 0 to 1; a value of 1 indicates perfect ordering, a value of 0.5 indicates that the predictor is not better than random guessing, and a value of 0 indicates complete disagreement.

Table 3.3 shows an example of estimating *CI* from our data in Table 2.3 and imaginary predicted survival times. In this example, 6 out of 11 pairs of comparable patients are concordant, so the *CI* in this example is 0.55.

Patient	Actual Survival	Censored Time	Predicted Survival
Patient$_0$	1	-	10
Patient$_1$	8	-	3
Patient$_2$	-	10	5
Patient$_3$	13	-	15
Patient$_4$	-	18	9
Patient$_5$	120	-	125

$\Downarrow$

Patient$_i$	Patient$_j$	t_i	t_j	p_i	p_j	Result
Patient$_0$	Patient$_1$	1	8	10	3	Discordant
Patient$_0$	Patient$_2$	1	>10	10	5	Discordant
Patient$_0$	Patient$_3$	1	13	10	15	Concordant
Patient$_0$	Patient$_4$	1	>18	10	9	Discordant
Patient$_0$	Patient$_5$	1	120	10	125	Concordant
Patient$_1$	Patient$_2$	8	>10	3	5	Concordant
Patient$_1$	Patient$_3$	8	13	3	15	Concordant
$\vdots$						

Table 3.3: An example of using *concordance index*

3.2.4 Correlation Coefficient

Another useful measurement between predictions and survival times is the *Pearson correlation coefficient*, which measures the covariance of T and P divided by the product of their standard devia-

tions.

$$\begin{aligned} \text{Let } Cov(T,P) &= \text{ covariance of } T \text{ and } P \\ \mu_T, \mu_P &= \text{ mean of } T \text{ and } P \\ \sigma_T, \sigma_P &= \text{ standard deviation of } T \text{ and } P \end{aligned}$$

The correlation coefficient is calculated by

$$Corr(T,P) = \frac{Cov(T,P)}{\sigma_T \times \sigma_P} = \frac{E[(T-\mu_T)(P-\mu_P)])}{\sigma_T, \sigma_P}$$

One issue in calculating correlation coefficient is that T is unknown for censored data, and therefore, we calculate the *correlation coefficient* on **uncensored** patients only. In general, we anticipate that an efficient predictor (i.e., with low *RAE* and high *CI*) should have high *correlation coefficient* as well. Although we do not use *correlation coefficient* as our main criteria in evaluating predictor, we consider this measurement as an extra criteria to assess our predictors.

3.2.5 Combinations and Interpretations

The *average relative absolute error* and the *concordance index* have their own strengthes and weaknesses, and we consider interpreting the combination of these two measurements. Imagine two predictors f_A and f_B such that their survival predictions rank patients in the same order, yet f_B over-estimates survival times for ALL patients. These two predictors will have the same *concordance index* but f_A will have smaller *RAE*. Figure 3.2 shows the visualization of f_A in the left figure and f_B in the right figure.

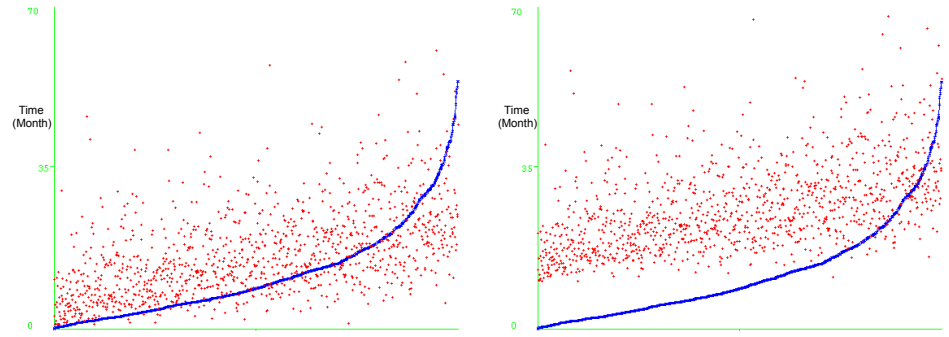


Figure 3.2: Visualizations of two predictors with the same *concordance index* but different *average relative absolute error*

Considering another situation where f_A and f_B have the same *RAE*, but f_B tends to over-estimate survival times for short survivors and under-estimate survival times for long survivors (i.e., consider f_B makes the same prediction for each individual patient). We prefer f_A over the f_B since f_A is better in descriminating short survivors versus long survivors. Figure 3.3 shows the visualization of f_A in the left figure and f_B in the right figure. For example, imagine a patient whose actual survival time is 6 months, f_A predicts 4 months and f_B predicts 12 months. The *RAE* for these

two predictors are both 0.5, but we prefer under-estimate rather than over-estimate. The reason is that, if a patient is dying, we would rather give the patient a warning ahead of time. Image another patient whose actual survival time is 24 months, f_A predicts 48 months and f_B predicts 12 months. The *RAE* for these two predictors are both 0.5 again, but f_A can predict that this patient will survive longer.

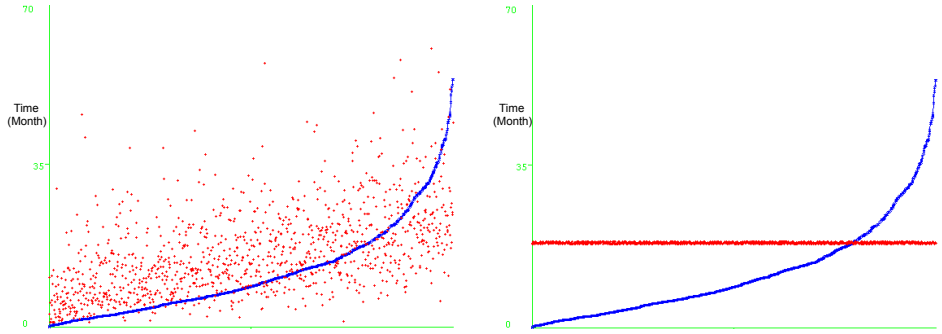


Figure 3.3: Visualizations of two predictors with the same *average relative absolute error* but different *concordance index*

In our work, we define the best model as the one that achieves the minimum *average relative absolute error*. If the *RAE* of two models are equivalent or the difference between them is not statistically significant, we will seek the model that is higher in its *concordance index*. (Although *CI* does not provide more information for individual patients, we will select the predictor that is more capable in ranking patients.)

Besides the *RAE* and the *CI*, we also examine the RAE^{95}, which measures the *relative absolute error* on the testing cases that lie within the 95% *confidence interval*. Recall that survival prediction problems suffer from the presence of outliers. It would be nicer if we knew which ones were outliers, but it is generally difficult. Our studies evaluate all patients in the unseen data set. We found, however, that a few outliers can skew our statistical analysis. In order to determine whether a predictor is significantly affected by the outliers, we consider evaluating the subset of "presumably normal" patients. It is a common practice in survival analysis to look at statistics within the 95% *confidence interval* [39]; here, we are "borrowing" their ideas — i.e., assuming that 5% are potential outliers and evaluating the performance of our predictors on the remaining 95% cases. Figure 3.4 illustrates the idea of eliminating the 5% of worst cases. That is, we make predictions for all patients, but our evaluation removes the worst 5% of them (the 5% with the highest *relative absolute error*).

For example, assuming that we have 40 patients in our testing data. After making predictions for each patient, we divide the testing data into the subset of under-estimated predictions ($p_i < t_i$) versus the subset of over-estimated predictions ($p_i > t_i$). Then, we sort the patients within each subset by their *relative absolute error*. For each subset, we will remove $40 \times 0.25 = 1$ patient.

How else can we evaluate a predictor? We can look at different variations of evaluation methods that we discussed in this section. In Section 3.2.2, we mentioned that the *average relative*

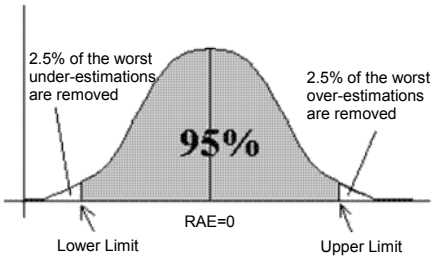


Figure 3.4: An illustration of 95% *confidence interval*

absolute error is not totally fair since it exerts large penalty for smaller predictions (see Table 3.2 for examples). Therefore, we consider evaluating the results of smaller predictions separately. In our experiment, we also estimate the *average L1 error* for predictions less than 12 months and the *average relative absolute error* for predictions greater or equal to 12 months.

Here is the list of statistics that we consider in evaluating survival predictors.

- Primary criteria (used to comparing different models)
  - *RAE*: the *average relative absolute error*
  - *CI*: the *concordance index*
- Secondary criteria
  - RAE^{95}: the *average relative absolute error* within the 95% *confidence interval*
  - $L1_{p<12}$: the *average L1 error* of predictions less than 12 months
  - $RAE_{p\geq 12}$: the *average relative absolute error* of predictions greater than or equal to 12 months
- Other interesting measurements
  - *CC*: the *correlation coefficient*
  - *Accuracy*: the *accuracy* of classification
  - *Sensitivity*: the *sensitivity* of classification
  - *Specificity*: the *specificity* of classification

3.3 Visualization

Another evaluation method is through visualization of some plots. For instance, Figure 3.5 shows a plot of true survival times and the predictions over a set of test cases. The blue line indicates the true survival times and the red points are the predicted survival times. The points will lie on the survival curve if the predictions are perfect. Our idea is that a reasonable approximation should predict a survival time for each patient that is not too far away from the patient's true survival time. Notice that this plot requires the true survival times (not available for censored patients), and therefore, we only plot the results of **uncensored** patients.

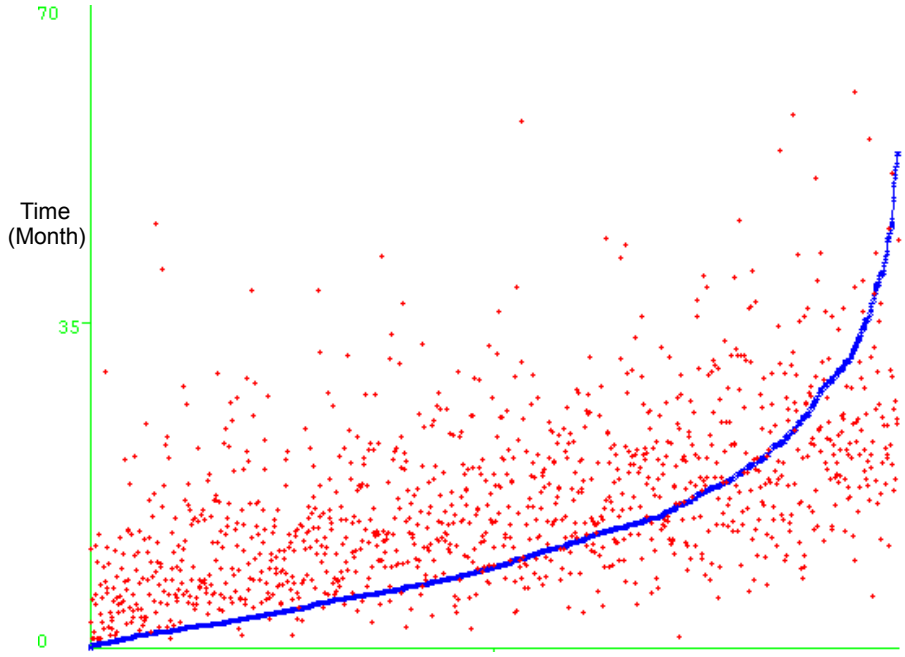


Figure 3.5: A visualization of actual survival times versus estimated survival times

Chapter 4

Methodology

In this work, our goal is to learn a model from historical patient data that can effectively predict survival times for novel patients. We build this model from a database of historical records of n patients $D = (X, T)$ where $X = \{X_1, X_2, ..., X_n\}$ represents the feature values of these n patients, including personal attributes, diagnostic assessments, and blood test results, and $T = [t_1, t_2, ..., t_n]$ represents the actual survival times or censored times of these n patients. Then, for each new patient with feature values X_{new}, we use the resulting model to predict a survival time p_{new} for this patient. We consider the task of survival prediction as a regression problem and base our solution on a combination of unsupervised and supervised learning.

Given a set of historical data D, our proposed framework of predicting survival times for individual patients is sketched in Figure 4.1. Our approach to this problem involves two phases, the learning phase and the performance phase. The learning phase has two primary steps. In the first step, we apply various grouping methods to find a set of partition rules R that segregates D into k smaller populations $D = \cup\{D_1, D_2, ..., D_k\}$. In the second step, we learn a set of predictors $F = \{f_1, f_2, ..., f_k\}$ from different regression methods — one for each sub-group such that predictor f_l is for the sub-population D_l. Then, we pick the most accurate combination of a grouping method and a regression algorithm as our final model. Algorithm 2 summarizes the procedure of the learning phase. Our methodologies include:

- Processing Features (Section 4.5): Medical data involves many features of the patients. We need to appropriately represent patients' information and identify the most characterizing features.

- Segregating Patients (Section 4.3): Different groups of patients may have different survival patterns. We try to segregate patients into smaller groups such that the differences between patients within the same group are minimized and the differences between patients across different groups are maximized.

- Eliminating Outliers (Section 4.4): Here, outliers are patients who are extremely different from the majority of patients, so outliers often exert problematic influence on the parameters of the learned models. We therefore attempt to eliminate them from our data.

- Handling Censoring (Section 4.1): supervised learning algorithms rely on class labels for training a predictor, but the class label of a censored patient is not the actual survival time of this patient but a lower bound of the actual survival time (see Section 2.2.1 for information about *censoring*). In order to use supervised learning algorithms, we propose several techniques to handle censored observations.

- Learning predictors (Section 4.2): As the relationship between features and survival time is still not understood, we consider various regression methods to learn a predictor from existing historical patients.

Algorithm 2 Learning a Survival Prediction Model

Input:

D: a set of patients' historical records

Output:

R: a set of partition rules that segregate patients into k sub-populations

$F = \{f_1, f_2, ..., f_k\}$: a set of k predictors for each sub-population

1: Pre-process D
2: Learn a set of partition rules R
3: Partition D into $D = \cup\{D_1, D_2, ..., D_k\}$ according to R
4: **for all** D_l in D **do**
5: $D'_l \leftarrow D_l -$ outliers
6: $D''_l \leftarrow D'_l$ with treated censored data
7: Learn a predictor f_l from D''_l
8: **return** R, F

In the performance phase, we can predict a specific value for each unseen patient with feature values X_{new} by using the final model (i.e., a set of partition rules R for grouping patients into sub-populations and a set of predictors F, one for each sub-population) we obtained from the learning phase. We can assign this new patient into one of the sub-population in D according to the set of partition rules R. Then, if this patient is not an outlier (i.e., this patient is not too different from the majority of patient in the same sub-population), we can predict a survival time $p_{new} = f_l(X_{new})$. Our procedure of the performance phase is summarized in Algorithm 3.

Algorithm 3 Using a Survival Prediction Model

Input:

X_{new}: feature values of a new patient

R: a set of partition rules that segregate patients into k sub-populations

F: a set of k predictors for each sub-population

Output:

p_{new}: the predicted survival time or -1 if unpredictable

1: Pre-process X_{new}
2: $l \leftarrow$ sub-population assignment, according to R
3: **if** X_{new} is an outlier of D_l **then**
4: **return** -1 – unpredictable
5: **else**
6: $p_{new} \leftarrow f_l(X_{new})$
7: **return** p_{new}

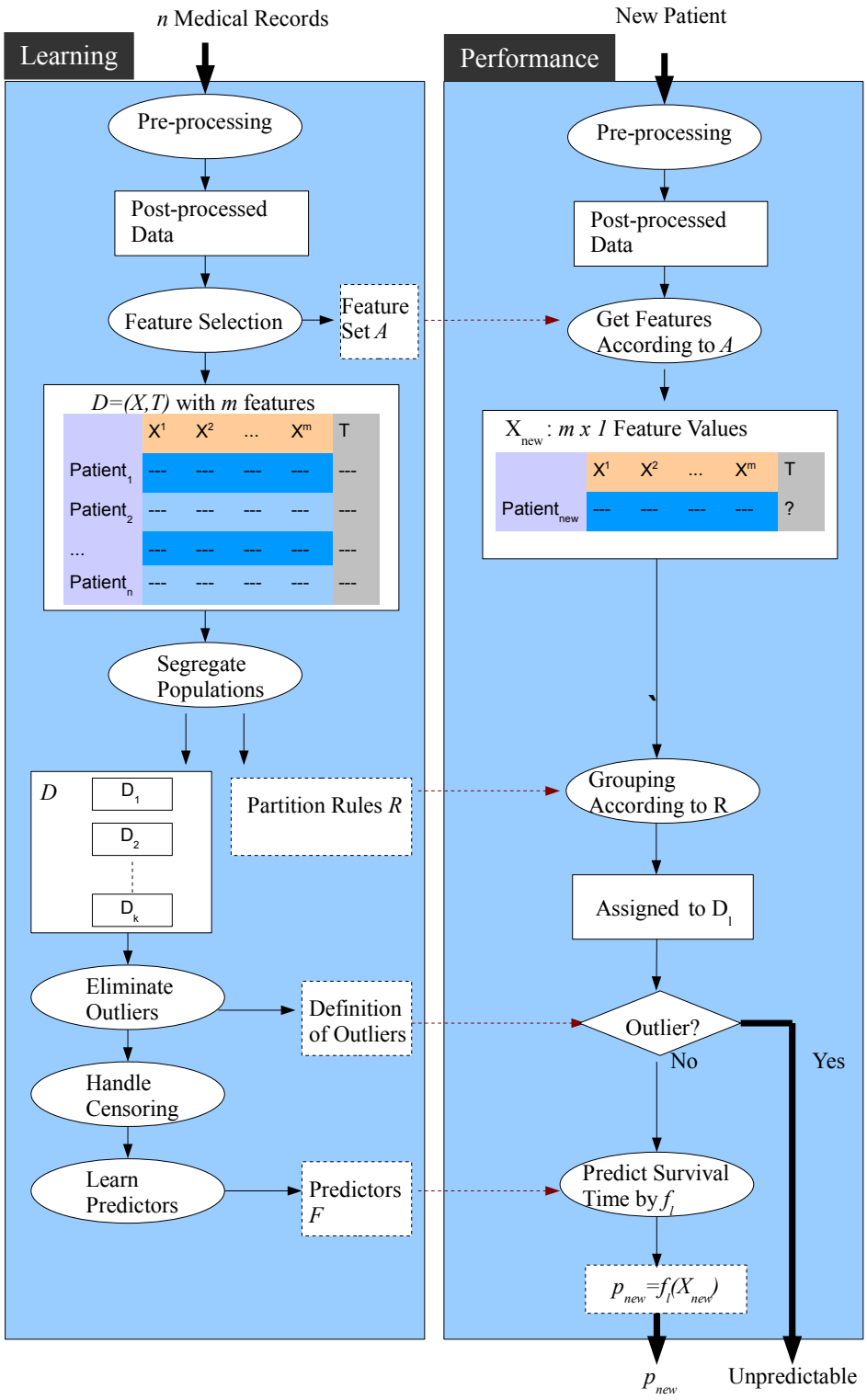


Figure 4.1: A framework of survival predictions

4.1 Censoring

Working with survival data is challenging as the data set is often incomplete. This is especially problematic when the class labels (survival times) are missing since supervised learning algorithms rely on these labels for training a predictor.

In survival analysis, missing survival times are called *censoring*. Even though the survival time is unknown for a censored patient, the censored time of this patient provides us important information about this censored patient — namely, a lower bound of the patient's actual survival time. Simply eliminating censored patients or treating the censored time as a survival time would bias our predictors.

Alternatively, we could just eliminate the censored patients. However, the quality of a predictor is highly correlated to the size of the data set, and survival data is usually insufficient for training an accurate predictor. To minimize the amount of lost information, the censored observations should not be eliminated.

In our framework, we apply several supervised learning algorithms. In order to use these algorithms, we need to include the survival times as the class label for both censored and uncensored patients. How could we utilize censored information effectively? We consider several approach to handle censored observations including:

- Approximate event times for censored observations (Section 4.1.1)
- Consider censored observations are uncensored, but with "lower weights" than uncensored observations (Section 4.1.2)

Recall our formulations of survival data in Table 2.1, we define t_i to be the class label (survival time or censored time) of the i^{th} patient, c_i to be the censored flag of the i^{th} patient where 0 indicates uncensored and 1 indicates censored, and $\Re_i = \{\text{Patient}_j | t_j \geq t_i\}$ to be the *risk set* at time t_i.

4.1.1 Approximation of Survival Time

One approach to handle censoring is to impute a survival time $\hat{t}_i$ for each censored observation. We consider the following three techniques: (1) adding a constant to the class label of each censored patient, (2) taking the average survival time of uncensored patients in the *risk set*, or (3) taking the average survival time of all patients in the *risk set*.

Approximation by Adding Constant Time

A naive approach is to add an expected residual time on top of the censored time for each censored patient, and consider this resulting observation as uncensored. For example, we can approximate a survival time for a censored patient by adding 12 months to the patient's censored time and then treating this patient as uncensored. Table 4.1 shows the result of this approach on our example data in Table 2.3.

Patient	t_i	c_i	$\hat{t}_i^C$
Patient_1	1	0	-
Patient_2	8	0	-
Patient_3	10	1	10+12=22
Patient_4	13	0	-
Patient_5	18	1	18+12=30
Patient_6	120	0	-

Table 4.1: An example of approximating event times by adding a constant

Approximation by Averaging over Uncensored Observations

However, approximating event times by adding a constant is not completely appropriate. Assuming that we have a population of patients such that lots of patients die at 5 years. Imagine that we have a patient censored at 4 years, we anticipate that this patient will survive approximately 1 year more. Therefore, we assume that the remaining time of a censored patient is somehow related to the survival times of the rest of the population, and we should treat each censored patient differently.

One of our approaches to approximating an event time for a censored patient is to take the average survival time of all uncensored patients who had survived longer than this patient's censored time. For each censored patient Patient_i who censored at time t_i, the estimated survival time $\hat{t}_i^U$ is computed as the average time, over all uncensored times longer than the censored time t_i:

$$\hat{t}_i^U = \frac{1}{|\{j|c_j = 0, t_j \geq t_i\}|} \sum_{\{j|c_j=0, t_j \geq t_i\}} t_j$$

Table 4.2 is an example of approximating event time from uncensored observations for the same data set in Table 2.3. Patient_3 is censored at time $c_i = 10$, and the survival times of Patient_4 and Patient_6 are longer than Patient_3, so we approximate the survival time of Patient_3 by taking the average survival time of Patient_4 and Patient_6. As another example, Patient_5 is censored at time $c_i = 18$, and Patient_6 is the only uncensored patient who survived longer than Patient_5; hence, the approximated survival time for Patient_5 is 120.

Patient	t_i	c_i	$\hat{t}_i^U$
Patient_1	1	0	-
Patient_2	8	0	-
Patient_3	10	1	(13+120)/2 =66.5
Patient_4	13	0	-
Patient_5	18	1	120/1 = 120
Patient_6	120	0	-

Table 4.2: An example of approximating event times by averaging over uncensored patients in the *risk set*

Approximation by Averaging over All Observations

Another approach is to approximate an event time by taking the average survival time of all patients in the *risk set*. This is different from the last technique in that this approach includes censored patients when computing the average survival time and the class label t_j of a censored patient Patient$_j$ in the *risk set* is replaced by the patient's approximated survival time $\hat{t}_j^A$. For each censored patient Patient$_i$ who censored at time t_i, the estimated survival time $\hat{t}_i^A$ is computed as the average time, over the survival times longer than the censored time t_i:

$$\hat{t}_i^A = \frac{1}{|\{j|t_j \geq t_i\}|} \left(\sum_{\{j|c_j=0, t_j \geq t_i\}} t_j + \sum_{\{j|c_j=1, t_j \geq t_i\}} \hat{t}_j^A \right)$$

Table 4.3 is an example of approximating event time from both censored and uncensored observations for our example data in Table 2.3. As the same example above, Patient$_3$ is censored at time $c_i = 10$, but now the approximated time is the average over both censored and uncensored patients who were still alive. In this example, Patient$_4$, Patient$_5$, and Patient$_6$ survived longer than 10, we approximate the survival time of Patient$_3$ to be the average survival time of Patient$_4$, Patient$_5$, and Patient$_6$.

Patient	t_i	c_i	$\hat{t}_i^A$
Patient$_1$	1	0	-
Patient$_2$	8	0	-
Patient$_3$	10	1	(13+120+120)/3 $\approx$ 84.33
Patient$_4$	13	0	-
Patient$_5$	18	1	120/1 = 120
Patient$_6$	120	0	-

Table 4.3: An example of approximating event times by averaging over the *risk set*

4.1.2 Weighting Censoring

Another approach on handling censored information is to consider the censored observations as uncensored data, but with "lower weights" than the uncensored observations. The basic concept is to consider censored observations as unreliable information, so the effect of censored data should be slightly smaller. Here, we assign a constant weight to each censored patient. We perform experiments with different constants, including 0.1, 0.2, and 0.5, and we use the experimental results to determine the most appropriate weight for censored data.

4.2 Regression

In our work, our objective is to find the best predictor for each individual patient. Recall our formulation in Table 2.1, given a set of n training data $D = (X, T)$ where $X = \{X_1, X_2, ..., X_n\}$ represents feature values of these n patients, and $T = [t_1, t_2, ..., t_n]$ represents the class labels of these n patients, our goal is to apply a learning algorithm $L(.)$ on the data D to produce a predictor $f = L(D)$. For each individual patient with feature values X_{new}, the predictor f returns a survival time $p_{new} = f(X_{new})$ for this patient.

Supervised learning builds regression models given a set of feature values X and class labels T. As mentioned before, survival data suffers from the presence of censored observations, which should be handled differently — i.e., it is problematic to simply treat them as uncensored data. We utilize censored data in the training phase in two ways:

1. Approximate event times for censored observations (Section 4.1) and then apply regular regression algorithms (will be discussed in Section 4.2.1)
2. Modify regression algorithms to accommodate censored observations (will be discussed in Section 4.2.2)

As the relationship between features and survival time is still not understood, we consider various algorithms to learn these models from existing historical patients. We select both linear and nonlinear regression algorithms, including the *linear regression*, the *support vector regression*, the *regression trees*, and some variations of the above. In our work, we consider the following 6 learning algorithms and we will discuss each of them in this section:

- *LIN*: *linear regression*
- *SVR*: *support vector regression*
- *RT*: *regression trees*
- *LINc*: *linear regression for censored targets*
- *SVRc*: *support vector regression for censored targets*
- *GAT*: *gating regression*

Recall our formulation in Table 2.1, given a data set $D = (X, T)$ of n instances each with m features, we define X to be a $n \times m$ feature matrix, X^j to be the feature values of the j^{th} feature, X_i to be the features values of the i^{th} patients, x_i^j to be the feature value of the j^{th} feature of the i^{th} patients, $T = [t_1, t_2, ..., t_n]$ to be a n-dimensional vector representing the class labels of these n patients, $P = [p_1, p_2, ..., p_n]$ to be a n-dimensional vector representing the predicted survival time of these n patients, and $C = [c_1, c_2, ..., c_n]$ to be a n-dimensional vector representing the censored flag of these n patients where 0 indicates uncensored and 1 indicates censored. Also, we use $\langle X_i, X_j \rangle$ to denote the dot product of two vectors X_i and X_j.

4.2.1 Regression Algorithms

This section covers the basic theory of the regression algorithms we considered, including the *linear regression*, the *support vector regression*, and the *regression trees*. Recall that our input data contains censored observations (i.e., the class label of a patient is a lower bound of the patient's actual survival time), but these learning algorithms require class labels for training a predictor. To apply these algorithms, we first approximate class labels for censored observations using the techniques we discussed in Section 4.1 and then uses these revised values as labels (i.e., then threat these instances as uncensored).

Linear Regression

The *linear regression* method assumes that the class labels T is nicely approximated as a linear combination of the features X. Let $\beta^0 \in \mathbb{R}$ and $\beta \in \mathbb{R}^m$ be the parameters of a predictor f (i.e., a linear function of features X) learned from the *linear regression* algorithm. Here, the parameters $\beta = \left[\beta^1, \beta^2, ..., \beta^m\right]$ are coefficients for feature values of each corresponding feature in X (i.e., β^1 for the feature values of the 1^{st} feature X^1, β^2 for the feature values of the 2^{nd} feature X^2, etc.), and the parameter β^0 is the intercept of this linear function. Given a m-dimensional feature vector X_i, the predictor f returns a prediction $p_i = f(X_i) = \beta^0 + \langle \beta, X_i \rangle$.

A typical method of estimating the parameters β^0 and β is to minimize the *residual sum of squares* (RSS) of the prediction P from T. The RSS is calculated by:

$$RSS = \sum_{i=1}^{n} (t_i - p_i)^2 \tag{4.1}$$

In practice, it is more convenient to use matrix operations to express this process [21]. Recall that X is a $n \times m$ matrix, we can add a n-dimensional column vector of 1's to X, such that $X' = [1 : X]$ is a $n \times (m+1)$ matrix. Then, we can combine β^0 and β so that $\beta' = \left[\beta^0, \beta^1, ..., \beta^m\right]$. Given a m-dimensional feature values of a patient X_i, the output of f can be calculated as $f(X_i) = \langle X_i' \beta' \rangle$, and Equation 4.1 can be rewrite as:

$$RSS = (T - X'\beta')^T (T - X'\beta') \tag{4.2}$$

To minimize the *residual sum of squares*, we set the derivative of Equation 4.2 equal to zero and solve for β' to obtain

$$\beta' = (X'^T X')^{-1} X'^T T$$

Support Vector Regression

The *support vector regression* (SVR) aims to find a function f such that the estimated class label $f(X_i)$ of a sample X_i has at most ϵ deviation from the class labels t_i for all training instances, and the parameters are as small as possible [55]. Here, ϵ is a control variable that determines the margin of acceptable error. For example, in the left figure of Figure 4.2, the greyed-out region is called the

ϵ-*tube*. Each point (the $\times$) is an instance, and the distance between each $\times$ and the middle line is the prediction error of that instance. The goal of *SVR* is to find a function f such that all $\times$ fall within this ϵ-*tube*.

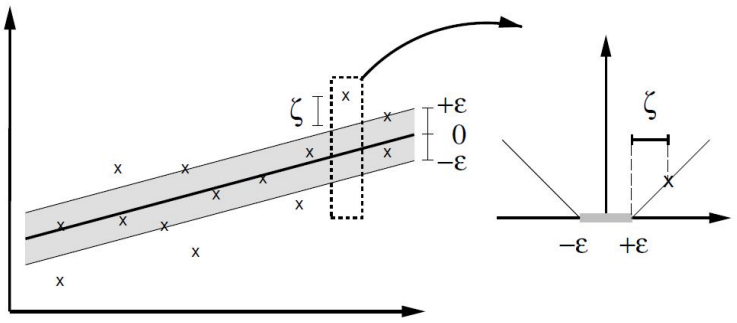


Figure 4.2: A visualization of the *support vector regression* with soft margin, extracted from [55].

This optimization problem might not be feasible (i.e., there exists no f such that all instances satisfy these constraints), and we may want to allow some errors; hence, Cortes and Vapnik introduced the slack variables ξ_i and ξ_i^* to cope with infeasible constraints [8]. Figure 4.2 illustrates the *support vector regression* with soft margin. In the left figure, a point $\times$ outside of the ϵ-*tube* is now acceptable but will be penalized by a loss function. The right figure shows the loss function where the *x-axis* is the prediction error $|t_i - p_i|$, and the *y-axis* is the amount of loss contributed by X_i. If $|t_i - p_i| < \epsilon$, the loss is 0; otherwise, the loss is the deviation of the prediction error $|t_i - p_i|$ from ϵ. More precisely, the slack variables are characterized by the loss function such that

$$\xi_i = \begin{cases} 0 \text{ if } t_i - p_i \leq \epsilon \\ t_i - p_i - \epsilon \text{ otherwise} \end{cases}$$
$$\xi_i^* = \begin{cases} 0 \text{ if } p_i - t_i \leq \epsilon \\ p_i - t_i - \epsilon \text{ otherwise} \end{cases}$$

Similar to the *linear regression*, we define parameters $\beta = \left[\beta^1, \beta^2, ..., \beta^m\right] \in \mathbb{R}^m$ as the vector of coefficients for feature values of each corresponding feature in X and $\beta^0 \in \mathbb{R}$ as the intercept of the resulting predictor. Given a m-dimensional feature vector X_i, the *SVR* estimates class label by:

$$f(X_i) = \langle \beta, X_i \rangle + \beta^0$$

The goal of the *SVR* with soft margin is to minimize the magnitude of the parameters β and the

sum of losses over all instances, which can be described as a convex optimization problem:

$$\min_{\beta} \frac{1}{2}\langle\beta,\beta\rangle + C\sum_{i=1}^{n}(\xi_i + \xi_i^*)$$
$$\text{subject to } \begin{cases} t_i - \langle\beta, X_i\rangle - \beta^0 \leq \epsilon + \xi_i \\ \langle\beta, X_i\rangle + \beta^0 - t_i \leq \epsilon + \xi_i^* \\ \xi_i, \xi_i^* \geq 0 \end{cases}$$

Here, C is a single regularization parameter that controls the trade-off between the smoothness of f and the amount of tolerable error larger than ϵ. The *SVR* can be solved by its dual formulation, which will reduce the complexity of this optimization problem. Letting α_i and α_i^* be the Lagrange multipliers, the dual optimization problem becomes:

$$\min_{\alpha_i,\alpha_i^*} \frac{1}{2}\sum_{i,j=1}^{n}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j*)\langle X_i, X_j\rangle + \epsilon\sum_{i}^{n}(\alpha_i + \alpha_i^*) - \sum_{i}^{n} t_i(\alpha_i - \alpha_i^*)$$
$$\text{subject to } \begin{cases} \sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0 \\ \alpha_i, \alpha_i^* \in [0, C] \end{cases}$$

The next step is to estimate the parameter α_i, α_i^* and β^0. The optimization process of *SVR* are not discussed here, as the details can be found in [53]. Smola and Scholkopf also proposed the *sequential minimal optimization* algorithm for optimizing the *support vector regression* by using a single threshold value [53]. Shevade et al. extends the work of Smola and Scholkopf to a more efficient implementation [50]. In our work, we use the Shevade et al.'s method to learned the *SVR* predictor. Once the parameters are solved, the predicted survival time of an individual patient with feature value X_{new} can be estimated by:

$$p_{new} = f(X_{new}) = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)\langle X_i, X_{new}\rangle + \beta^0$$

The *linear regression* and the *SVR* that we just discussed assume that the class labels can be approximated as a linear combination of the features. However, it is never been proven that the survival times can be described as a linear combination of the feature values. To address the potential problem of this assumption, the *SVR* algorithm can be nonlinear by incorporating a *kernel* method; that is, a function that maps each training instance from m-dimensional space to a higher dimensional space. (There is a large literature on the *kernel* method; cf. [2].) In our work, we test our *SVR* with the following *kernel* functions:

$$\textit{Polynomial of degree 2} : K_2(X_i, X_j) = (1 + \langle X_i, X_j\rangle)^2$$
$$\textit{Gaussian Radial Basis} : K_{GRB}(X_i, X_j) = e^{-\sum_{k=1}^{m}(X_i^k - X_j^k)^2}$$

The learning process of the kernelized *SVR* is similar to that of the original SVR, except that the

dot-product is replaced by the *kernel* function.

$$\min_{\alpha_i,\alpha_i^*} \frac{1}{2} \sum_{i,j=1}^{n} (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j*)K(X_i, X_j) + \epsilon \sum_{i}^{n} (\alpha_i + \alpha_i^*) - \sum_{i}^{n} t_i(\alpha_i - \alpha_i^*)$$

$$\text{subject to } \begin{cases} \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) = 0 \\ \alpha_i, \alpha_i^* \in [0, C] \end{cases}$$

where $K(,)$ is a *kernel* function

Once the parameter β^0, α_i, and α_i^* are solved, the predicted survival time of an individual patient with feature value X_{new} can be estimated by:

$$p_{new} = f(X_{new}) = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*)K(X_i, X_{new}) + \beta^0$$

Regression Trees

The *Classification and Regression Trees (CART)* is an algorithm that generates a decision tree [4], and the *regression trees* is a type of CART, which generates a decision tree and then applies regression method on the decision tree for predicting continuous class labels [44].

Given a data set D, we aim to partition D into a tree of k leaf nodes (i.e., D is divided into k subsets $\{D_1, D_2, ..., D_k\}$) and train a predictor using the *linear regression* algorithm (as discussed in Section 4.2.1) on each subset D_i. The learning process of the *regression tree* consists of the following components: (1) A set of *partition rules*, (2) a *splitting criterion*, (3) a *stopping criterion*, (4) a learning algorithm, and (5) a *pruning criterion*. We will discuss our components in this section.

Here is a brief summary of the learning steps. This learning system produces the tree node by node; on the root of the tree, we seek a *partition rule* R_1 that splits D into $D_{R_1,-}$ and $D_{R_1,+}$. In this splitting process, the challenge is finding the "best" partition rule, where the goodness is evaluated by the *splitting criterion*. We can repeat this process recursively on each subset of D (i.e., seek a partition rule R_2 to split $D_{R_1,-}$ and another partition rule R_3 to split $D_{R_1,+}$) until the *stopping criterion* is met. After this recursively splitting process, the partitions will form a tree-like structure. Algorithm 4 shows the general framework of growing a tree. Then, the *pruning criterion* is applied to remove some previous defined partition rules that do not contribute to regression accuracy on unseen data. Finally, a predictor is trained using the *linear regression* on the subset of data within each corresponding node.

Figure 4.3 shows an example of splitting a tree. The input data D is divided into 5 groups $D = \cup\{D_1, D_2, D_3, D_4, D_5\}$ in this example. The left figure shows the partitions of the data in two dimensional representation, and the right figure shows the tree representation. In the right figure, each internal node is associated with a partition rule that divides the input data of that node into two subsets. For example, at the root of the tree, the data D is divided into two subsets D_- and D_+ by the partition rule R_1. At the end of the splitting process, the tree has 5 leaf-nodes, each represents the corresponding subset $\{D_1, D_2, D_3, D_4, D_5\}$.

Algorithm 4 TREE-SPLIT (D)

Input:

D: the input data set

Output:

$Tree$

if stopping criterion are met **then**
  $Tree \leftarrow$ LEAF-NODE (D)
else
  for $R_i \in$ all possible partition rules R **do**
    $score_i \leftarrow Eval(D, R_i)$
  $i^* \leftarrow \arg\max\{score_i\}$
  Partition D into $D_{R_{i^*},-}$ and $D_{R_{i*},+}$ according to R_{i*}
  $Tree \leftarrow$ INTERNAL-NODE (R_i^*)
  $Tree.TrueBranch \leftarrow$ TREE-SPLIT($D_{R_{i^*},+}$)
  $Tree.FalseBranch \leftarrow$ TREE-SPLIT($D_{R_{i^*},-}$)
return $Tree$

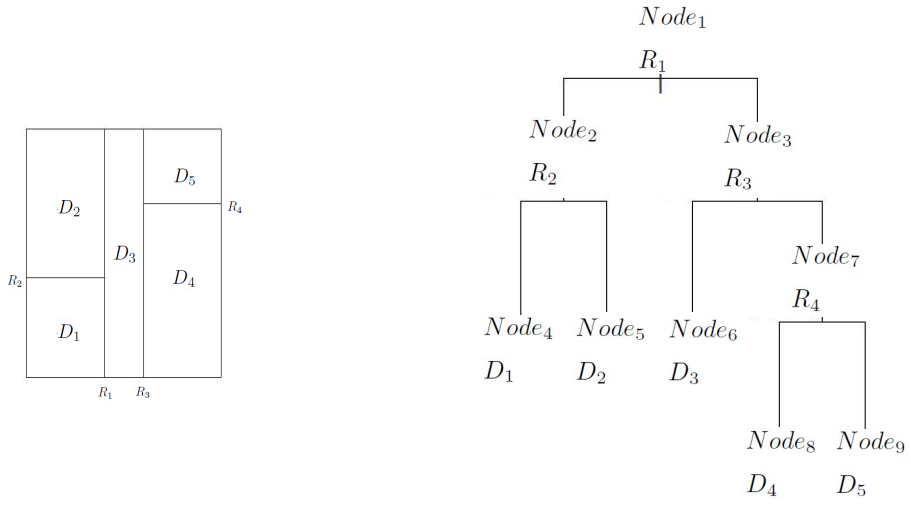


Figure 4.3: An example of tree splitting

Here, we will describe each component of the *regression trees* learning mechanism:

1. Partition Rules

   A partition rule R is a question of the form "Is $X^j \leq z^j$?" where the superscript j specifies the j^{th} feature of an instance X. Here, the j^{th} feature is called the *splitting feature*, and z^j is called the *splitting value*.

   In our data sample $D = (X, T)$, X contains a mixture of discrete and continuous features. If X^j is a discrete variable, then the splitting value z^j can be an outcome of X^j. Otherwise, a partition rule can be a binary test based on comparing the value of X^j against a threshold value z^j.

   At each node $Node_l$, a partition rule $R_l : X^j \leq z^j?$ divides the data D_l into subsets by using X^j at z^j — that is, D_l is divided into $D_{R_l,-}$ and $D_{R_l,+}$ such that $D_{R_l,-} = \{(X_i, t_i) \in$

$D_l|x_i^j \leq z^j\}$ and $D_{R_l,+} = \{(X_i, t_i) \in D_l|x_i^j > z^j\}$. That is, $D_{R_l,-}$ contains all instances $\in D_l$ whose feature value of the j^{th} feature $\leq z^j$, and $D_{R_l,+}$ contains all instances whose feature value of the j^{th} feature $> z^j$.

2. Splitting Criterion

   A splitting criterion is designed to evaluate a partition rule. Recall that the *regression trees* learning system produces the tree node by node. At each node $Node_l$, given a training sample D_l, we seek the "best" partition rule R_l to split D_l into $D_{R_l,-}$ and $D_{R_l,+}$. The challenge is to find the "best" R_l and determine whether it is "worthwhile" to split D_l. We have some liberties in how we define "best" and "worthwhile". The *splitting criterion* is designed to evaluate each R_l at $Node_l$.

   In our work, we used the *expected reduction in error* introduced by Quinlan as the splitting criterion for our *regression trees* algorithm [44]. The goal of this splitting criterion is to maximize the similarity between patients within the same subset, where "similarity" of a group is measured by the standard deviation of the class labels in that group. At each node, the partition rule with the highest *expected reduction in error* among all possible partition rules in that node is selected. More precisely, given a data set D, we first compute σ_T, the standard deviation of class labels T in D_l. For each possible partition rule that splits D into $D_{R,-}$ and $D_{R,+}$, we then calculate $\sigma_{T_{R,-}}$ and $\sigma_{T_{R,+}}$ (i.e., the standard deviation of class labels of $D_{R,-}$ and $D_{R,+}$ respectively). The *expected reduction in error*, denoted by $\nabla error(D, R)$, is defined as

$$\nabla error(D, R) = \sigma_T - \left(\frac{|D_{R,-}|}{|D|} \times \sigma_{T_{R,-}} + \frac{|D_{R,+}|}{|D|} \times \sigma_{T_{R,+}} \right)$$

3. Learning Algorithm

   A learning algorithm $L(.)$ can be applied at each node of the tree. Here, $L(.)$ could be any standard regression methods, and the *linear regression* method (Section 4.2.1) is used in our work. Given a training data D_l and a learning algorithm $L(.)$ at each node $Node_l$ of the tree, a linear predictor $f_l = L(D_l)$ is constructed. Given a new patient with feature values X_{new}, this patient is assigned to one of the leaf nodes according to the set of partition rules R. Assuming that this patient is assigned to $Node_l$, the predicted survival time p_{new} for this patient is the output of the corresponding predictor $f_l(X_{new})$ in $Node_l$.

   How good is our linear model? We are often interested in the *expected prediction error* of a predictor f on unseen data. Given a training set $D_{training}$ and a testing set $D_{testing}$, a typical way to evaluate the performance of a predictor $f = L(D_{training})$ is to calculate the *average L1 error* (Section 3.2), which measures the average magnitude of difference between actual class label t_i and estimated class label p_i. Quinlan states that the *average L1 error* usually underestimates the *expected prediction error* of f when the number of instances in $D_{training}$

is small and the number of parameters in f is relatively large [44]. One way to address this potential problem is to penalize linear models with large number of parameters constructed from small training samples. Letting n be the number of instances in $D_{training}$ and v be the number of parameters in f, assuming that all models are restricted to $n \geq v$, we multiply the *average L1 error* by $\frac{n+v}{n-v}$. Given a testing set $D_{testing}$, the *expected prediction error* of a predictor f is evaluated by

$$Eval_{RT}(f, D_{testing}) = \frac{n+v}{n-v} \times \frac{1}{|D_{testing}|} \sum_{i=1}^{|D_{testing}|} |t_i - p_i| \tag{4.3}$$

Imagine two linear models f_1 and f_2 are constructed from the same number of training samples n such that f_1 has v_1 parameters and f_2 has v_2 parameters. If $v_1 > v_2$, then f_1 will be penalized more (by $\frac{n+v}{n-v}$) than f_2, which indicates that f_2 is a better predictor.

4. Pruning Criterion

   The *pruning criterion* is designed to evaluate a partition. The pruning process removes splits from the initial tree that does not contribute to better regression results. This is accomplished by comparing the performance of the linear model at each node.

   In more detail, each node $Node_l$ is associated with a data set D_l, and we want to measure the similarity of D_l in $Node_l$. A 3-fold cross-validation (Section 3.1) is applied to evaluate the performance of the *linear regression* learner $L(.)$ (Section 4.2.1) on the data set D_l, and the quality of $L(.)$ on D_l is measured by $Err_{CV}(L, D_l)$ — i.e., the *average cross-validation error*. A *k-fold cross-validation* learns k linear predictors, and the performance of each predictor f_l on each hold out set $D_{l,holdout}$ in the cross-validation process is calculated using Equation 4.3, the *expected prediction error* $Eval_{RT}(f_l, D_{l,holdout})$. The pruning step takes the bottom-up approach, in which the process starts from the bottommost leaf and works toward to the root. If the *average CV error* of an internal node is smaller than the sum of the *average CV errors* of its descendent nodes (i.e., the quality of $L(.)$ is better before splitting), then this split will be pruned away and this internal node is turned into a leaf node. That is, if $Err_{CV}(L, D_l) < (\frac{|D_{R_l,-}|}{|D_l|} Err_{CV}(L, D_{R_l,-}) + \frac{|D_{R_l,+}|}{|D_l|} Err_{CV}(L, D_{R_l,+})$, the split at $Node_l$ will be removed.

Figure 4.4 shows an example of the *regression trees* model. This example *regression tree* consists of a set of 4 partition rules $R = \{R_1, R_2, R_3, R_4\}$ and a set of 5 predictors $F = \{f_1, f_2, f_3, f_4, f_5\}$. The partition rules R divide the data D into 5 regions $\{D_1, D_2, D_3, D_4, D_5\}$. Each region D_l in D is associated with a corresponding predictor f_l in F which is learned using the *linear regression* algorithm on the data D_l.

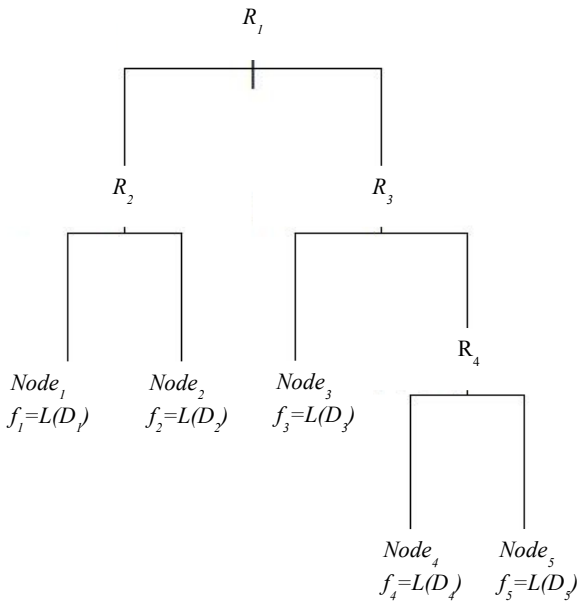


Figure 4.4: An example of using the *regression trees*

4.2.2 Regression Algorithms for Censored data

Survival data suffers from the presence of censored observations, but most supervised learning methods require class labels for learning a predictor. Simply eliminate censored observations from our data set or treat them as uncensored data will bias the resulting predictors. The previous section (Section 4.2.1) discusses our approach of imputing a survival time for each censored patient before learning a predictor. In this section, we will discuss our approach of modifying the *linear regression* and the *support vector regression* to utilize censored data.

Linear Regression for Censored Targets

In Section 4.2.1, we discussed the *linear regression* method, which used predictors that estimate the class label $p_i = f(X_i) = \beta^0 + \langle \beta, X_i \rangle$, based on parameters β^0 and β that are estimated by minimizing the *residual sum of squared error* $RSS = \sum_{i=1}^{n} (t_i - p_i)^2$. Buckley and James introduce a method to train a linear predictor that allows the class labels to be unspecified — by eliminating the penalty on overestimating the prediction for censored data during training phase [5]. That is, the parameters β^0 and β are estimated by minimizing

$$RSS_{Buckley} = \sum_{\{i|(c_i=0)\text{or}(c_1=1,p_i<t_i)\}} (t_i - p_i)^2$$

We also show how to express the RSS using matrix format, such that $RSS = (T - X'\beta')^T (T - X'\beta')$ (Equation 4.2), which meant the parameters $\beta' = \left[\beta^0, \beta^1, ..., \beta^m\right]$ can be solved by $\beta' = (X'^T X')^{-1} X'^T T$. Orbe et al. [37] introduce a method of learning the parameters β' by weighted

RSS. Their work assumes that there exist a linear relationship between the feature values X and the logarithm of class labels $\ln(T)$ and tries to minimize the weighted *residual sum of squares* $RSS_{Orbe} = W(\ln(T) - X'\beta')^T(\ln(T) - X'\beta')$ where W is a $n \times n$ diagonal matrix formed with the weights of each instance. The parameter β' can be solved by:

$$\beta' = (X'^T W X')^{-1} X'^T W \ln(T)$$

In our work, we incorporate the ideas of Buckley and James and Orbe et al. on learning a predictor with censored data. Recall that $P = [p_1, p_2, ..., p_n]$ denotes the predicted survival times such that $p_i = X_i'\beta'$and $C = [c_1, c_2, ..., c_n]$ denotes the censored flag where the subscript i specifies the i^{th} patient. We employ a $n \times n$ diagonal matrix W such that

$$W_{ii} = \begin{cases} 0, \text{ if } c_i = 1 \text{ and } p_i > c_i \\ 1, \text{ otherwise} \end{cases}$$

That is, if the prediction for a censored patient is longer than the patient's censored time, we consider this prediction to be correct. The parameters β' can be estimated by

$$\beta' = (X'^T W X')^{-1} X'^T W Y$$

Support Vector Regression for Censored Targets

In Section 4.2.1, we discuss the *support vector regression*, which aims to minimize the training error controlled by the error margin parameter ϵ and the regularization parameter C. The *SVR* has proven to be a robust and powerful algorithm in various applications. However, the conventional *SVR* cannot handle the difference between censored and uncensored observations. Khan and Zubek introduce a variation of the SVR, the *support vector regression for censored targets (SVRc)* to account this problem [28].

First, the parameters ϵ and C for censored data and uncensored data are separated. Recall that ϵ defines the acceptable margin of error, and C controls the amount of loss in the SVR. Khan and Zubek asymmetrically modify the loss function by introducing new parameters to replace ϵ and C in the original SVR. The *SVRc* uses subscript n to denote uncensored data and subscript c to denote censored data. For example, ϵ_n defines the acceptable margin of error for uncensored data, and ϵ_c defines the acceptable margin for censored data.

Secondly, the *SVRc* introduces separated parameters for underestimation and overestimation and uses the superscript $*$ to indicate overestimation. In case of censored data, ϵ is replaced by ϵ_c^* and ϵ_c where ϵ_c^* defines the acceptable margin when prediction is greater than the actual survival time, and ϵ_c defines the acceptable margin when prediction is less than the actual survival time. Similarly, C is replaced by C_c^* and C_c where C_c^* controls the penalty of overestimation and C_c controls the penalty of underestimation.

The censored observations are handled by setting C_c^* to be smaller than C_c and ϵ_c^* to be larger than ϵ_c, so the penalty is smaller when the prediction is longer than the censored time. Figure 4.5

illustrates the modifications and parameters in the loss function of SVRc.

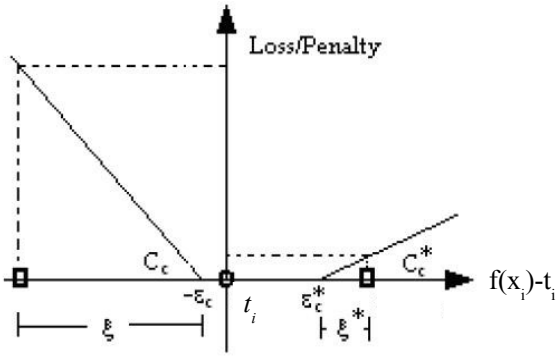


Figure 4.5: A visualization of the loss function of the *SVRc* (extracted from [28])

The objective function has become:

$$\min_{\beta} \frac{1}{2}\langle\beta,\beta\rangle + \sum_{i=1}^{n}(C_i\xi_i + C_i^*\xi_i^*)$$

$$\text{subject to } \begin{cases} t_i - \langle\beta, X_i\rangle - \beta^0 \leq \epsilon_i + \xi_i \\ \langle\beta, X_i\rangle + \beta^0 - t_i \leq \epsilon_i^* + \xi_i^* \\ \xi_i, \xi_i^* \geq 0 \end{cases}$$

$$\text{where } \begin{cases} C_i = c_i C_c + (1-c_i)C_n \\ C_i^* = c_i C_c^* + (1-c_i)C_n^* \\ \epsilon_i = c_i\epsilon_c + (1-c_i)\epsilon_n \\ \epsilon_i^* = c_i\epsilon_c^* + (1-c_i)\epsilon_n^* \end{cases}$$

4.2.3 Gating Regression

Since the relationship between features and survival time is still not well understood, we consider learning several predictors from existing historical patients and selecting the best algorithm for our task of survival prediction. Here, we assume that some learning algorithms are better for some input data but not the others.

Therefore, the *gating regression* approach will learn a set of candidate predictors from a set of learning algorithms L (including all learning algorithms that we described in the previous two sections) and automatically select the "best" algorithm among all candidate algorithms in L for a given data set.

First of all, we needs to define an evaluation criterion to determine the notion of the "best" regressor. Given a data set $D_{training}$ and a set of learning algorithm L, we perform experiments with a 3-folds cross-validation (Section 3.1) on each learning algorithm $L_i(.)$ in L, and use the *average cross validation error* $Err_{CV}(L_i, D_{training})$ to rank each $L_i(.)$. Since there is no single

best way to evaluate survival predictors on the hold-out set, we also try different measurements in our experiments. The candidate evaluation methods include *average L1 error*, *average L2 error*, *average relative absolute error*, and *correlation coefficient* (see Section 3.2 for details on these error functions).

Figure 4.6 illustrates the process of the *gating regression*. Given an input data set $D_{training}$, and a set of candidate learning algorithms $L = \{L_1(.), L_2(.), ..., L_5(.)\}$, the *gating regression* finds the learning algorithm with the lowest *average CV error* and returns a predictor f_{best} constructed from that algorithm. Algorithm 5 shows the pseudo-code of the *gating regression*

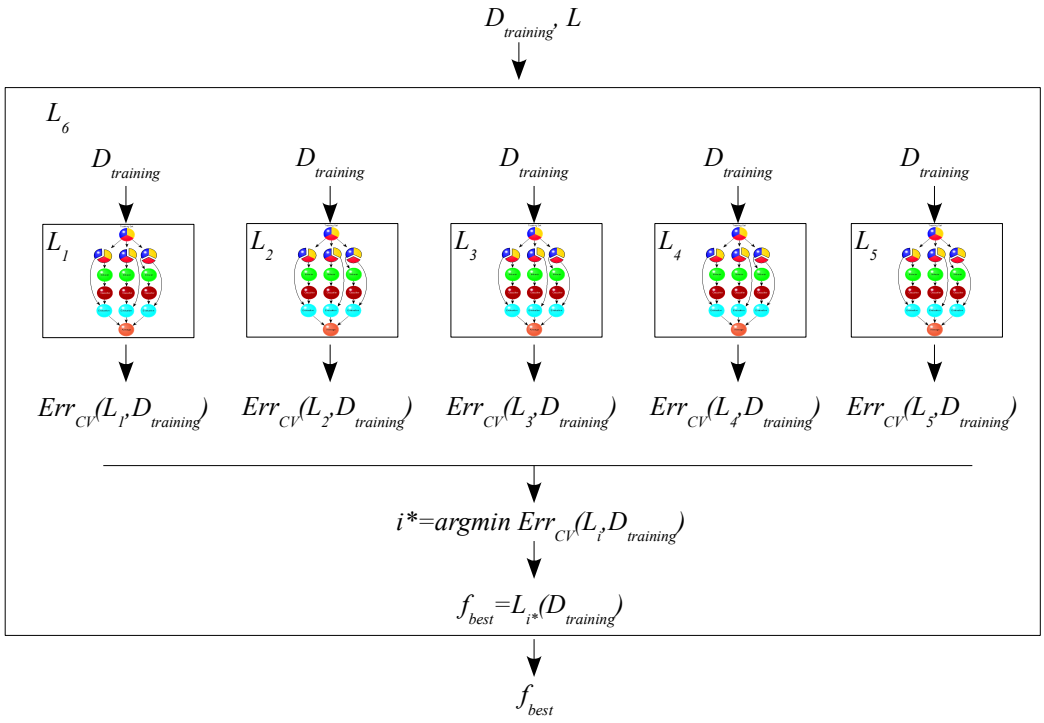


Figure 4.6: The *gating regression* algorithm

Algorithm 5 The *gating regression*

Input:

$D_{training}$: data set

$L = \{L_1(.), L_2(.), ..., L_p(.)\}$: a set of p learning algorithms

k: the number k of folds in a *k-fold cross-validation*

Output:

f_{best}: the best predictor

Divide $D_{training}$ into k folds $D_{training} = \cup\{D_1, D_2, ..., D_k\}$
for all $L_i(.)$ in L **do**
  for all D_j in $D_{training}$ **do**
    $D_{testing,j} \leftarrow D_j$
    $D_{training,j} \leftarrow D_{training} - D_j$
    $f \leftarrow L_i(D_{training,j})$
    $score_{i,j} \leftarrow eval(f, D_{testing,j})$
$Err_{CV}(L_i, D_{training}) \leftarrow \frac{1}{k} \sum_{j=1}^{k} score_{i,j}$
$i^* \leftarrow \arg\min_i Err_{CV}(L_i, D_{training})$
$f_{best} \leftarrow L_{i^*}(D_{training})$
return f_{best}

4.3 Grouping

One of the challenges in survival prediction is that patients are heterogeneous, with different survival patterns for different subgroups of patients. For example, the variable cigarette smoking may be an important factor for lung cancer patient but perhaps not as critical as for pancreas cancer patients. In order to overcome this issue of different dependency relations, we attempt to learn a predictor for each risk group. Unfortunately, it is not known which group of patients share the same survival pattern. Therefore, in this thesis, a major task is to design an appropriate method that can effectively "group" patients — i.e., segregate patients with different survival distributions.

Given a data set $D = (X, T)$, where X is the feature matrix and T is the vector of class labels, we define (in Table 2.1) X^j represents the feature values of the j^{th} feature over patients, X_i represents the set of feature values of the i^{th} patient, x_i^j represents the feature value of the j^{th} feature of the i^{th} patient, and t_i represents the class label of the i^{th} patient. The goal of segregating patients is to partition D into k subsets $\{D_1, D_2, ..., D_k\}$ such that the difference between patients in the same subset is minimized, and also the difference between patients across different subsets are maximized. This section summarizes our methods on grouping D into subcategories. We consider two types of grouping mechanisms:

- The *Classification and Regression Trees (CART)*, which is *dependent* to the class labels, will be discussed in Section 4.3.1
- The *clustering*, which is *independent* to the class labels, will be discussed in Section 4.3.2

Recall that some patients do not have their event time (i.e., for a censored patient, the class label is a lower bound on the patient's actual survival time), and these censored data may bias the result of grouping. In order to see if the presence of censored observations will affect the grouping mechanism, we have chosen two approaches, where one requires class labels (*CART*) and the other does not (clustering).

4.3.1 Classification and Regression Tree

The *Classification and Regression Trees* (*CART*) is an algorithm that generates a decision tree [4]. Given a data set D, we aim to partition D into a tree of k leaf nodes (i.e., D is divided into k subsets $\{D_1, D_2, ..., D_k\}$).

In Section 4.2.1, we discussed the *regression trees* algorithm, a type of *CART* that is designed for predicting continuous class labels. Similar to the learning process of the *regression trees*, the learning process of the *CART* requires (1) a set of *partition rules*, (2) a *splitting criterion*, (3) a *stopping criterion*, and (4) a *pruning criterion*. The *CART* is different with the *regression tree* in that the *CART* is a generalization of the *regression trees* and the *CART* does not necessarily apply any regression models. Here, our goal of using *CART* is to partition D into k subsets $\{D_1, D_2, ..., D_k\}$, and our target is to obtain the set of *partition rules* R to partition D.

Splitting Criterion

Recall that the *CART* recursively splits the data node by node. At each node $Node_l$, D_l is divided into two subsets $D_{R_l,-}$ and $D_{R_l,+}$ by the best partition rule R_l among all possible partition rules in that node. (In fact, we can partition D into more than two subsets at each node, but we consider only binary split in our work.) The challenge is to find the "best" R_l and determine whether it is "worthwhile" to split D_l. We have some liberties in how we define "best" and "worthwhile". The *splitting criterion* is designed to evaluate each R_l at $Node_l$. In our work, we test two types of splitting criteria, including the *log-rank statistics* and the *gain-ratio*.

- *Log-rank statistics*

  As described in Section 2.3, the Log-rank test is a hypothesis test in survival analysis method that can quantify the difference between two risk groups. Given two populations, the null hypothesis of Log-rank test is that these two groups are drawn from a common distribution. A *p-value* close to 1 indicates that the hypothesis is likely to be true, whereas a *p-value* close to 0 indicates that the hypothesis is unlikely (i.e., these two risk groups are dissimilar).

  Radespiel-Troger et al. [46] use several types of measurements from the *log-rank statistic* as splitting criteria for their decision trees learning algorithm. In our work, we applied their idea and use the *p-value* from the *log-rank test* test as our splitting criterion for segregating patients. More precisely, given data D at each node, there exists a set of different partition rules R (i.e., different pairs of splitting attribute X^j and splitting value z^j). Each R_i in R divides D into $D_{R_i,-}$ and $D_{R_i,+}$, and then, each R_i is evaluated by the *p-value* between the associated pair of $D_{R_i,-}$ and $D_{R_i,+}$. Their algorithm then selects the R_i with the smallest *p-value* (i.e., $D_{R_i,-}$ and $D_{R_i,+}$ are most heterogeneous) among all R.

  Recall that the *p-value* of a prognostic factor less than 0.05 is considered to be significant. Therefore, we terminate the splitting process when the *p-values* of all possible partition rules are larger than 0.05, which indicates that no partition rule can effectively divide this population into smaller homogeneous groups.

  Consider our imaginary survival data that we introduced in Table 2.2 and Table 2.3 and suppose that we group patients use the *CART* with log-rank statistic as the splitting criterion on this data, Figure 4.7 shows an example result of segregation. The first table shows partial estimated *p-values* from the first split. When we group patients D by their X^{ps}, such that patients in group D_- have their $X^{ps} \leq 3.5$ and patients in another group D_+ have their $X^{ps} > 3.5$, the *p-value* between these two sub-groups is the smallest (0.02) among all splits, which indicates that these two sub-populations are most diverse. Assuming that we repeat the same splitting process for its descendant nodes (i.e., each internal node carries out similar tasks to find the best partition rule with the smallest *p-value*), in this example, these patients are segregated into 4 disjoint subsets.

X^j	Z	*P-value*
X^{gender}	0.5	0.35
X^{age}	45	0.89
⋮	⋮	⋮
X^{lung}	0.5	0.22
$X^{pancreas}$	0.5	0.13
X^{ps}	1.5	0.85
X^{ps}	2.5	0.35
X^{ps}	3.5	**0.02**
⋮	⋮	⋮

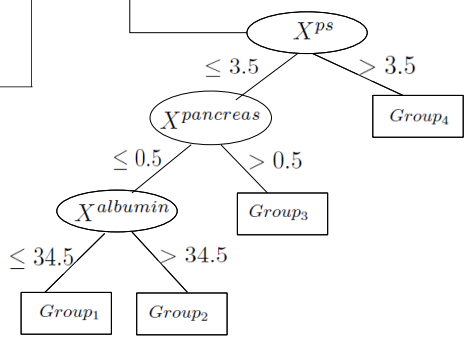


Figure 4.7: An example of using the *log-rank statistics* as the splitting criterion

- *Gain-ratio*

  Gain-ratio criterion is based on the concept of *entropy* in information theory [45]. Here, we want to use the *gain-ratio* to evaluate each possible partition rule. Given a data set of n patients D, we define $T = [t_1, t_2, ..., t_n]$ where each t_i is the class label of the i^{th} patient. In information theory, the *entropy* of D measures the average amount of information needed to identify the class label of an instance. Let $Pr(t_i)$ be the probability density of a class label t_i, the *entropy* is calculated by:

$$\textit{Entropy}(D) = -\sum_{i=1}^{n} [Pr(t_i) \times \log_2 (Pr(t_i))]$$

  In order to obtain *entropy* of a set of observations D, we need to know the probability of a class label t_i among T. The class labels T is a continuous variable, and the probability of T cannot be easily calculated. Hence, we applied the *Gaussian-kernel-density estimator* to approximate the probability density of T [43]. Letting h be a control parameter of the estimator where $h = 1$ or $h = 2$ are common choices. (There are many literature discussing the parameter h, cf. [43], so the detail is not discussed here.) The probability density of the i^{th} class label t_i can be approximated by:

$$Pr(t_i) = \frac{1}{n}\sum_{j=1}^{n} \frac{1}{\sqrt{2\pi h^2}} e^{-\frac{1}{2h^2}(t_i - t_j)^2}$$

  Consider D is partitioned into two subsets $\{D_{R,-}, D_{R,+}\}$ by a partition rule R, the expected

information needed can be calculated as the weighted sum over these two subsets:

$$\textit{Entropy}_R(D) = \frac{|D_{R,-}|}{|D|} \times Entropy(D_{R,-}) + \frac{|D_{R,+}|}{|D|} \times Entropy(D_{R,+})$$

The *information gain* measures the information that we earn by partitioning D according to a partition rule R.

$$\textit{Information Gain}(D, R) = Entropy(D) - Entropy_R(D)$$

We can use the *information gain* as our splitting criterion by taking the splitting rule that has the maximum information gain. However, this measurement tends to be biased when there are many distinct outcomes, which is often the case in medical data (e.g., consider patients' identification number, weights, blood test results, or any feature that involves continuous measurement or unique identifiers) For example, patients' identification numbers are unique in the data set. We can split on this attributes by grouping one patient into one subset and all other patients into the other subset. In this case, $\textit{Entropy}_R(D) = 0$, and $\textit{Information Gain}(D, R)$ is large. Nevertheless, using patients' id numbers is unlikely to be a useful split. Quinlan suggests that we normalize the *information gain* by the *split information*, which represents the information generated by dividing D into $\{D_-, D_+\}$ [45].

$$\textit{Split Info}\ (D, R) = -\frac{|D_{R,-}|}{|D|} \times \log_2\left(\frac{|D_{R,-}|}{|D|}\right) - \frac{|D_{R,+}|}{|D|} \times \log_2\left(\frac{|D_{R,+}|}{|D|}\right)$$

Finally, the *gain ratio* of the partition rule R is the normalized information gain:

$$\textit{Gain Ratio}\ (D, R) = \frac{Information\ Gain(D, R)}{Split\ Info(D, R)}$$

Pruning Criterion

Recall that the pruning process removes splits from the initial tree that does not contribute to better classification or regression results (Section 4.2.1). This is accomplished by comparing the *expected prediction error* of the unseen data that will be experienced at each node.

Similar to the pruning process of the *regression trees* (Section 4.2.1), this process takes a bottom-up approach, in which the process starts from the bottommost leaf and works toward to the root. At each $Node_l$, we incorporate a *3-fold cross-validation* to obtain the *average cross-validation error* using an algorithm $L(.)$ on the data D_l, denoted by $Err_{CV}(L, D_l)$ (see Algorithm 1 for details). Here, $L(.)$ can be any regression algorithms, and the *linear regression* (Section 4.2.1) is applied in our work.

The cross-validation error of a predictor on each hold-out set is evaluated by the *expected prediction error*. There are many ways to calculate the *expected prediction error*, and there is no single best measurement for the pruning process. In our work, we tested several of them in our experiments, including the *average L1 error*, the *average L2 error*, and the *average relative absolute error* (Section 3.2).

We also make some minor modification on each of the conventional measurement when censored observations are encountered. For a censored patient, since we only know a lower bound of this patient's survival time, we do not add any penalty to the error function if the predicted survival time is longer than the patient's censored time. Given a data sample D with n patients, we define that t_i to be the actual survival time of the i^{th} patient, p_i to be the predicted survival time of the i^{th} patient, c_i to be the censored flag of the i^{th} patient where 0 indicates uncensored and 1 indicates censored. The *average L1 error* is defined as $\frac{1}{n}\sum_{i=1}^{n}|p_i - t_i|$, and the *modified average L1 error* is calculated by:

$$\frac{1}{|\{i|(c_i = 0) \text{ or } (c_i = 1 \text{ and } p_i < t_i)\}|}\left(\sum_{\{i|c_i=0\}}|p_i - t_i| + \sum_{\{i|c_i=1, p_i<t_i\}}|p_i - t_i|\right)$$

By using the same idea (no penalty for a censored data if $p_i > t_i$), the *average L2 error* and the *average relative absolute error* can be modified similarly. Below is a list of our candidate functions for *expected prediction error*.

- *average L1 error*
- *average L2 error*
- *average relative absolute error*
- *modified average L1 error*
- *modified average L2 error*
- *modified average relative absolute error*

Table 4.4 shows the result of using these measurements on our example data and predictions (Table 2.3). Here, Patient_4 is a censored patient whose censored time is 18 and predicted survival time is 27. Therefore, this patient contributes no error to the *modified L1 error*, the *modified L2 error*, and the *modified relative absolute error*. The last row of Table 4.4 shows the *average L1 error*, the *average L2 error*, the *average RAE*, the *modified average L1 error*, the *modified average L2 error*, and the *modified average RAE*.

Patient_i	t_i	c_i	p_i	*L1*	*L2*	*RAE*	*modified L1*	*modified L2*	*modified RAE*
Patient_0	1	0	10	9	81	0.90	9	81	0.90
Patient_1	8	0	3	5	25	1.67	5	25	1.67
Patient_2	10	1	5	5	25	1.00	5	25	1.00
Patient_3	13	0	15	2	4	0.13	2	4	0.13
Patient_4	18	1	27	9	81	1.00	0	0	0.00
Patient_5	120	0	125	5	25	0.04	5	25	0.04
Average	-	-	-	5.83	6.27	0.79	4.33	26.67	0.62

Table 4.4: An example of using candidate error functions

4.3.2 Clustering

Clustering is an unsupervised learning algorithm, which is a common technique for grouping similar objects into smaller subsets, where each subset is called a "cluster". The goal of clustering is to partition data such that instances within a cluster are closer to one another and instances across clusters are far from each other. Several methods for clustering have been proposed, and we consider the *expectation-maximization clustering* algorithm.

Expectation-Maximization

Expectation-Maximization (EM) method is a statistical method of maximum likelihood estimation [13]. Here, we use *EM* to compute the probability density modelled as a mixture of multiple Gaussian distributions, which is accomplished by finding parameters to optimize the log-likelihood of this mixture model.

Let X be a $n \times m$ matrix that represents a set of n data point in a m-dimensional space where X^j indicates the j^{th} dimension of X and X_i indicates the i^{th} point of X, then the multivariate Gaussian density function of X is calculated by:

$$Pr(X_i, \mu, \Sigma) = \frac{1}{\sqrt{[(2\pi)^m]|\Sigma|}} e^{\left(-\frac{1}{2}(X_i-\mu)\Sigma^{-1}(X_i-\mu)^T\right)}$$

where μ is the m-dimensional mean vector of X

Σ is the $m \times m$ covariance matrix of X

$|\Sigma|$ is the determinant of Σ

Instead of modelling X using one multivariate Gaussian distribution, assuming that we can fit X as a mixture of k multivariate Gaussian distributions, the probability density of X is a weighted mixture of these k distributions:

$$Pr(X_i; \theta) = \sum_{l=1}^{k} W_l Pr(X_i; \mu_l; \Sigma_l)$$

where θ is a set of parameters,

including
- W_l is the weight of the l^{th} Gaussian distribution
- μ_l is the m-dimensional mean vector of the l^{th} Gaussian distribution
- Σ_l is the $m \times m$ covariance matrix of the l^{th} Gaussian distribution

Figure 4.8 is an example of a uni-variate mixture model. The left figure shows the histogram of data points, which does not follow a single Gaussian distribution (indicated by the blue curve). In the right figure, we can see that the data seems to fit better with two separate Gaussian distributions.

The quality of a mixture model is measured by the likelihood function — i.e., the probability that the points X is generated from k Gaussian distributions:

$$L(\theta) = \prod_{i=1}^{n} Pr(X_i; \theta)$$

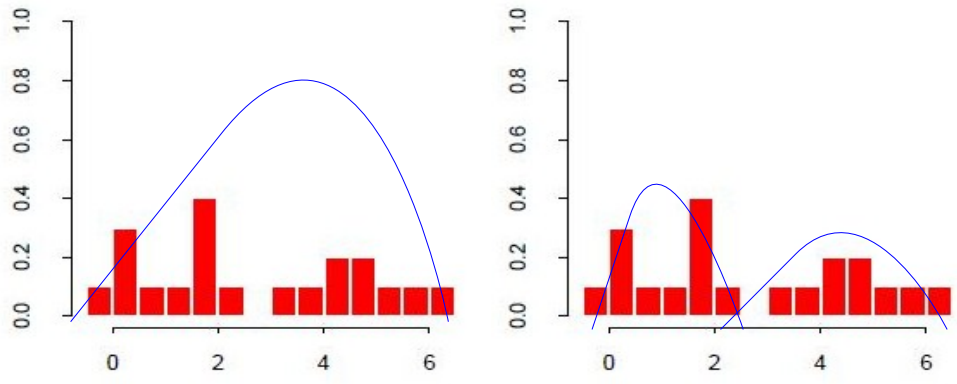


Figure 4.8: An example of a mixture model

In practice, it is more convenient to compute the logarithm of the likelihood function, log-likelihood $logL(\theta)$, calculated as:

$$logL(\theta) = \sum_{i=1}^{n} \log(Pr(X_i; \theta)) \tag{4.4}$$

As logarithmic is monotonic, the parameter θ that maximizes $logL(\theta)$ will also maximize $L(\theta)$.

EM Clustering

Given a data set D of n patients, each with m features (recall our formulation in Table 2.1), we define X as a $n \times m$ feature matrix in D and X_i as the feature values of the i^{th} patients in D, we can consider $X = \{X_1, X_2, ..., X_n\}$ to be n data points in m-dimensional space. The goal of using *EM clustering* is to group these n data points into k clusters (i.e., partition D into k subsets such that $D = \cup D_i$), where each cluster $Cluster_l$ is a multivariate Gaussian distribution:

$$Cluster_l \sim N(\mu_l, \Sigma_l) \text{ for } l \in 1, 2, ..., k$$

EM clustering optimizes the log-likelihood of this mixture model (Equation 4.4) and estimates the mean μ_l and variance σ_l of each cluster $Cluster_l$.

Figure 4.9 illustrates the concept of a mixture model. The left figure shows the data points in 2-dimensional space. The right figure shows 4 clusters where each red point indicate the centre of a cluster. Each cluster $Cluster_l$ is a Gaussian distribution with mean μ_l and variance Σ_l.

1. Determining the number of clusters k

   In our work, *EM* is used to determine the number of clusters k for the data D. *EM* employes a *5-fold cross-validation* to maximize the log-likelihood. At each fold, we starts with a single cluster and continues adding one cluster at a time until the log-likelihood of the mixture model of the hold-out set decreases.

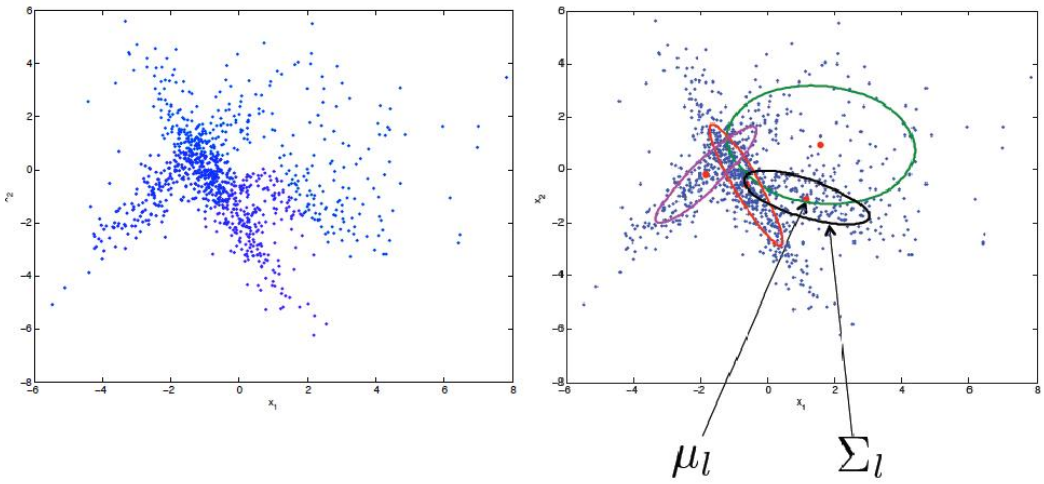


Figure 4.9: An illustration of *EM Clustering* extracted from [31]

2. Calculating the probability of belonging to each cluster

   Given a mixture model of k clusters $\{Cluster_1, Cluster_2, ..., Cluster_k\}$ and a set of n data points $X = \{X_1, X_2, ..., X_n\}$ in m-dimensional space where X_i denotes the values of the i^{th} point in X, the result of this mixture model is a set of k m-dimensional vectors of means, k $m \times m$ covariance matrices, and k scalar weights for these k clusters. From this result of the mixture model, we can then calculate a set of k probabilities for each point X_i, each measuring the likelihood that X_i belongs to the corresponding cluster. More precisely, for each X_i, the probability density under each cluster $Cluster_l$ is estimated by the Gaussian density function of that cluster:

$$\begin{aligned} Pr(X_i \in Cluster_1) &= Pr(X_i; \mu_1, \Sigma_1) \\ Pr(X_i \in Cluster_2) &= Pr(X_i; \mu_2, \Sigma_2) \\ &\vdots \\ Pr(X_i \in Cluster_k) &= Pr(X_i; \mu_k, \Sigma_k) \end{aligned}$$

3. Assigning Cluster

   Finally, we can choose the most likely cluster assignment

$$g_i = \arg_l \max Pr(X_i \in Cluster_l)$$

   for each X_i – i.e., $g_i = l$ where $Pr(X_i \in Cluster_l)$ is the highest among k clusters).

Given a data set D of n patients each with m features where X is a $n \times m$ feature matrix, X^j represents the feature values of the j^{th} feature, and X_i represents the feature values of the i^{th} patient in D, *EM clustering* is applied to group patients with "similar" feature values — i.e.,

patients whose feature values are similar have higher probability to be in the same cluster. Figure 4.10 shows an example of grouping patients using a *EM* mixture model for clustering. For ease of visualization, assuming that we only have 2 features for each patient: age and albumin — i.e., $X = (X^{age}, X^{albumin})$. We use *EM* to find 4 clusters $Cluster_{purple}$, $Cluster_{red}$, $Cluster_{green}$, and $Cluster_{black}$ (as indicated by different colours in the figure). Next, we use *EM* to learn a mixture model of 4 multivariate Gaussian distributions, and then, we can use the result of this mixture model to estimate the likelihood that a patient belonging to each of these 4 clusters. For patient Patient$_i$ with features X_i, the probability estimates are: $Pr(X_i \in Cluster_{purple}) = 0.43$, $Pr(X_i \in Cluster_{red}) = 0.30$, $Pr(X_i \in Cluster_{black}) = 0.17$, and $Pr(X_i \in Cluster_{green}) = 0.10$. In this case, Patient$_i$ is more likely to be similar to patients in $Cluster_{purple}$.

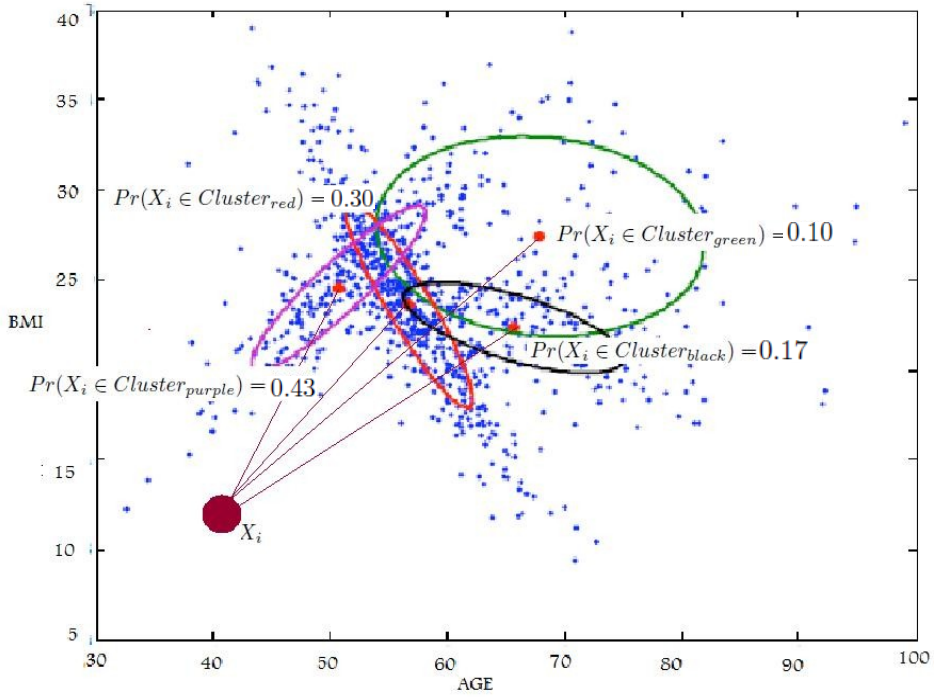


Figure 4.10: An example of *EM Clustering*

4.4 Outliers

Similar to many other machine learning problems, a challenge in this work is the presence of *outliers* — i.e., patients who are extremely different from the rest of the population. Given a data set $D = (X, T)$ where X_i denotes the feature values of i^{th} patients and X^j denotes the feature values of the j^{th} feature, an outlier is a patient whose feature values X_i are extremely different from the majority of X. For example, consider our imaginary data in Table 2.3, assuming that only two features, X^{age} and $X^{albumin}$, are available, the feature values of Patient$_3$ is much different with the feature values of other 5 patients. Figure 4.11 shows a 2-dimensional visualization of this example. We can consider the feature set of n patients each with m features, $X = \{X_1, X_2, ..., X_n\}$, are n data points in a m-dimensional space. (Here, we show only X^{age} and $X^{albumin}$ from the data set.) If we plot X^{age} on the x-axis and $X^{albumin}$ on the y-axis, we can see that Patient$_3$ is isolated from the rest of patients.

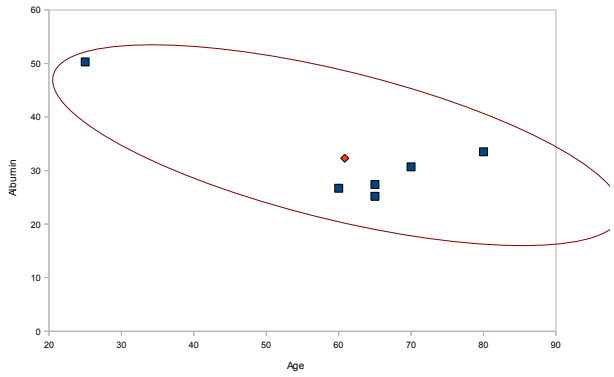


Figure 4.11: A visualization of an outlier in a 2-dimensional space

In this m-dimensional space, an outlier can be characterized as a point far away from the centre of X. Therefore, we try to detect outliers by measuring the distance between each data point X_i and the centre of X. In Figure 4.11, the red rhombus denotes the centre of X. We can see that the distance between Patient$_3$ and the centre is relatively longer than the distance between any other patient and the centre.

Outliers often exert problematic influence on the parameters and so should be excluded before training the model. For example, if we train a predictor f using the *linear regression* (Section 4.2.1) on X, we get parameters $\beta = \left[\beta^0, \beta^{age}, \beta^{albumin}\right] = [283.5865, -1.8807, -4.3604]$. Assuming that we eliminate Patient$_3$ from our data and train another predictor f', we get another parameters $\beta' = [308.1640, -10.7157, 15.7457]$. How well do f and f' perform? Table 4.5 shows the result of predictions on the same training data. After eliminating Patient$_3$, the *average relative absolute error* over the remaining patients drops from 4.14 to 2.98.

	X^{age}	$X^{albumin}$	t_i	$f(X_i)$	$f'(X_i)$	$RAE(\beta)$	$RAE(\beta')$
Patient$_0$	80	33.5	1	0.50	0.50	1.00	1.00
Patient$_1$	70	30.7	8	18.07	41.46	0.56	0.81
Patient$_2$	65	25.2	10	51.46	8.43	0.81	0.19
Patient$_4$	65	27.4	18	41.86	43.08	0.57	0.58
Patient$_5$	60	26.7	120	54.31	85.63	1.21	0.40
Average						4.14	2.98

Table 4.5: An example of experimental results with and without outliers

Detecting outliers in high-dimensional space is not trivial since one cannot rely on 2-dimensional plots nor any other visualization methods. The complexity is magnified when there are multiple outliers present because it will be harder to define "outliers" versus "majority". In our work, we attempt to eliminate outliers from our training and testing set, and we will discuss our approach of applying the *Mahalanobis distance* and the *minimum covariance determinant estimator* in this section.

4.4.1 Mahalanobis Distance

Our approach to handling outliers is to eliminate instances (patients) that are too far away from the centre (average) of the data. We measure the distance between a point and the centre using some distance-based methods such as the *Euclidean distance (ED)* and the *Mahalanobis distance (MD)* [34]. Both methods measure the distance between two points, but the *MD* differs from the *ED* in that it takes into account the correlations of the data. Given a set of n data points $X = \{X_1, X_2, ..., X_n\}$ in a m-dimensional space, the *Euclidean distance* and the *Mahalanobis distance* of each point X_i to the centre of X, μ_X (the m-dimensional mean vector of X), is calculated by:

$$ED_X(X_i) = \sqrt{(X_i - \mu_X)(X_i - \mu_X)^T}$$
$$MD_X(X_i) = \sqrt{(X_i - \mu_X)\Sigma_X^{-1}(X_i - \mu_X)^T}$$
where Σ_X is a $m \times m$ covariance matrix of X

Using the same example (X^{age} and $X^{albumin}$ from Table 2.3), we can calculate:

- The arithmetic mean of X : $\mu_X = \begin{bmatrix} \mu_X^{age} & \mu_X^{albumin} \end{bmatrix} = \begin{bmatrix} 60.8333 & 32.3000 \end{bmatrix}$
- The covariance matrix of X : $\Sigma_X = \begin{bmatrix} 354.1667 & -136.4000 \\ -136.4000 & 86.7560 \end{bmatrix}$
- The inverse covariance matrix of X : $\Sigma_X^{-1} = \begin{bmatrix} 0.0072 & 0.0113 \\ 0.0113 & 0.0292 \end{bmatrix}$

Consider Patient$_0$ whose $X_0 = \begin{bmatrix} X_0^{age} & X_0^{albumin} \end{bmatrix} = \begin{bmatrix} 80 & 33.5 \end{bmatrix}$, we can calculate the distance between X_0 and the centre of X.

$$
\begin{aligned}
MD_X(X_0) &= \sqrt{(X_0 - \mu_X)\Sigma_X^{-1}(X_0 - \mu_X)^T} \\
&= \sqrt{\begin{bmatrix} 80 - 60.8333 & 33.5 - 32.3000 \end{bmatrix} \begin{bmatrix} 0.0072 & 0.0113 \\ 0.0113 & 0.0292 \end{bmatrix} \begin{bmatrix} 80 - 60.8333 \\ 33.5 - 32.3000 \end{bmatrix}} \\
&= 1.7858
\end{aligned}
$$

Table 4.6 shows the *Mahalanobis distance* of each patient from the data set in Table 2.3. Recall that Patient$_3$ was an obvious outlier from 2-dimensional visualization (Figure 4.11); here, the result of *MD* verifies that Patient$_3$ is furthermost point from the centre.

	X^{age}	$X^{albumin}$	$MD_X(X_i)$
Patient$_0$	80	33.5	1.7858
Patient$_1$	70	30.7	0.5883
Patient$_2$	65	25.2	0.9651
Patient$_3$	25	50.3	**2.0349**
Patient$_4$	65	27.4	0.6052
Patient$_5$	60	26.7	1.0131

Table 4.6: An example of using the *Mahalanobis estimator*

4.4.2 Minimum Covariance Determinant Estimator

However, conventional *Mahalanobis estimator* suffers from *masking effect*, in which the mean and covariance are affected by the outliers. Imagine a data set consisting of a small cluster and an outlier that is far away from any other points; here, the outlier will drag the centre of the crowd in its direction. As a result, the outlier does not necessarily to have a large *MD*, and a non-outlier might have a large *MD*. For example, in Table 4.6, the *MD* of Patient$_0$ is relatively large and may be considered as an outlier from this statistical result. The problem is that Patient$_3$ drags the centre of X in its direction (see the centre of X in Figure 4.11), so that Patient$_0$ appears to be far away from the centre. However, if Patient$_3$ is excluded from the data set, Figure 4.12 shows that the centre of the data (the red diamond) is in the middle of the five remaining points, and Patient$_0$ is not far from the centre.

Rousseeuw and van Driessen proposed a robust method, the *minimum covariance determinant (MCD) estimator*, to replace the formal *Mahalanobis estimator* [47]. Given a set of n data points $X = \{X_1, X_2, ..., X_n\}$, recall that the *Mahalanobis estimator* omeasures the distance between a data point X_i and the centre of X. A key difference between these two estimators is that the *MCD estimator* considers the centre of a *subset* of X. More precisely, the *MCD estimator* finds a subset of h data points $H \subset X$ and measures the distance between a data point X_i and the centre of H.

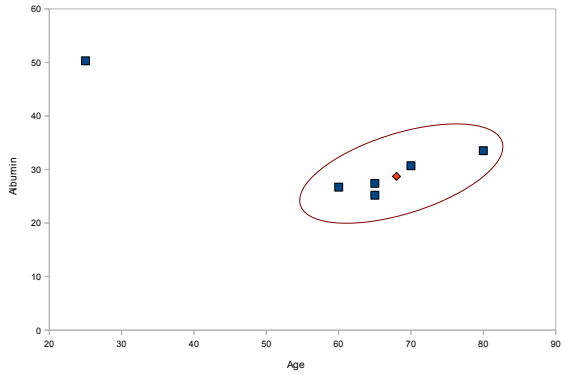


Figure 4.12: A visualization of an outlier in a 2-dimensional space

This *robust Mahalanobis distance* of each point X_i is then calculated by:

$$MD_H(X_i) = \sqrt{(X_i - \mu_H)\Sigma_H^{-1}(X_i - \mu_H)^T}$$

$$\text{where } \begin{cases} \mu_H \text{ is a } m\text{-dimensional mean vector of } H \\ \Sigma_H \text{ is a } m \times m \text{ covariance matrix of } H \end{cases}$$

Consider our example again and assuming that we can find a subset H such that $H = X - X_3$,

- The arithmetic mean of H : $\mu_H = \begin{bmatrix} 68.0000 & 28.7000 \end{bmatrix}$
- The covariance matrix of H : $\Sigma_H = \begin{bmatrix} 57.5000 & 23.0000 \\ 23.0000 & 11.2450 \end{bmatrix}$
- The inverse covariance matrix of H : $\Sigma_H^{-1} = \begin{bmatrix} 0.0956 & -0.1956 \\ -0.1956 & 0.4890 \end{bmatrix}$

The *robust Mahalanobis distance* of each data point is shown in Table 4.7. The distance between Patient$_3$ and the centre is extremely large (27.7185), while the distances between rest of the data points to the centre are relatively smaller (< 2). The result supports our assumption that the *MCD estimator* is more effective in differentiating outliers and non-outliers.

	X^{age}	$X^{albumin}$	$MD_H(X_i)$
Patient$_0$	80	33.5	1.5825
Patient$_1$	70	30.7	0.8796
Patient$_2$	65	25.2	1.6563
Patient$_3$	25	50.3	**27.7185**
Patient$_4$	65	27.4	0.4018
Patient$_5$	60	26.7	1.3480

Table 4.7: An example of using the *minimum covariance determinant estimator*

The "optimal" subset $H \subset X$ of size h is the set of data points whose covariance matrix has the smallest determinant among all possible subsets of size h. This "optimal" H is difficult to compute since it involves evaluating all $\binom{n}{h}$ subsets of size h. Therefore, Rousseeuw and van Driessen

proposed an efficient method to approximate H, which starts from an initial subset and gradually "improves" it (i.e., iteratively derives a new subset with a smaller determinant of covariance matrix) [47].

Here is a brief description on the process of deriving a new subset H_{new} from a subset H_{old}. Suppose we are given a set of data points $X = \{X_1, X_2, ..., X_n\}$ and the subset $H_{old} \subset X$, and we want to compute a subset of h data points $H_{new} \subset X$ such that the determinant of the covariance matrix of H_{new} is smaller than the determinant of the covariance matrix of H_{old}.

1. Calculate mean and the covariance matrix of H_{old}
   $\mu_{H_{old}} =$ the m-dimensional mean vector of H_{old}
   $\Sigma_{H_{old}} =$ the $m \times m$ covariance matrix of H_{old}

2. Calculate the *Mahalanobis distance* between each X_i to the centre of H_{old}
   $MD_{H_{old}}(X_i) = \sqrt{(X_i - \mu_{H_{old}})\Sigma^{-1}_{H_{old}}(X_i - \mu_{H_{old}})^T}$ for $i = 1, 2, ..., n$.

3. Sort all X_i by their *Mahalanobis distance* $MD_{H_{old}}(X_i)$
   Let $\pi = (\pi_1, \pi_2, ..., \pi_n)$ be the permutation of sorting such that
   $MD_{H_{old}}(\pi_1) \leq MD_{H_{old}}(\pi_2) \leq ... \leq MD_{H_{old}}(\pi_n)$.

4. Let $H_{new} = \{\pi_1, \pi_2, ..., \pi_h\}$
   (i.e., the set of the first h data points with the smallest *Mahalanobis distance*).

We can repeat the same process until H_{old} and H_{new} converge. ($det(\Sigma_{H_{new}}) \leq det(\Sigma_{H_{old}})$ with equality if and only if $\mu_{H_{new}} = \mu_{H_{old}}$ and $\Sigma_{H_{new}} = \Sigma_{H_{old}}$.)

Rousseeuw and van Driessen also suggests a method to produce an initial subset H_1. Let H_1 be a set of j randomly-selected data points in X where $j < h$. Then H_1 is extended by sequentially adding a randomly-selected observation, one at a time, until the determinant of the covariance matrix of H_1 is strictly greater than 0.

Given a feature matrix X, Algorithm 6 shows the pseudocode for computing the subset H of the *MCD estimator*:

Algorithm 6 Subset H Approximation

Input: X: $n \times m$ feature matrix

Output: H: $h \times m$ feature matrix (subset of the rows of X)

$k = 1$

$H_k \leftarrow \{j$ randomly-selected data points in $X\}$

repeat

  $H_k \leftarrow H_k \cup \{$ a randomly-selected point in $X_i\}$

until $det(\Sigma_{H_k}) > 0$

repeat

  for $i = 1$ to n **do**

    $MD_{H_k}(X_i) \leftarrow \sqrt{(X_i - \mu_{H_k})\Sigma_{H_k}^{-1}(X_i - \mu_{H_k})^T}$

  Sort X by $MD_{H_k}(X_i)$ such that $MD_{H_k}(\pi_1) \leq MD_{H_k}(\pi_2) \leq ... \leq MD_{H_k}(\pi_n)$

  $\pi \leftarrow (\pi_1, \pi_2, ..., \pi_n)$

  $H_{k+1} \leftarrow \{\pi_1, \pi_2, ..., \pi_h\}$

until $|det(\Sigma_{H_k}) - det(\Sigma_{H_{k+1}})| < \epsilon$

return H_{k+1}

4.5 Preprocessing

Medical data is a large set of real world information, which usually includes a mixture of binary, discrete, and continuous variables. In many machine learning problems, how information is formulated and what features are included are critical to the quality of the learned predictors.

Given a large set of medical records of patients' personal attributes, clinical assessments, blood test results, survival times, and censored times, we aim to process these records to produce a data set $D = (X, T)$ with feature values X and class labels T, such that X optimally characterizes T. Here, we will discuss our feature representation in Section 4.5.1, our feature selection methods in Section 4.5.2, and our class label representation in Section 4.5.3.

4.5.1 Feature Representation

The task of survival prediction begins by pre-processing the original survival data, and our first step is to design an appropriate data representation with some domain specific knowledge so that the pre-processed data is usable for our prediction framework. Survival data usually contains categorical features. For example, "Gender" is either "male" or "female", and "Cancer Type" can be "lung cancer", "pancreas cancer", or "colon-rectal cancer" (see the top table in Table 4.8). For the algorithms that we consider in this framework (e.g., regression methods, clustering methods, etc), it is relatively easier to use only numerical features.

Patient_i	Gender	Age	Cancer Type	...	Event Time	Censored Flag
Patient_0	male	60	pancreas	...	1	0
Patient_1	female	70	pancreas	...	8	0
Patient_2	female	65	lung	...	10	1
Patient_3	male	35	lung	...	13	0
Patient_4	female	80	pancreas	...	18	1
Patient_5	female	75	pancreas	...	120	0
Patient_6	male	72	colon-rectal	...	125	1

$\Downarrow$

Patient_i	X^{gender}	X^{age}	X^{lung}	$X^{pancreas}$	$X^{colon-rectal}$	...	t_i	c_i
Patient_0	0	60	0	1	0	...	1	0
Patient_1	1	70	0	1	0	...	8	0
Patient_2	1	65	1	0	0	...	10	1
Patient_3	0	35	1	0	0	...	13	0
Patient_4	1	80	0	1	0	...	18	1
Patient_5	1	75	0	1	0	...	120	0
Patient_6	0	72	0	0	1	...	125	1

Table 4.8: An example of pre-processed and post-processed survival data

If a categorical attribute has only two distinct categories, we can convert this categorical attribute into a binary attribute by defining one category to be 0 and the other category to be 1. For example, in Table 4.8, "Gender" is one of the characteristic of patients that contains two categories, "male" and "female". Without loss of generality, we can denote "male" as 0 and "female" as 1.

If a categorical feature has more than 2 attribute values, this feature can be converted into multiple binary features. For instance, in Table 4.8, "Cancer Type" (feature values of the cancer types of patients) has three attribute values: "pancreas", "lung", and "colon-rectal". In our data representation, "Cancer Type" is expanded into three binary features X^{lung}, $X^{pancreas}$, and $X^{colon-rectal}$, and "Cancer Type" is deleted from the data set.

4.5.2 Feature Selection

Some features may be irrelevant to survival times — e.g., we expect that toothache will not matter. To improve the performance of our predictors, we attempt to eliminate irrelevant features from our raw data. In machine learning, many sophisticated methods have been applied to identify irrelevant features; here, we consider the following two representative feature selection approaches: *subset selection by backward wrapper* and *variable ranking by mutual information*.

Subset Selection by Backward Wrapper

Given a data set $D = (X, T)$ where X represents the feature values in D, letting A be the set of features in X, the *subset selection* seeks the "best" subset of features $A^* \subset A$ over many combinations of subsets from A [32].

The *wrappers* method is a *subset selection* method that finds A^* by comparing the performance of a predictor on different subsets of A. Given a learning algorithm $L(.)$, we seek a subset $A^* \subset A$ such that the *average CV error* (Section 3.1) of $L(.)$ is minimized. Since it is computationally too complicated to evaluate all combinations of subsets, the *greedy search* is incorporated to approximate the optimal solution. This search algorithm iteratively generates a new candidate subset of attributes $A' \subset A$ and evaluates the predictive power of $L(.)$ on $D' = (X', T)$ where X' is the feature values of attributes A'.

There are two "directions" on how to generate a new candidate subset: the *backward elimination* and the *forward selection*. The *backward elimination* begins with the whole feature set A and sequentially excludes one feature at a time. At each iteration, we aim to eliminate the feature (among all available features in that iteration) such that the *average CV error* is the lowest after removing that feature. A similar approach is the *forward selection*, which attempts to select a subset of features in an incremental manner. More details on these two approaches can be found in [24]. In our work, we consider the *backward elimination*. Algorithm 7 shows the procedure of the *subset selection by backward wrapper* algorithm.

Variable Ranking by Mutual Information

In *variable ranking*, the goal is to rank the "relevance" between each feature and the class label and eliminate features that are low in their "rankings" [17]. Here, we need to define a ranking criterion to evaluate the "relevance" between each pair of feature and class label. In our work, we

Algorithm 7 Subset Selection by Backward Wrapper Algorithm

Input:

D: a data set with a set of features A, feature values X, and class labels T

$L(.)$: a learning algorithm

Output:

A_k: the approximated optimal subset of features

$k \leftarrow 1$
$A_k \leftarrow A$
$X_k \leftarrow X$
$score_k \leftarrow Err_{CV}(L, (X_k, T))$
repeat
  for $i = 1$ to $|A_k|$ **do**
    $X_i \leftarrow X_k$ with the i^{th} feature values removed
    $score_i \leftarrow Err_{CV}(L, (X_i, T))$
  $i^* \leftarrow \arg\min Err_{CV}\{score_i\}$
  if $score_{i^*} < score_k$ **then**
    $A_{k+1} \leftarrow A_k$ with the i^* feature removed
    $X_{k+1} \leftarrow X_k$ with the i^* feature values removed
    $score_{k+1} \leftarrow score_{i^*}$
    Increment k
until stopping criterion is met
return A_k

consider using the *mutual information*, which measures the information that two variables share, as our ranking criterion.

Recall our formulation in Table 2.1, given a data set $D = (X, T)$ of n observations each with m features where X is the $n \times m$ feature matrix, T is the n-dimensional vector of the class labels, X^j denotes the j^{th} feature in X, x_i^j denotes the feature value of the j^{th} feature of the i^{th} patient, and t_i denotes the class label of the i^{th} patient, the *mutual information* between X^j and T is defined as

$$MI(X^j, T) = \sum_{x \in X^j} \sum_{t \in T} Pr(x, t) \log \left(\frac{Pr(x, t)}{Pr(x)Pr(t)} \right)$$

where $Pr(x, t)$ is the joint probability density of x and t

$Pr(x)$ is the marginal probability density of x

$Pr(t)$ is the marginal probability density of t

The *mutual information* quantifies the dependence between X^j and T. If X^j and T are independent, their joint probability density $Pr(X^j, T) = Pr(X^j)Pr(T)$, and so their *mutual information* $MI(X^j, T) = 0$

This quantity is hard to compute when one of the variables is continuous; in our survival data, the class labels T is a continuous variable and the feature values X^j is either a discrete or a continuous variable. Here, we apply some density estimation methods to approximate the marginal probability densities.

Similar to the idea that we used to approximate the probability density of the class labels T when

calculating the *entropy* in Section 4.3.1, the marginal probability density and the joint probability density can be approximated by the *Gaussian kernel estimator* [43] . Given a data set $D = (X, T)$ of n patients, we approximate the marginal probability density of x by

$$Pr(x) = \frac{1}{n}\sum_{k=1}^{n} \frac{1}{\sqrt{2\pi h^2}} e^{-\frac{1}{2h^2}(x-x_k^j)^2}$$

Here, h is a control variable of this estimator (see [43]). Similarly, the marginal probability density of the class label of the k^{th} patient can be approximated by

$$Pr(t) = \frac{1}{n}\sum_{k=1}^{n} \frac{1}{\sqrt{2\pi h^2}} e^{-\frac{1}{2h^2}(t-t_k)^2}$$

The joint probability density of the feature value x and the survival time t can be estimated by:

$$Pr(x, t) = \frac{1}{n}\sum_{k=1}^{n} \frac{1}{2\pi h^2} e^{-\frac{1}{2h^2}[(x-x_k^j)^2+(t-t_k)^2]}$$

After simplification, the *mutual information* can be approximated by:

$$\widehat{MI}(X^j, T) = \frac{1}{n}\sum_{i=1}^{n} \log \frac{n\sum_{k=1}^{n} e^{-\frac{1}{2h^2}[(x_i^j-x_k^j)^2+(t_i-t_k)^2]}}{\sum_{k=1}^{n} e^{-\frac{1}{2h^2}(x_i^j-x_k^j)^2}\sum_{k=1}^{n} e^{-\frac{1}{2h^2}(t_i-t_k)^2}}$$

4.5.3 Log-Space Transformation

In survival analysis, an *accelerated failure time (AFT)* model is a parametric model that assumes that the the effect of a variable is linearly related to the **logarithm** of the survival time [3]. More precisely, given a data set of n patients each with m features, let T be the n-dimensional vector of survival times and X be the $n \times m$ feature matrix, the *AFT* model assumes that

$$\log(T) \approx \beta_0 + \beta X$$

where β is a m-dimensional vector of parameters and β_0 is a scalar parameter. Ying et al. [56] applied the *linear regression* to learned the parameters β and β_0 for their *AFT* model that predicts the probability of survival for individual patients.

In our work, we apply the ideas of the *AFT* and the above work to our survival prediction problem. Here, we simply transform T into logarithmic space — i.e., use $\log(T)$ as the class labels for training our predictors. To test the effectiveness of this logarithmic assumption, we experimentally compare the performance of predictors before and after this log-space transformation. That is, for each learning algorithm that we discussed in Section 4.2, we will compare the performance of using T versus using $\log(T)$ as the class labels. To distinguish these cases, we refer the resulting predictors as the *regular (REG)* predictor and the *logarithmic (LOG)* predictor.

Chapter 5

Experiments

In this work, we aim to find the best combination of techniques that we discussed in the previous chapter (Chapter 4) for our survival prediction system. This chapter summarizes our experiments on applying these methodologies to various prediction tasks on real data. In Section 5.1, we will describe our experimental setups, including our data set, our implementations, and our evaluation methods. In Section 5.2, we will list some experimental outcomes that we obtained with our survival prediction system. (Detailed experimental results are provided in Appendix B.) In Section 5.3, we will discuss what we learned from our experimental results.

5.1 Experimental Setups

In this section, we will describe our experimental setups, including the description of our testing data, the methods on examining our approaches, and the evaluation of our experimental results.

5.1.1 Data Set

Our data set was based on data generously provided by the Cross Cancer Institute. This data set contains 2402 patients, in which 1142 patients were uncensored and 1260 were censored. (See Section 2.2.1 for information about *censoring*) The event times (survival times or censored times) of these patients range from 0.03 to 71.95 months. The average event time among all patients (both censored and uncensored) is 25.74 ± 19.10 months.

For each patient, we have three broad categories of data, which we refer to as the personal attributes, clinical assessments, and blood test results. The personal attributes include gender, BMI, date of birth, date of death, date of reference, etc. The clinical assessments include fifteen diagnosis results such as whether the patient feels full, has no appetite, experiences problem with swallowing, etc. The clinical assessments also include a measurement called the *physician global assessment (PGA)*, which is a widely used guideline in clinical trials. The *PGA* of a patient measures the overall performance status of this patient, and this measurement is scored by a physician (based on his/her impression after inspecting this patient). The blood test results include nine numerical measurements, such as LDH serum, white blood cell counts, etc. The list of features in the raw data is provided in Appendix A.1

However, this data set is missing numerous entries, so we selected only the attributes that are missing relatively few values for incorporation into our data set. Also, we perform some pre-processing work on the raw data so it is usable for our framework (Section 4.5). The resulting pre-processed data contains 46 features; Appendix A.1 provides a list of descriptions and the histograms for these features.

5.1.2 Methods

Our survival prediction system is implemented using the Java programming language on the developmental software Eclipse. We incorporate Waikato Environment for Knowledge Analysis (Weka),

a machine learning framework developed by the University of Waikato, into our implementation [19]. Weka contains a collection of machine learning algorithms for data mining tasks, and many algorithms can be applied and modified directly from Eclipse platform.

Recall that our learning phase has two major steps. In the first step, we apply various grouping methods to segregate patients into smaller populations. In the second step, we apply different regression models to each sub-population that we obtained from the first step. Besides these two steps, we also analyze several minor techniques that we discussed in Chapter 4, such as outlier detection methods, imputation methods, and log-space transformation methods. We use experiments to select the best combination of techniques as our final model.

For the convenience of identifying each model, we refer these models by the mechanisms applied on them such that each name is formatted as *AA-BBB-CCC-DDDDDD-EEEE*. For each model, AA $\in$ { NG, LR, GR, EC } indicates its grouping method (here, NG means no grouping), BBB $\in$ { REG, LOG } indicates whether log-space transformation is applied, CCC $\in$ { ALL, MCD } indicates whether outliers are eliminated, DDDDDD $\in$ { AVE050, AVE100, CEN050, CEN100 } shows the methods on handling censored data, and EEEE $\in$ { MED, AVE, LIN, SVR, RT, LINc, SVRc, GAT } specifies the learning algorithm. For example, LR-REG-MCD-AVE050-LINc refers to the learning system that (1) segmented the patients using *CART* with *log-rank statistics* as the splitting criterion, (2) used the original data (not the log-transformed version), (3) eliminated the potential outliers using the *MCD estimator*, (4) set the class labels of censored data to be the average survival times of uncensored patients in the risk set, (5) weighted each censored patient as 50% of uncensored patient, and (6) return a predictor learned using the *linear regression for censored targets*. The definition of each combination can be found in Appendix B.1.

In our work, our goal is to make predictions for *new* patients — i.e., it is crucial that our models perform well on *novel* data. To approximate the performance of each model on such unseen data, we run *5-fold cross-validation* in each experiment, and report the average (and variance) of these five folds to analyze their performance.

5.1.3 Evaluation

In Section 3.2, we discussed several methods for evaluating the performance of a model and how to estimate these measurements. In Section 3.2.5, we define the best model to be the combination of techniques that achieves the minimum *average relative absolute error*. If the *RAE* of two models are equivalent (i.e., the difference is not statistically significant), we seek the model that has higher *concordance index*.

We also examine the *relative absolute error* within *95% confidence interval* (see Section 3.2.5 for details and examples). Our studies evaluate all patients in the unseen data set; we found, however, that the presence of a few outliers could skew our statistical analysis. In order to determine whether or not a predictor is significantly affected by the outliers, we also consider the RAE^{95} score, which

examines the testing cases lie within the 95% *confidence interval*.

In addition, we can also separate the evaluation of small predicted times versus large predicted times. Recall that the relative error is not completely fair since it exerts large penalty to smaller predictions (see Section 3.2.2 for details and examples). Therefore, for each experiment, we also look at the *average L1 error* of predictions less than 12 months and the *average relative absolute error* of predictions greater than or equal to 12 months

Through out this section, we will use the following statistics to interpret our experimental results:

- *RAE*: the *average relative absolute error*
- *CI*: the *concordance index*
- RAE^{95}: the *average relative absolute error* within the 95% *confidence interval*
- $L1_{p<12}$: the *average L1 error* of predictions less than 12 months
- $RAE_{p\geq 12}$: the *average relative absolute error* of predictions greater than or equal to 12 months

Also, recall that we need to know the actual survival times to calculate the *RAE* and the *L1*, and plot the visualization, which are not available for censored data; and therefore, the *RAE*, the RAE^{95}, the $L1_{p<12}$, the $RAE_{p\geq 12}$, and the visualizations shown in this section apply to **uncensored** data only. The *concordance index* considers all comparable pairs of patients, and therefore considers both uncensored and censored data are included. (Although, of course, it ignores the incomparable pairs — e.g., when both are censored, etc.). For most of the results, we will focus on the *RAE* score; we will mention the other measures in Section 5.3.

Besides statistical results, we will also evaluate the outcomes by visualizing the plot of actual survival times versus the predicted survival times over a set of test patients (Section 3.3).

5.2 Experimental Results

This section describes and summarizes our experimental results on different combinations of techniques. The first objective is to test whether it is advantageous to segregate patients in predicting survival times for cancer patients. The second goal is to discover which learning algorithms can construct the best predictors for our data. Also, we test our approaches to handling censored data, eliminating outliers, and log-space transformation. Finally, we will find the best combination of methods that can best predict survival times for individual patients.

5.2.1 Baseline

The naive baseline is to take the median or the average of the class labels over the whole population as the prediction for each individual patient. Recall that our data contains censored observations — i.e., the class label of a censored patient is the lower bound of the patient's actual survival time. In this experiment, we simply treat censored patients as uncensored and use their censored times as

survival times. Notice that this idea is problematic! Later on, we will use this baseline measurement to test if our techniques can improve the performance of predictors. Table 5.1 shows the performance of the baseline.

Grouping Patients: NG (no grouping) Log-space transformation: No Outlier Detection: None Handling Censored Data: No					
Regressor	*RAE*	*CI*	RAE^{95}	$LI_{p<12}$	$RAE_{p>12}$
NG-REG-ALL-CEN100-MED	0.5891±0.01 [1]	0.5000±0.00	0.5638±0.01	Undefined	0.5891±0.01
NG-REG-ALL-CEN100-AVE	0.6081±0.00	0.5000±0.00	0.5875±0.00	Undefined	0.6081±0.00

Table 5.1: Experimental results on the *baseline*

Although one can argue that the *RAE* is within reasonable range, the *concordance index* shows that this model is not effective. (Recall that $CI = 0.5$ means the predictor is not better than random guessing) Also, $LI_{p<12}$ is undefined since these two models never predict a time shorter than 12 months. Figure 5.1 plots true survival times (blue) versus predicted survival times (red), which shows that these predictions (the horizontal line) are not precise enough for individual patients.

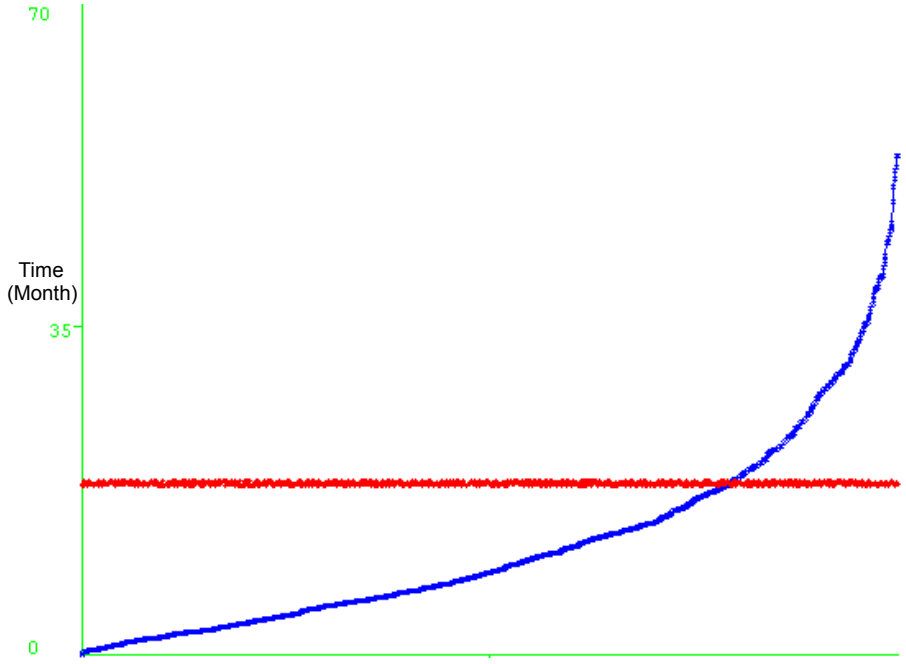


Figure 5.1: A Visualization of true survival times versus predicted survival times

5.2.2 Handcrafted Tree

In Section 2.1, we discussed some general medical research methods. Many clinical research projects analyzes two populations of patients, where the populations were split on one feature based

[1] Here and in subsequent tables, this value refers to the standard deviation of 5 folds (in a 5-fold cross-validation).

on the researchers' prior knowledge (NOT from learning). In Section 2.3.1, we also introduced the most popular evaluation in survival analysis, the *Kaplan-Meier estimator* [27], which measures the performance of a prognostic factor. How well do those methods work for our task, of estimating a survival time for each individual patient?

Since we are not medical experts, we have no prior knowledge on how to split the populations. Therefore, we decide to be generous by trying all features (with various splitting values if the feature is a continuous variable) and evaluating the resulting predictor using the best split. Notice that this is "cheating" because we use testing data to determine the best feature (i.e., it is what an all-knowledgeable person would do, if he/she was constrained to only one feature). Since prior knowledge is not necessarily the best guide to segregate patients, the predictor in this experiment should perform better or equivalent to the predictors built upon prior knowledge.

Table 5.2 shows the result of the best split, where patients with their *physician global assessment (PGA)* $= 4$ are in one group, and the rest of patients are in another. We can see that the *RAE* and *concordance index* both suggest that these predictions are no better than simply use the average of the entire population.

Grouping Patients: Handcraft Log-space transformation: No Outlier Detection: None Handling Censored Data: No					
Regressor	*RAE*	*CI*	RAE^{95}	$LI_{p<12}$	$RAE_{p>12}$
KM-REG-ALL-CEN100-MED	0.5942±0.01	0.5090±0.01	0.5649±0.01	4.8967±1.08	0.5863±0.01
KM-REG-ALL-CEN100-AVE	0.6067±0.00	0.5090±0.01	0.5859±0.00	Undefined	0.6067±0.00

Table 5.2: Experimental results on a handcrafted tree using the *Kaplan-Meier estimator*

5.2.3 Learning Algorithms

In Section 4.2.1, we discussed three basic regression methods:

LIN : the *linear regression*

SVR: the *support vector regression*

RT : the *regression trees*

Grouping Patients: NG (no grouping) Log-space transformation: No Outlier Detection: None Handling Censored Data: CEN (Use censored times as the class labels)					
Regressor	*RAE*	*CI*	RAE^{95}	$LI_{p<12}$	$RAE_{p>12}$
NG-REG-ALL-CEN100-LIN	0.5592±0.02	0.7442±0.01	0.5220±0.02	4.2071±0.25	0.5403±0.01
NG-REG-ALL-CEN100-SVR	0.5582±0.03	0.7434±0.01	0.5287±0.03	4.9575±0.41	0.5434±0.02
NG-REG-ALL-CEN100-RT	0.5601±0.02	0.7483±0.02	0.5266±0.01	4.7212±0.81	0.5438±0.01

Table 5.3: Experimental results on conventional learning algorithms

In this experiment, the goal is to test the difference between conventional regression methods and the models that simply use the median or mean survival time (such as our baseline models); no other

techniques is applied and the class labels of censored data are the censored time. Table 5.3 shows the result of each learning algorithm. The differences between each pair of learning algorithms are not statistically significant, but there is a significant improvement on *RAE* and *concordance index* when comparing each predictor to the baseline model ($p < 0.05$ by paired t-test).

Figure 5.2 shows the visualization of true survival times versus predicted survival times (over the uncensored patients) of a predictor learned from the *support vector regression*. (Since the difference between the results of these models are not statistically significant, and the visualization plots are similar, we only display one of them)

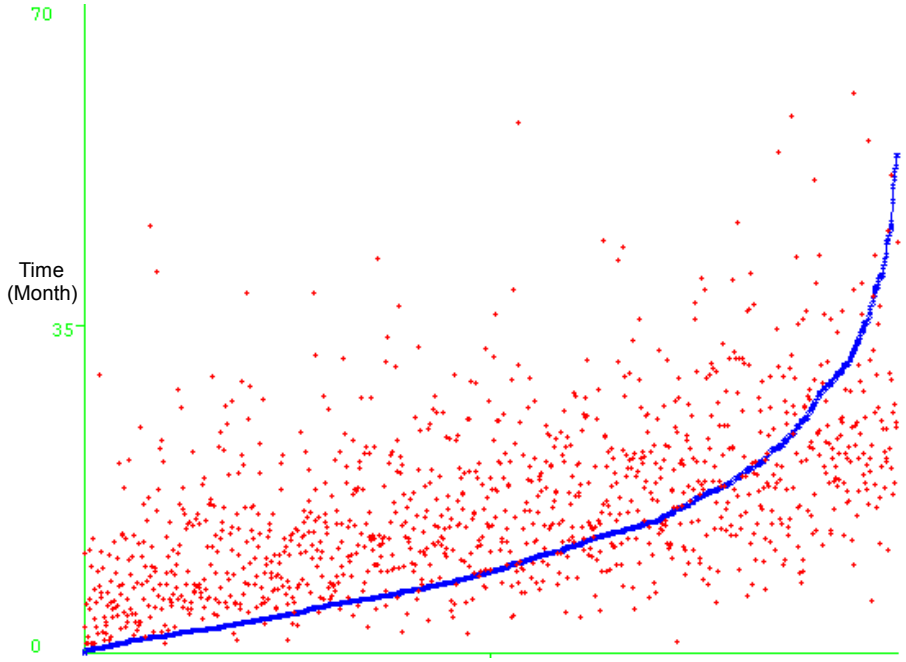


Figure 5.2: A Visualization of true survival times versus predicted survival times

5.2.4 Handling Censored Data

Recall that our data contains censored observations, where the class label of a censored patient is the lower bound of the patient's actual survival time. The result of the last experiment is problematic since it simply treats the censored times as event times. In Section 4.1, we discussed three techniques for approximating an event time for each censored patient. In this experiment, the estimated survival time of each censored patient is approximated as the average survival time of all uncensored patients in the risk set, and *LIN*, *SVR*, and *RT* use these revised labels for training its predictor.

We introduced two learning algorithms that were modified to accommodate censored data: *LINc* (the *linear regression for censored targets*) and *SVRc* (the *support vector regression for censored targets*) in Section 4.2.2. We also discuss a learning algorithm that automatically select the best learning algorithm, the *gating regression*, in Section 4.2.3. From now on, we will include all six

learning algorithms for each experiment.

Table 5.4 summarizes the results of our six learning algorithms after handling censored data. Although there is no reduction in RAE_{ave}, *RT* and *LINc* have improved *CI* score (but not statistically significant).

Grouping Patients: NG (No Grouping) Log-space transformation: No Outlier Detection: None Censored Data: use estimated survival times, weight = 0.5					
Regressor	*RAE*	*CI*	RAE^{95}	$Ll_{p<12}$	$RAE_{p>12}$
NG-REG-ALL-AVE050-LIN	0.5524±0.02	0.7498±0.01	0.5102±0.02	4.1004±0.22	0.5257±0.01
NG-REG-ALL-AVE050-SVR	0.5538±0.04	0.7493±0.01	0.4980±0.02	4.1903±0.39	0.5152±0.02
NG-REG-ALL-AVE050-RT	0.5630±0.04	0.7517±0.01	0.5225±0.02	5.1055±1.06	0.5281±0.02
NG-REG-ALL-CEN100-LINc	0.5589±0.02	0.7445±0.01	0.5215±0.02	4.1591±0.29	0.5405±0.01
NG-REG-ALL-CEN100-SVRc	0.5751±0.03	0.7499±0.01	0.5409±0.03	4.6095±0.54	0.5590±0.03
NG-REG-ALL-AVE050-GAT	0.5622±0.02	0.7448±0.01	0.5152±0.02	4.5236±0.46	0.5274±0.02

Table 5.4: Experimental results on handling censored data

5.2.5 Grouping

We use three different grouping approaches in this work, including

LR: the *CART* with the *log-rank statistics* as its splitting criterion (Section 4.3.1)

GR: the *CART* with the *gain-ratio* as its splitting criterion (Section 4.3.1)

EC: the *EM clustering* (Section 4.3.2)

To evaluate the performance of our grouping approaches, we need to first consider how to predict the survival time for each patient. Here, we first place the patient into a sub-population, then predict that the patient will live the mean (or median) of that sub-population. More precisely, during the training process, a subset of patients arrive at each leaf node; we then set the "label" of that node as the mean (or median) of the survival values of those patients. Then, to predict the survival time for a new patient, we first drop this patient into the tree to determine the relevant leaf node, and assign that patient to the label of that node.

Figure 5.3 shows a resulting decision tree produced by the *CART* with the *log-rank statistics* as the splitting criterion. Here, the population is divided into three sub-populations: Group 1, Group 2, and Group 3. The population is first divided by the *physician global assessment (PGA)* such that patients whose *PGA* is 4 are one sub-population (Group 3). The rest of patients are further divided as pancreas cancer patients (Group 2) versus other cancer patients (Group 1). For each sub-population, the number of patients and the mean survival time is shown in its corresponding leaf node. More resulting trees are provided in Table B.4 in Appdendix B.

Table 5.5 shows the results of the *CART* with the *log-rank statistics* and the *gain-ratio* as its splitting criterion. Unfortunately, using this setting is not better than using regression algorithms without grouping.

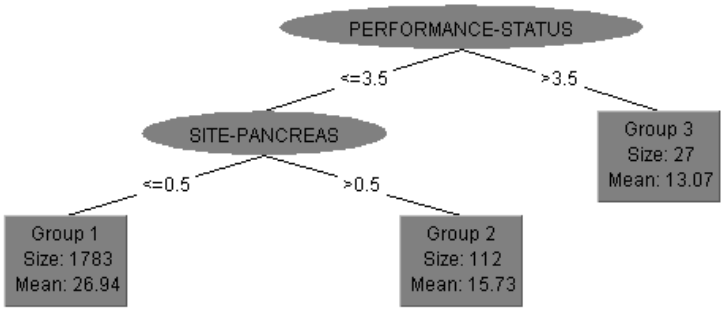


Figure 5.3: Examples of decision trees split by the *CART* with the *log-rank statistics* as the splitting criterion

Log-space transformation: No Outlier Detection: None Censored Data: AVE(use estimated survival times), weight = 0.5					
Regressor	*RAE*	*CI*	RAE^{95}	$LI_{p<12}$	$RAE_{p>12}$
LR-REG-ALL-AVE050-MED	0.6019±0.01	0.5251±0.01	0.5693±0.01	-	0.6019±0.01
LR-REG-ALL-AVE050-AVE	0.6120±0.01	0.5252±0.01	0.5887±0.01	-	0.6120±0.01
GR-REG-ALL-AVE050-MED	0.5897±0.00	0.5576±0.02	0.5642±0.00	-	0.5897±0.00
GR-REG-ALL-AVE050-AVE	0.6075±0.00	0.5823±0.01	0.5873±0.00	-	0.6075±0.00

Table 5.5: Experimental results on grouping methods

5.2.6 Combination of CART and Regressions

In this experiment, we first group the patients using the *classification and regression trees* with the *log-rank statistics* as the splitting criterion. At each node of the resulting decision tree, we consider applying our six learning algorithms. Table 5.6 shows the result of each of these learners.

Again, there exists no statistical difference between different learning algorithms in this experiment. After applying this grouping method, unfortunately, we did not get any improvement. On the other hand, the result in this experiment is better than the predictor that simply uses the mean survival time of a sub-population as a prediction for each patient in that sub-population. By using paired t-test on each regressor against the predictors that use median survival time or average survival time, the models trained using the *linear regression* , *regression trees*, em linear regression for censored targets, and *gating regression* are statistically better with $p < 0.05$.

5.2.7 Outliers Detection

Can we do better? Recall that *outliers* are patients who are extremely different from the majority; these outliers often exert problematic affect on the resulting models. In this experiment, we applied

Grouping Patients: LR (*CART* with *log-rank statistics*) Log-space transformation: No Outlier Detection: None Censored Data: AVE (use estimated survival times), weight = 0.5					
Regressor	*RAE*	*CI*	RAE^{95}	$L1_{p<12}$	$RAE_{p>12}$
LR-REG-ALL-AVE050-LIN	0.5598±0.02	0.7421±0.01	0.5168±0.02	4.1554±0.32	0.5307±0.01
LR-REG-ALL-AVE050-SVR	0.5746±0.04	0.7400±0.01	0.5107±0.01	4.3420±0.19	0.5157±0.03
LR-REG-ALL-AVE050-RT	0.5679±0.02	0.7470±0.02	0.5265±0.02	5.1883±0.60	0.5297±0.02
LR-REG-ALL-CEN100-LINc	0.5618±0.02	0.7410±0.01	0.5252±0.02	3.9586±0.41	0.5480±0.01
LR-REG-ALL-CEN100-SVRc	0.5907±0.03	0.7424±0.01	0.5404±0.02	4.4074±0.40	0.5465±0.02
LR-REG-ALL-AVE050-GET	0.5705±0.01	0.7476±0.02	0.5213±0.01	4.6896±0.24	0.5300±0.02

Table 5.6: Experimental results on combining *CART* with *log-rank statistics* and regressions

the outlier detection and elimination techniques that we discussed in Section 4.4 to remove potential outliers from both training and testing data. That is, we choose to tell some patients that he/she cannot be evaluated. In average, this meant removing around 3% of the testing samples.

To estimate the RAE^{95}, we first eliminate 3% instances that have relatively larger *Mahalanobis distance* and then remove 5% instances from the remaining 97% that have the worst *relative absolute error*. Notice that this two mechanisms are different. We do not know whether or not the removed outliers have higher prediction error (we could have made adequate predictions for them). In contrast, we know that our predictor makes less decent predictions for the instances outside of the confidence interval. Therefore, the removal process of confidence interval is not as practical as that of the outlier detections since we do not know whether or not a **new** patient is in the confidence interval at the time of prediction.

Table 5.7 summarizes the result after applying our six learning algorithms and eliminating outliers at each node of the decision tree. Even through the improvement is not statistically significant when compared to the result that does not eliminate outliers, we will keep this setting (i.e., removing outliers from both training data and testing data).

Grouping Patients: NG (No Grouping) Log-space transformation: No Outliers Detection: MCD (*Mahalanobis Distance* with *MCD estimator*) Censored Data: AVE (use estimated survival times), weight = 0.5					
Regressor	*RAE*	*CI*	RAE^{95}	$L1_{p<12}$	$RAE_{p>12}$
NG-REG-MCD-AVE050-LIN	0.5506±0.03	0.7498±0.01	0.5085±0.02	4.1940±0.43	0.5227±0.02
NG-REG-MCD-AVE050-SVR	0.5503±0.04	0.7501±0.01	0.4953±0.02	4.1312±0.40	0.5125±0.03
NG-REG-MCD-AVE050-RT	0.5619±0.03	0.7552±0.01	0.5180±0.02	5.2004±0.66	0.5175±0.02
NG-REG-MCD-CEN100-LINc	0.5506±0.02	0.7518±0.01	0.5191±0.01	4.1870±0.34	0.5355±0.01
NG-REG-MCD-CEN100-SVRc	0.5766±0.02	0.7530±0.01	0.5477±0.01	4.8107±0.38	0.5621±0.01
NG-REG-MCD-AVE050-GET	0.5556±0.02	0.7502±0.01	0.5107±0.02	4.9558±0.31	0.5125±0.02

Table 5.7: Experimental results on regression methods, after eliminating 3% outliers

5.2.8 Log-space Transformation

In Section 4.5.3, we discussed the idea of log-space transformation, such that each class label t_i is replaced by $\log t_i$. Table 5.8 shows the results after using log-transformation, and here, we show the results before and after grouping by *CART* with *log-rank statistics*. By comparing to Table 5.4

and Table 5.6, there is a significant improvement on both the RAE and the *CI* after using log-space transformation. The predictors learned using the *linear regression*, the *support vector regression*, and the *gating regression* obtained a *average relative absolute error* around 0.53%, which is our best result. Also, several predictors can achieve higher than 0.76 *concordance index*. The details of the following models will be found in Appendix B.3.

Grouping Patients: NG (No Grouping) Log-space transformation: Yes Outliers Detection: *Mahalanobis Distance* with *MCD estimator* Censored Data: AVE (use estimated survival times)					
Model	*RAE*	*CI*	RAE^{95}	$LI_{p<12}$	$RAE_{p>12}$
NG-LOG-MCD-AVE100-LIN	0.5385±0.04	0.7649±0.01	0.4926±0.03	4.5260±0.46	0.5068±0.04
NG-LOG-MCD-AVE100-SVR	0.5376±0.03	0.7612±0.01	0.4955±0.03	4.3355±0.40	0.5064±0.03
NG-LOG-MCD-AVE100-RT	0.5536±0.03	0.7536±0.01	0.5073±0.02	4.6130±0.39	0.5140±0.02
NG-LOG-MCD-CEN100-LINc	0.5468±0.05	0.7650±0.01	0.4964±0.04	4.3729±0.53	0.5013±0.04
NG-LOG-MCD-CEN100-SVRc	0.5755±0.04	0.7600±0.01	0.5347±0.03	4.3591±0.41	0.5555±0.03
NG-LOG-MCD-AVE100-GAT	0.5376±0.03	0.7612±0.01	0.4955±0.03	4.3355±0.39	0.5063±0.03
Grouping Patients: LR (*CART* with *log-rank statistics*) Log-space transformation: Yes Outliers Detection: MCD (*Mahalanobis Distance* with *MCD estimator*) Censored Weight: AVE (use estimated survival times)					
Model	*RAE*	*CI*	RAE^{95}	$LI_{p<12}$	$RAE_{p>12}$
LR-LOG-MCD-AVE100-LIN	0.5676±0.05	0.7650±0.01	0.5085±0.04	4.5858±0.61	0.4992±0.04
LR-LOG-MCD-AVE100-SVR	0.5729±0.04	0.7582±0.01	0.5129±0.03	4.4462±0.37	0.5025±0.03
LR-LOG-MCD-AVE100-RT	0.5616±0.03	0.7533±0.01	0.5106±0.02	4.9125±0.52	0.5116±0.02
LR-LOG-MCD-CEN100-LINc	0.5601±0.04	0.7603±0.01	0.5058±0.03	4.4828±0.44	0.5007±0.04
LR-LOG-MCD-CEN100-SVRc	0.6044±0.04	0.7530±0.01	0.5507±0.02	4.6020±0.44	0.5563±0.02
LR-LOG-MCD-AVE100-GAT	0.5533±0.03	0.7577±0.01	0.5043±0.02	4.5273±0.23	0.5098±0.03

Table 5.8: Experimental results on regression methods in logarithmic space

Table 5.9 shows the results (p-value) of paired-t test between each predictor and the baseline methods (i.e., the models which simply use the median survival time or the average survival time of the population as the prediction for each individual patient). The predictors trained using the *support vector regression*, the *regression trees*, and the *gating regression* are statistically better than both baseline models.

Model	Median Survival Time	Average Survival Time
NG-LOG-MCD-AVE100-LIN	0.075	0.031 (<0.05)
NG-LOG-MCD-AVE100-SVR	0.038 (<0.05)	0.014 (<0.05)
NG-LOG-MCD-AVE100-RT	0.037 (<0.05)	0.017 (<0.05)
NG-LOG-MCD-CEN100-LINc	0.159	0.068
NG-LOG-MCD-CEN100-SVRc	0.515	0.152
NG-LOG-MCD-AVE100-GAT	0.038(<0.05)	0.014(<0.05)

Table 5.9: P-values of pair-t test

Also, Figure 5.4 plots true survival times versus predicted survival times for all patients using the predictor learned using the *linear regression*. More resulting figures can be found in Appendix B.3.

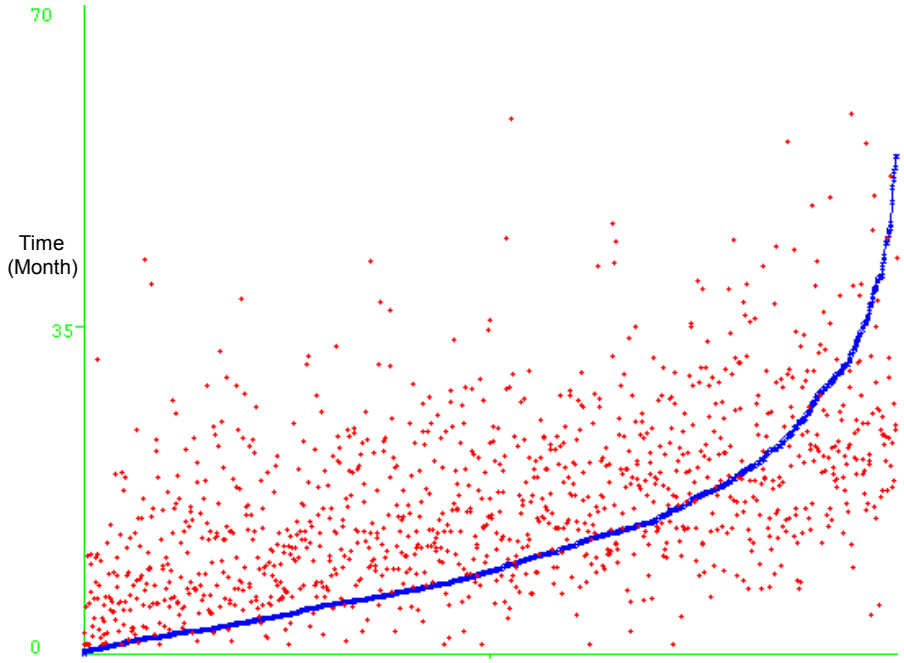


Figure 5.4: A Visualization of true survival times versus predicted survival times

5.2.9 Classification

As we mentioned in Section 2.3, many researchers in survival analysis are interested in looking at the median survival time and determining whether a given patient will survive longer than the median. We also use our predictors to classify each patient into "long survivor" versus "short survivor" where the classification boundary is the median survival time of the whole population. Table 5.10 shows the accuracy, sensitivity, and specificity of each classifier.

After log-space transformation, eliminating outliers, and handling censored data, most predictors were able to achieve at least 70% accuracy. The *gating regression*, the *support vector regression*, the *support vector regression for censored target* have slightly higher accuracy among our six learning algorithms, but the differences are not statistically significant. We can see an improvement (statistically significant) as comparing to the baseline methods which simply use the median or the average survival time. The confusion matrices of each classifier and other conventional classification methods are shown in Table B.8.

Baseline			
Model	Accuracy	Sensitivity	Specificity
BASELINE-MED	0.5388	0.1954	0.7973
BASELINE-AVE	0.5705	0.0000	1.0000
Grouping Patients: NG (No Grouping) Log-space transformation: Yes Outliers Detection: MCD (*Mahalanobis Distance* with *MCD estimator*) Censored Data: AVE (average survival time of uncensored data in the risk set)			
Model	Accuracy	Sensitivity	Specificity
NG-LOG-MCD-AVE100-LIN	0.7197	0.8146	0.6415
NG-LOG-MCD-AVE100-SVR	0.7387	0.7673	0.7153
NG-LOG-MCD-AVE100-RT	0.6955	0.8084	0.6025
NG-LOG-MCD-AVE100-LINc	0.7150	0.8321	0.6186
NG-LOG-MCD-AVE100-SVRc	0.7373	0.6406	0.8169
NG-LOG-MCD-AVE100-GAT	0.7392	0.7662	0.7169
Grouping Patients: LR (*CART* with *log-rank statistics*) Log-space transformation: Yes Outliers Detection: MCD (*Mahalanobis Distance* with *MCD estimator*) Censored Data: AVE (average survival time of uncensored data in the risk set)			
Model	Accuracy	Sensitivity	Specificity
LR-LOG-MCD-AVE100-LIN	0.7215	0.8031	0.6553
LR-LOG-MCD-AVE100-SVR	0.7337	0.7550	0.7165
LR-LOG-MCD-AVE100-RT	0.7014	0.8199	0.6053
LR-LOG-MCD-AVE100-LINc	0.7154	0.8262	0.6256
LR-LOG-MCD-AVE100-SVRc	0.7351	0.6524	0.8022
LR-LOG-MCD-AVE100-GAT	0.7337	0.7634	0.7097

Table 5.10: Experimental results on classification

5.3 Discussion

In this work, our goal is to learn a model from historical patient data that can effectively predict survival times for novel patients. We build this model from a data set of patients historical records, including personal attributes, diagnostic assessments, and blood test results. We consider survival prediction as a regression problem and base our solution on a combination of unsupervised and supervised learning. In Chapter 4, we introduced our survival prediction framework, which includes the following steps: (1) processing data, (2) segregating populations, (3) eliminating outliers, (4) handling censored information, and then at performance time, (5) predicting survival times for each individual patients. In our experiments, we tested various combinations of approaches on these steps, with the goal of selecting the best combination for our framework. In this section, we will discuss the results on these approaches for each step.

In our experiments, the quality of each predictor was evaluated by its *average relative absolute error* (primary) and its *concordance index* (secondary) where these measurements are the average results of *5-fold cross-validation*. Table 5.11 shows the top 10 best models. Here, the best model obtains an *average relative absolute error* of 0.5376 ± 0.03 in predicting a survival time for each individual (uncensored) patient. For classification task, several combinations can achieve at least

Regressor	*RAE*	*CI*	RAE^{95}	$Ll_{p<12}$	$RAE_{p>12}$
NG-LOG-MCD-AVE100-GAT	0.5376±0.03	0.7612±0.01	0.4955±0.03	4.3355±0.39	0.5063±0.03
NG-LOG-MCD-AVE100-SVR	0.5376±0.03	0.7612±0.01	0.4955±0.03	4.3355±0.40	0.5064±0.03
NG-LOG-ALL-AVE100-LIN	0.5380±0.04	0.7679±0.01	0.4929±0.03	4.4588±0.42	0.4955±0.03
NG-LOG-MCD-AVE100-LIN	0.5385±0.04	0.7649±0.01	0.4926±0.03	4.5260±0.46	0.5068±0.04
NG-LOG-ALL-AVE100-GAT	0.5392±0.03	0.7640±0.01	0.4952±0.02	4.2810±0.32	0.5049±0.03
NG-LOG-ALL-AVE100-SVR	0.5393±0.03	0.7641±0.01	0.4967±0.03	4.2782±0.32	0.5055±0.03
NG-LOG-ALL-AVE100-RT	0.5418±0.03	0.7580±0.01	0.4966±0.02	4.5625±0.36	0.5014±0.04
NG-LOG-ALL-CEN100-LINc	0.5462±0.04	0.7678±0.01	0.4968±0.03	4.4718±0.49	0.4909±0.04
NG-LOG-MCD-AVE100-LINc	0.5468±0.05	0.7650±0.01	0.4964±0.04	4.3729±0.53	0.5013±0.04
NG-REG-ALL-AVE100-SVR	0.5468±0.03	0.7432±0.01	0.5092±0.02	3.9596±0.47	0.5267±0.02

Table 5.11: Top 10 models

70% accuracy. The results presented in this chapter suggest that we can effectively predict patient's survival times by taking the following steps:

1. Processing Data: representing raw data using numerical attributes such that categorical attributes are transformed into a binary attribute or multiple binary attributes and transforming the class labels into logarithmic space.

2. Segregating Patients: no need to segregating patients if the target is to minimize the *average relative absolute error*.

3. Handling Censoring: imputing a survival time for each censored patient by taking the average survival time of uncensored patients in the risk set of that patient.

4. Eliminating Outliers: calculating the *Mahalanobis distance* using the *minimum covariance estimator* for each patient and removing those who are relatively too far in their distances.

5. Learning Predictors: for each sub-population, learn a predictor using the *linear regression*, the *support vector regression*, or the variations of them.

We will briefly discuss the results of each step above. There are a few approaches that we thought should be effective, but experimental results did not support our ideas. Here, we will provide our assumptions about their limitations.

5.3.1 Processing Features

In Section 4.5.2, we discussed two feature selection methods, the *subset selection by backward wrapper* and the *variable ranking by mutual information*. These two methods appear to perform similarly on the regression accuracy (i.e., not statistically significant). One potential reason is that, in most cases, the evaluation scores for many features are very close. We performed several experiments with different control parameters, and these methods either keep almost all features or remove too many features. Another reason might be that these two methods are not appropriate for our data set or for survival prediction problems. Therefore, a potential future work is to test different feature selection methods or evaluation criteria (other than the *mutual information* or *expected prediction errors* as in our work).

5.3.2 Segregating Patients

We tested three grouping methods, including (1) the *CART* with the *log-rank statistics*, (2) the *CART* with *gain-ratio*, and (3) *EM clustering* (Section 4.3). In most experiments, (1) was the most effective grouping method and (3) was the worst. Unfortunately, we found that even the best grouping method, the *CART* with the *log-rank statistics*, is very competitive to the predictors without using grouping method. We therefore suggest that we choose the simpler one — no need to grouping patients.

On the other hand, predictors built with the *CART* with the *log-rank statistics* can have better performance in predicting survival times for short survivors. In our experiments, LOG predictors usually achieve lower *relative absolute error*; we found, however, that these LOG predictors often have larger $L1_{p<12}$. Table 5.12 shows the top 5 models with the lowest $L1_{p<12}$. After segregating patients using the *CART* with the *log-rank statistics*, the *linear regression for censored targets* obtains the lowest $L1_{p<12}$ among all models in our experiment.

Regressor	*RAE*	*CI*	RAE^{95}	$L1_{p<12}$	$RAE_{p>12}$
LR-REG-ALL-CEN100-LINc	0.5618±0.02	0.7410±0.01	0.5252±0.02	3.9586±0.41	0.5480±0.01
NG-REG-ALL-AVE050-LIN	0.5524±0.02	0.7498±0.01	0.5102±0.02	4.1004±0.22	0.5257±0.01
NG-REG-MCD-AVE050-SVR	0.5503±0.04	0.7501±0.01	0.4953±0.02	4.1312±0.40	0.5125±0.03
LR-REG-ALL-AVE050-LIN	0.5598±0.01	0.7421±0.01	0.5168±0.02	4.1554±0.32	0.5307±0.01
NG-REG-ALL-CEN100-LINc	0.5589±0.02	0.7445±0.01	0.5215±0.02	4.1591±0.29	0.5405±0.01

Table 5.12: Top 5 models with the lowest $L1_{p<12}$

In reality, it is somehow more crucial to make accurate predictions for short survivors. Standard treatments for cancer patients usually take certain amount of time; for example, the duration of treatment is three months for lung cancer [18]. If a patient is expected to die before the end of the treatment, it is usually not recommend to undergo the treatment (these treatments could be risky and reduce quality of life). Therefore, if the goal of a survival prediction system is to make predicts for short survivors, we recommend that *CART* with the *log-rank statistics* should be applied without log-space transformation.

Here are our thoughts on the reason that *log-rank statistics* outperforms the *gain-ratio* as the splitting criterion in *CART*. The *gain-ratio* was based on the class labels, which are not the actual survival times for all patients. For a censored patient, the class label is a lower bound (without imputation) or an approximation (with imputation) of the patient's actual survival time. This problem will probably bias the measurement of the *gain-ratio*. In contrast, *log-rank statistics* effectively incorporates censored observations since this method only requires the information on the size of the risk set and the number of deaths and each distinct time point.

The *EM clustering* was unfortunately not very effective in our work — indeed, the results of using this method are worse than the results of not using it. Recall that the *EM clustering* groups patients with similar feature values (Section 4.3.2). The fact that this method does not depend on class labels may be why this is not appropriate for survival prediction problems. There are many other clustering methods that do not use labels; and one potential future work is to verify this

assumption by testing other clustering approaches.

5.3.3 Learning Predictors

The *support vector regression for censored targets (SVRc)* was shown to be effective in [28], but the performance on regression is not exceptional in our work. However, we notice that the *SVRc*'s *training error* is lower than other learning algorithms, which suggests that the *SVRc* might have been over-fitted to the training data. The predictor may be improved by changing the control parameters or incorporating some regularization methods. (This is future work.) Aside from the regression results, the *SVRc* usually achieves the highest accuracy in the classification task, which suggests that this combination might be a possible solution if one is interested in differentiating long survivors versus short survivors.

Previously, we assumed that the relationship between features and survival times is not linear, and therefore, we attempted to address this problem by incorporating the *kernel* methods. We tested two *kernel* functions, K_2 and K_{GRB} (Section 4.2.1) for our *SVR* and *SVRc*, but their performance are not better than those without kernels. Therefore, we assume that linear algorithms are adequate for survival predictions.

Recall that one of our goal is to find the best learning algorithm among six algorithms (Section 4.2) for our prediction framework. In many cases, we found that the results of several algorithms very competitive. For example, in Table 5.11, NG-LOG-MCD-AVE100-GAT is lower in its *relative error* while NG-LOG-ALL-AVE100-LIN is higher in its *concordance index*, but the differences between these measurements among different learning algorithms are not statistically significant (by paired t-test). It is hard to determine which one is the best regression algorithm. Therefore, we assume that most regression methods can be applied to obtain comparable outcome for survival predictions.

5.3.4 Handling Censoring

In Section 4.1.1, we proposed three approaches to approximate an event time for each censored patient, including (1) adding a constant, (2) taking the average survival time over uncensored patients in the risk set, and (3) taking the average survival time over all patients in the risk set. Overall, we found (1) is not as effective as (2), and (3) tends to over-estimate survival times. (Therefore, in this section, we only show the results that use (2) to estimate survival time for censored data versus the results that simply use censored time as the class labels.)

We also attempt to lower the "weights" of censored data (Section 4.1.2). In our experiments, we tested two constant weights, 0.1 and 0.5, for each censored patient, and we did not see improvement from this technique after log-space transformation. (However, we found improvement when using regular predictors.) We can only assume that this approach was unfortunately ineffective in handling censored observations for our best model.

As we mentioned in Section 5.3.3, the learning algorithms that we tested are very comparable. One interesting observation is that, the *linear regression* uses the estimated survival times while the *linear regression for censored targets* uses the original censored times for training a predictor, and their outcomes have no statistical difference. Hence, we assume that these two methods (either changing the class labels or modifying the algorithm) for handling censored data are comparable.

Chapter 6

Conclusion

Survival prediction is the task of predicting the length of time that an individual patient will survive; accurate predictions can give doctors better guidelines on selecting treatments and planning futures. This differs from the standard *survival analysis*, which focuses on population-based studies and tries to discover the prognostic factors and/or analyze the median survival times of different groups of patients.

This work provides a system that predicts survival times for individual cancer patients based on personal attributes, clinical assessments, and blood test results. We view the task of survival predictions as a regression problem, which maps the characteristic of each patients to his/her survival time. As the relationship between features and survival time is still not understood, we consider various ways to learn these models from historical patient records. This is challenging due to the presence of irrelevant features, outliers, and missing class labels in processing medical/clinical data. This dissertation describes our approach for overcoming these, and other challenges, producing techniques that can predicts survival times.

Our experiments show that the *linear regression*, the *support vector regression*, and the *gating regression* are effective: each predictor can obtain an *average relative absolute error* lower than 0.54 over all of the testing instances (where the *average relative absolute error* of a regressor is $E[\frac{|t-p|}{p}]$ where t is the true survival time and p is our prediction for each patient). We also use our regressors to classify each patient into "long survivor" versus "short survivor" where the classification boundary is the median survival time of the entire population; here, we show that several regressors can achieve at least 70% accuracy. These experimental results verify that we can effectively predict patients' survival times with a combination of statistical and machine learning approaches.

Many of these techniques involve first segregating patients into smaller sub-populations; we had anticipated that this would improve the accuracy of our survival predictions; however, our experimental results did not support this expectation. Nevertheless, we found that two aspects that we tested in this work, *log-space transformation* and outlier elimination by *minimum covariance estimation*, could effectively improve the overall performance of our predictors.

There are several ways we might obtain more accurate results. One idea is to implement regularization methods on the *SVRc* algorithm to address its problem of over-fitting the training samples. Second, recalling that roughly 50% of our data set is censored, it might be useful to explore better techniques for utilizing censored data.

Our current system, as is, is already effective. We expect that any further improvements on the survival prediction research would pave the way towards the use of individual prognosis system in an actual clinical setting.

Bibliography

[1] D. G. Beer, S. L. Kardia, C.-C. Huang, T. J. Giordano, A. M. Levin, D. E. Misek, L. Lin, G. Chen, T. G. Gharib, D. G. Thomas, M. L. Lizyness, R. Kuick, S. Hayasaka, J. M. Taylor, M. D. Iannettoni, M. B. Orringer, and S. Hanash. Gene-expression profiles predict survival of patients with lung adenocarcinoma. *Nature Medicine*, 8, 2002.

[2] B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, Pittsburgh, PA, USA, 1992. ACM Press.

[3] M. J. Bradburn, T. G. Clark, S. B. Love, and D. G. Altman. Survival analysis part ii: Multivariate data analysis - an introduction to concepts and methods. *British Journal of Cancer*, 89 (3), 2003.

[4] L. Breiman, J. Friedman, R. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth Inc, Belmont, CA, 1984.

[5] J. Buckley and I. James. Linear regression with censored data. *Biometrika*, 66, 1979.

[6] J. G. Carbonell, R. S. Michalski, and T. M. Mitchell. Machine learning: A historical and methodological analysis. *AI Magazine*, 4(3), 1983.

[7] N. A. Christakis and E. B. Lamont. Extent and determinants of error in doctors' prognoses in terminally ill patients: prospective cohort study. *British Medical Journal*, 320, 2000.

[8] C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20, 1995.

[9] D. R. Cox. Regression models and life-tables. *Journal of the Royal Statistical Society. Series B (Methodological)*, 34(2), 1972.

[10] D. R. Cox. *Analysis of Survival Data*. Chapman and Hall, London, UK, 1984.

[11] C. de Investigacin Prncipe Felipe. Prophet: A tool for building a class predictor. 2007.

[12] D. Delen, G. Walker, and A. Kadam. Predicting breast cancer survivability: A comparison of three data mining methods. *Artificial Intelligence in Medicine*, 34, 2005.

[13] A. P. Dempster, N. M. Laird, and D. Rubin. Maximum likelihood estimation from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, 39(1), 1977.

[14] R. Dybowski, V. Gant, P. Weller, and R. Chang. Prediction of outcome in critically ill patients using artificial neural network synthesised by genetic algorithm. *The Lancet*, 347, 1996.

[15] A. S. Elstein, L. S. Shulman, and S. A. Sprafka. *Medical Problem Solving: An Analysis of Clinical Reasonging*. Harvard University Press, Cambridge, Massachusetts, USA, 1978.

[16] T. Fawcett. An introduction to roc analysis. *Pattern Recognition Letters*, 27, 2005.

[17] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3, 2003.

[18] R. G. Hagerty, P. N. Butow, P. A. Ellis, E. A. Lobb, S. Pendlebury, N. Leighl, D. Goldstein, S. K. Lo, and M. H. Tattersall. Cancer patient preferences for communication of prognosis in the metastatic setting. *Journal of Clinical Oncology*, 22(9), 2004.

[19] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: An update. *SIGKDD Explorations*, 11, 2009.

[20] F. E. Harrell, K. L. Lee, and D. B. Mark. Tutorial in biostatistics multivariable prognostic models. *Statistics in Medicine*, 15, 1996.

[21] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, New York, NY, USA, 2001.

[22] R. A. Howard and J. E. Matheson. Influence diagrams. *Readings on the Principles and Applications of Decision Analysis*, 2, 1983.

[23] K. Jayasurya, G. Fund, S. Yu, C. Dehing-Oberije, D. Ruysscher, A. Hope, W. Neve, Y. Lievens, P. Lambin, and A. L. Dekker. Comparison of bayesian network and support vector machine models for two-year survival prediction in lung cancer patients treated with radiotherapy. *Medical Physicists*, 37, 2010.

[24] G. H. John, R. Kohavi, and K. Pfleger. Irrelevant features and the subset selection problem. In *Proceedings of the Eleventh International Conference on Machine Learning*, San Franciso, CA, USA, 1994. Morgan Kaufmann.

[25] J. D. Kalbfleisch and R. L. Prentice. *The Statistical Analysis of Failure Time Data*. Wiley Series in Probability and Mathematical Statistics, 1980.

[26] P. S. Kamath, R. H. Wiesner, M. Malinchoc, W. Kremers, T. M. Therneau, C. L. Kosberg, G. D'Amico, E. R. Dickson, and W. R. Kim. A model to predict survival in patients with end-stage liver disease. *Hepatology*, 33(2), 2003.

[27] E. L. Kaplan and P. Meier. Nonparametric estimation from incomplete observations. *Journal of the American Statistical Association*, 53, 1958.

[28] F. M. Khan and V. B. Zubek. Support vector regression for censored data (svrc): A novel tool for survival analysis. In *Eighth IEEE International Conference on Data Mining*, Pisa, Italy, 2008.

[29] W. A. Knaus, D. P. Wagner, E. A. Draper, J. E. Zimmerman, M. Bergner, P. G. Bastos, C. A. Sirio, D. J. Murphy, T. Lotring, and A. Damiano. The apache iii prognostic system. risk prediction of hospital mortality for critically ill hospitalized adults. *Chest*, 100(6), 1991.

[30] Y.-J. Lee1, O. L. Mangasarian, and W. Wolberg. Survival-time classification of breast cancer patients. *Computational Optimization and Applications*, 25(1-3), 2003.

[31] C. Lippert. Clustering-tutorial. http://agbs.kyb.tuebingen.mpg.de/wikis/bg/tutorial_GMM.pdf.

[32] H. Liu and L. Yu. Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on Knowledge and Data Engineering*, 17(4), 2005.

[33] O. Luaces, J. R. Quevedo, F. Taboada, and G. M. Albaiceta. Prediction of probability of survival in critically ill patients optimizing the area under the roc curve. In *International Joint Conference of Artificial Intelligence*, Hyderabad, India, 2007.

[34] P. C. Mahalanobis. On the generalised distance in statistics. In *Proceedings of the National Institute of Sciences of India*, 1936.

[35] N. Mantel and W. Haenszel. Statistical aspects of the analysis of data from retrospective studies of disease. *Journal of national Cancer Institute*, 22, 1959.

[36] L. Ohno-Machado. Modelling medical prognosis: Survival analysis techniques. *Journal of Biomedical Informatics*, 34, 2001.

[37] J. Orbe, E. Ferreira, and V. Nunez-Anton. Censored partial regression. *Biostatistics*, 4, 2003.

[38] J. Pearl. Reverend bayes on inference engines: A distributed hierarchical approach. In *Proceedings of the National Conference on Artificial Intelligence*, San Mateo, CA, USA, 1982.

[39] R. Peto, M. C. Pike, P. Armitage, N. E. Breslow, D. R. Cox, S. V. Howard, and N. Mantel. Design and anallysis of randomized clinical trails requiring prolonged observation of each patient. *British Journal of Cancer*, 35, 1977.

[40] R. Peto, M. C. Pike, P. Armitage, N. E. Breslow, D. R. Cox, S. V. Howard, N. Mantel, K. McPherson, J. Peto, and P. G. Smith. Design and analysis of randomized clinical trails requiring prolonged observation of each patient. *British Journal of Cancer*, 35(1), 1997.

[41] R. Picard and D. Cook. Cross-validation of regression models. *Journal of American Statistical Association*, 79(387), 1984.

[42] P. M. Prsent, M. Christos, and S. B. Haibe-kains. Use of machine learning in bioinformatics. *Annual Academique*, 2004.

[43] P. Qui, A. J. Gentles, and S. K. Plevritis. Fast calculation of pairwise mutual information for gene regulatory network reconstruction. *Computer Methods and Programs in Biomedicine*, 94, 2009.

[44] J. R. Quinlan. Learning with continuous classes. In *Proceedings of 5th Australian Joint Conference on Artificial Intelligence*, Singapore, 1992.

[45] J. R. Quinlan. *C4.5 Programs for Machine Learning*. Morgan Kaufmann Publishers, Inc, 1993.

[46] M. Radespiel-Troger, T. Rabenstein, H. T. Schneider, and B. Lausen. Comparison of tree-based methods for prognostic stratification of survival data. *Artificial Intelligence in Medicine*, 28, 2003.

[47] P. J. Rousseeuw and K. van Driessen. A fast algorithm for the minimum covariance determinant estimator. *Technometrics*, 41(3), 1999.

[48] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, 2003.

[49] P. Seve, I. Ray-Coquard, V. Trillet-Lenoir, M. Sawyer, J. Hanson, C. Broussolle, S. Negrier, C. Dumontet, and J. Mackey. Low serum albumin levels and liver metastasis are powerful prognostic markers for survival in patients with carcinomas of unknown primary site. *Cancer*, 109(11), 2006.

[50] S. K. Shevade, S. S. Keerthi, C. Bhattacharyya, and K. R. K. Murthy. Improvements to the smo algorithm for svm regression. *IEEE Transations on Neural Networks*, 11(5), 2000.

[51] P. K. Shivaswamy, W. Chu, and M. Jansche. A support vector approach to censored targets. In *the Seventh IEEE International Conference on Data Mining*, California, USA, 2007. IEEE Computer Society.

[52] E. H. Shortliffe and J. J. Cimino. *Biomedical Informatics: Computer Applications in Health Care and Biomedicine*. Springer, New York, NY, USA, 2006.

[53] A. J. Smola and B. Scholkopf. A tutorial on support vector regression. *NeuroCOLT Technical Report NC-TR-98-030*, 1998.

[54] M. Ture, F. Tokatli, and I. Kurt. Using kaplanmeier analysis together with decision tree methods in determining recurrence-free survival of breast cancer patients. *Expert Systems with Applications*, 36, 2009.

[55] V. Vapnik. *The Nature of Statistical Learning Theory: Data Mining, Inferenece, and Prediction*. Wiley, New York, NY, USA, 1998.

[56] Z. Ying, L. J. Wei, and J. S. Lin. Prediction of survival probability based on a linear regression model. *Biometrika*, 79(1), 1992.

[57] B. Zupan, J. Demsar, M. W. Kattan, J. R. Beck, and I. Bratko. Machine learning for survival analysis: A case study on recurrence of prostate cancer. *Artificial Intelligence in Medicine*, 20, 2000.

Appendix A

Data Set

A.1 Attribute Descriptions

Table A.1: Raw Data

Feature Name	Data Type	Range
Personal Attributes		
Date of Birth	Date	YYYY-MM-DD
Date of Reference	Date	YYYY-MM-DD
Date of Death	Date	YYYY-MM-DD
Gender	Nominal	{"Male", "Female" }
Age	Numerical	$\mathbb{R}$
Age Older than 65	Nominal	{ "age < 65", "age ≥ 65" }
BMI	Numerical	$\mathbb{R}$
Weight Change	Numerical	$\mathbb{R}$
Clinical Assessments		
BOX 1 Score	Numerical	$\mathbb{R}$
BOX 2 Score	Numerical	$\mathbb{R}$
BOX 3 Score	Numerical	$\mathbb{R}$
BOX 4 Score	Numerical	$\mathbb{R}$
Food Intake	Nominal	{"normal", "less", "little", "liquid" }
No Problem	Nominal	{"No", "Yes"}
No Appetite	Nominal	{"No", "Yes"}
Nausea	Nominal	{"No", "Yes"}
Constipation	Nominal	{"No", "Yes"}
Taste Funny	Nominal	{"No", "Yes"}
Smelling Problem	Nominal	{"No", "Yes"}
Swallowing Problem	Nominal	{"No", "Yes"}
Feel Full	Nominal	{"No", "Yes"}
Feel Pain	Nominal	{"No", "Yes"}
Dental Problem	Nominal	{"No", "Yes"}
Vomitting	Nominal	{"No", "Yes"}
Diarrhea	Nominal	{"No", "Yes"}
Sore Mouth	Nominal	{"No", "Yes"}
Dry Mouth	Nominal	{"No", "Yes"}
Activity	Nominal	{0, 1, 2, 3, 4}
PS-PGA	Nominal	{0, 1, 2, 3, 4}
Cancer Stage	Nominal	{1, 2, 3, 4 }
Cancer Type	Nominal	{"Brunchus-Lung", "Colon-Rectal","Esophagus", "Head/Neck','Pancrease','Stomach", "Misc"}
Blood Test Results		
Granulocytes	Numerical	$\mathbb{R}$
LDH Serum	Numerical	$\mathbb{R}$
Lymphocytes	Numerical	$\mathbb{R}$
Platelet	Numerical	$\mathbb{R}$
WBC Count	Numerical	$\mathbb{R}$
Calcium Serum	Numerical	$\mathbb{R}$
HGB	Numerical	$\mathbb{R}$
Creatinine Serum	Numerical	$\mathbb{R}$
Albumin	Numerical	$\mathbb{R}$

Table A.2: Pre-processed Data

Feature Name	Variable Name	Data Type	Range
Personal Attributes			
Gender	X^{gender}	Binary	0: "Male", 1: "Female"
Age	X^{age}	Numerical	$\mathbb{R}$
Age Older than 65	X^{age-65}	Binary	{0: age $<$ 65 1: age $\geq$ 65 }
BMI	X^{bmi}	Numerical	$\mathbb{R}$
Weight Change	X^{wl}	Numerical	$\mathbb{R}$
BOX1	X^{box-1}	Numerical	$\mathbb{R}$
BOX2	X^{box-2}	Numerical	$\mathbb{R}$
Clinical Assessments			
No Problem	$X^{no-prob}$	Binary	{ 0: "No", 1: "Yes" }
No Appetite	X^{no-app}	Binary	{ 0: "No", 1: "Yes" }
Nausea	X^{nausea}	Binary	{ 0: "No", 1: "Yes" }
Constipation	X^{const}	Binary	{ 0: "No", 1: "Yes" }
Taste Funny	X^{taste}	Binary	{ 0: "No", 1: "Yes" }
Smelling Problem	X^{smell}	Binary	{ 0: "No", 1: "Yes" }
Swallowing Problem	$X^{swallow}$	Binary	{ 0: "No", 1: "Yes" }
Feel Full	X^{full}	Binary	{ 0: "No", 1: "Yes" }
Feel Pain	X^{pain}	Binary	{ 0: "No", 1: "Yes" }
Dental Problem	X^{dental}	Binary	{ 0: "No", 1: "Yes" }
Vomitting	X^{vomit}	Binary	{ 0: "No", 1: "Yes" }
Diarrhea	$X^{diarrhea}$	Binary	{ 0: "No", 1: "Yes" }
Sore Mouth	$X^{sore-mouth}$	Binary	{ 0: "No", 1: "Yes" }
Dry Mouth	$X^{dry-mouth}$	Binary	{ 0: "No", 1: "Yes" }
Physician Global Assessments			
PS-PGA (0)	X^{ps-0}	Binary	{ 0: "No", 1: "Yes" }
PS-PGA (1)	X^{ps-1}	Binary	{ 0: "No", 1: "Yes" }
PS-PGA (2)	X^{ps-2}	Binary	{ 0: "No", 1: "Yes" }
PS-PGA (3)	X^{ps-3}	Binary	{ 0: "No", 1: "Yes" }
PS-PGA (4)	X^{ps-4}	Binary	{ 0: "No", 1: "Yes" }
Tumour Sites and Stages			
Brunchus-Lung Cancer	$X^{dx-lung}$	Binary	{ 0: "No", 1: "Yes" }
Colon-rectal Cancer	$X^{dx-colon}$	Binary	{ 0: "No", 1: "Yes" }
Esophagus Cancer	$X^{dx-esop}$	Binary	{ 0: "No", 1: "Yes" }
Head/Neck Cancer	$X^{dx-head}$	Binary	{ 0: "No", 1: "Yes" }
Pancrease Cancer	$X^{dx-panc}$	Binary	{ 0: "No", 1: "Yes" }
Stomach Cancer	$X^{dx-stom}$	Binary	{ 0: "No", 1: "Yes" }
Misc Cancer	$X^{dx-misc}$	Binary	{ 0: "No", 1: "Yes" }
Cancer Stage (1)	$X^{stage-1}$	Binary	{ 0: "No", 1: "Yes" }
Cancer Stage (2)	$X^{stage-2}$	Binary	{ 0: "No", 1: "Yes" }
Cancer Stage (3)	$X^{stage-3}$	Binary	{ 0: "No", 1: "Yes" }
Cancer Stage (4)	$X^{stage-4}$	Binary	{ 0: "No", 1: "Yes" }
Blood Test Results			
Granulocytes	$X^{blood-gra}$	Numerical	$\mathbb{R}$
LDH Serum	$X^{blood-ldh}$	Numerical	$\mathbb{R}$
Lymphocytes	$X^{blood-lym}$	Numerical	$\mathbb{R}$
Platelet	$X^{blood-pla}$	Numerical	$\mathbb{R}$
WBC Count	$X^{blood-wbc}$	Numerical	$\mathbb{R}$
Calcium Serum	$X^{blood-cal}$	Numerical	$\mathbb{R}$

Continued on next page

Table A.2 – continued from previous page			
Feature Name	Variable Name	Data Type	Range
HGB	$X^{blood-hgb}$	Numerical	$\mathbb{R}$
Creatinine Serum	$X^{blood-cre}$	Numerical	$\mathbb{R}$
Albumin	$X^{blood-alb}$	Numerical	$\mathbb{R}$
Labels			
Time (Event or Censored)	t_i	Numerical	$\mathbb{R}$
Censored Flag	c_i	Binary	{ 0: Uncensored, 1: Censored }

A.2 Histogram

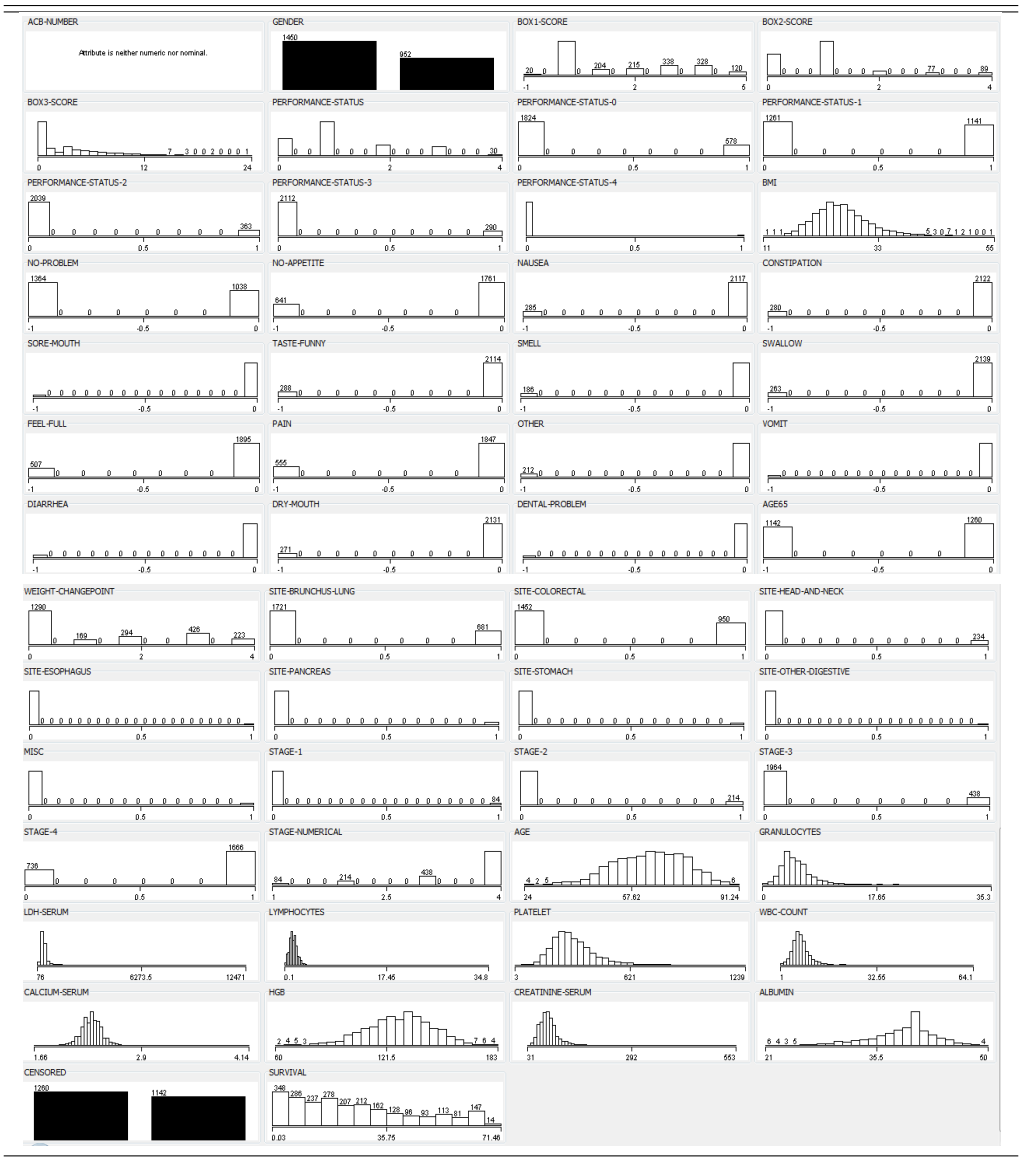


Table A.3: Histograms of features

Appendix B

Detailed Experimental Results

B.1 Experiment Methods

Table B.1: A List of methods for predicting survival times

Method Name	Descriptions
	Grouping Patients
NG	no grouping
LR	*classification and regression trees* with *log-rank statistics* as the splitting criterion
GR	*classification and regression trees* with *gain-ratio* as the splitting criterion
EC	*EM clustering*
	Log-space Transformation
REG	regular space
LOG	logarithmic space
	Outlier Detection
ALL	no outlier detection, keep all instances
MCD	use *Mahalanobis distance* with *MCD estimator*
	Handling Censored Data
CEN100	use censored time as the class label
CEN050	use censored time as the class label, and weight = 0.5
AVE100	take the average survival time of uncensored patient in the risk set
AVE050	take the average survival time of uncensored patient in the risk set, and weight = 0.5
	Regression Algorithms
MED	use median survival time of the population
AVE	use average survival time of the population
LIN	*linear regression*
SVR	*support vector regression*
RT	*regression trees*
LINc	*learner regression for censored targets*
SVRc	*support vector regression for censored targets*
GAT	*gating regression*

B.2 Detailed Experimental Results

Table B.2: Experimental Methods

Methods	Log-space	Outlier Detection	Class label of censored data	Censored Weights	Algorithm
Baseline					
BASELINE-MED	No	No	Censored Time		Use median survival time
BASELINE-AVE	No	No	Censored Time		Use average survival time
Grouping Method: Handcraft tree					
HANDCRAFT-LIN	No	No	Censored Time		Use median survival time
HANDCRAFT-SVR	No	No	Censored Time		Use average survival time
Grouping Method: No grouping					
NG-CEN100-ALL-REG-LIN	No	No	Censored Time		*linear regression*
NG-CEN100-ALL-REG-SVR	No	No	Censored Time		*support vector regression*
NG-CEN100-ALL-REG-RT	No	No	Censored Time		*regression trees*
NG-REG-ALL-AVE050-MED	No	No	Estimated time	0.5	Use median survival time
NG-REG-ALL-AVE050-AVE	No	No	Estimated time	0.5	Use average survival time
NG-REG-ALL-AVE050-LIN	No	No	Estimated time	0.5	*linear regression*
NG-REG-ALL-AVE050-SVR	No	No	Estimated time	0.5	*support vector regression*
NG-REG-ALL-AVE050-RT	No	No	Estimated time	0.5	*regression trees*
NG-REG-ALL-AVE050-GAT	No	No	Estimated time	0.5	*gating regression*
NG-REG-ALL-CEN050-LINc	No	No	Censored Time	0.5	*linear regression for censored targets*
NG-REG-ALL-CEN050-SVRc	No	No	Censored Time	0.5	*support vector regression for censored targets*
NG-REG-ALL-AVE100-MED	No	No	Estimated time	1	Use median survival time
NG-REG-ALL-AVE100-AVE	No	No	Estimated time	1	Use average survival time
NG-REG-ALL-AVE100-LIN	No	No	Estimated time	1	*linear regression*
NG-REG-ALL-AVE100-SVR	No	No	Estimated time	1	*support vector regression*
NG-REG-ALL-AVE100-RT	No	No	Estimated time	1	*regression trees*
NG-REG-ALL-AVE100-GAT	No	No	Estimated time	1	*gating regression*
NG-REG-ALL-CEN100-LINc	No	No	Censored Time	1	*linear regression for censored targets*
NG-REG-ALL-CEN100-SVRc	No	No	Censored Time	1	*support vector regression for censored targets*
NG-REG-MCD-AVE050-MED	No	Yes	Estimated time	0.5	Use median survival time
NG-REG-MCD-AVE050-AVE	No	Yes	Estimated time	0.5	Use average survival time
NG-REG-MCD-AVE050-LIN	No	Yes	Estimated time	0.5	*linear regression*
NG-REG-MCD-AVE050-SVR	No	Yes	Estimated time	0.5	*support vector regression*
NG-REG-MCD-AVE050-RT	No	Yes	Estimated time	0.5	*regression trees*
NG-REG-MCD-AVE050-GAT	No	Yes	Estimated time	0.5	*gating regression*
NG-REG-MCD-CEN050-LINc	No	Yes	Censored Time	0.5	*linear regression for censored targets*
NG-REG-MCD-CEN050-SVRc	No	Yes	Censored Time	0.5	*support vector regression for censored targets*
NG-REG-MCD-AVE100-MED	No	Yes	Estimated time	1	Use median survival time
NG-REG-MCD-AVE100-AVE	No	Yes	Estimated time	1	Use average survival time
NG-REG-MCD-AVE100-LIN	No	Yes	Estimated time	1	*linear regression*
NG-REG-MCD-AVE100-SVR	No	Yes	Estimated time	1	*support vector regression*
NG-REG-MCD-AVE100-RT	No	Yes	Estimated time	1	*regression trees*
NG-REG-MCD-AVE100-GAT	No	Yes	Estimated time	1	*gating regression*
NG-REG-MCD-CEN100-LINc	No	Yes	Censored Time	1	*linear regression for censored targets*
NG-REG-MCD-CEN100-SVRc	No	Yes	Censored Time	1	*support vector regression for censored targets*
NG-LOG-ALL-AVE050-MED	Yes	No	Estimated time	0.5	Use median survival time
NG-LOG-ALL-AVE050-AVE	Yes	No	Estimated time	0.5	Use average survival time
NG-LOG-ALL-AVE050-LIN	Yes	No	Estimated time	0.5	*linear regression*
NG-LOG-ALL-AVE050-SVR	Yes	No	Estimated time	0.5	*support vector regression*
NG-LOG-ALL-AVE050-RT	Yes	No	Estimated time	0.5	*regression trees*
NG-LOG-ALL-AVE050-GAT	Yes	No	Estimated time	0.5	*gating regression*
NG-LOG-ALL-CEN050-LINc	Yes	No	Censored Time	0.5	*linear regression for censored targets*
NG-LOG-ALL-CEN050-SVRc	Yes	No	Censored Time	0.5	*support vector regression for censored targets*
NG-LOG-ALL-AVE100-MED	Yes	No	Estimated time	1	Use median survival time
NG-LOG-ALL-AVE100-AVE	Yes	No	Estimated time	1	Use average survival time
NG-LOG-ALL-AVE100-LIN	Yes	No	Estimated time	1	*linear regression*
NG-LOG-ALL-AVE100-SVR	Yes	No	Estimated time	1	*support vector regression*
NG-LOG-ALL-AVE100-RT	Yes	No	Estimated time	1	*regression trees*
NG-LOG-ALL-AVE100-GAT	Yes	No	Estimated time	1	*gating regression*
NG-LOG-ALL-CEN100-LINc	Yes	No	Censored Time	1	*linear regression for censored targets*
NG-LOG-ALL-CEN100-SVRc	Yes	No	Censored Time	1	*support vector regression for censored targets*

Continued on next page

Table B.2 – continued from previous page

Methods	Log-space	Outlier Detection	Class label of censored data	Censored Weights	Algorithm
NG-LOG-MCD-AVE050-MED	Yes	Yes	Estimated time	0.5	Use median survival time
NG-LOG-MCD-AVE050-AVE	Yes	Yes	Estimated time	0.5	Use average survival time
NG-LOG-MCD-AVE050-LIN	Yes	Yes	Estimated time	0.5	*linear regression*
NG-LOG-MCD-AVE050-SVR	Yes	Yes	Estimated time	0.5	*support vector regression*
NG-LOG-MCD-AVE050-RT	Yes	Yes	Estimated time	0.5	*regression trees*
NG-LOG-MCD-AVE050-GAT	Yes	Yes	Estimated time	0.5	*gating regression*
NG-LOG-MCD-CEN050-LINc	Yes	Yes	Censored Time	0.5	*linear regression for censored targets*
NG-LOG-MCD-CEN050-SVRc	Yes	Yes	Censored Time	0.5	*support vector regression for censored targets*
NG-LOG-MCD-AVE100-MED	Yes	Yes	Estimated time	1	Use median survival time
NG-LOG-MCD-AVE100-AVE	Yes	Yes	Estimated time	1	Use average survival time
NG-LOG-MCD-AVE100-LIN	Yes	Yes	Estimated time	1	*linear regression*
NG-LOG-MCD-AVE100-SVR	Yes	Yes	Estimated time	1	*support vector regression*
NG-LOG-MCD-AVE100-RT	Yes	Yes	Estimated time	1	*regression trees*
NG-LOG-MCD-AVE100-GAT	Yes	Yes	Estimated time	1	*gating regression*
NG-LOG-MCD-CEN100-LINc	Yes	Yes	Censored Time	1	*linear regression for censored targets*
NG-LOG-MCD-CEN100-SVRc	Yes	Yes	Censored Time	1	*support vector regression for censored targets*
Grouping Method: *CART* with *log-rank statistics* as splitting criterion					
LR-REG-ALL-AVE050-MED	No	No	Estimated time	0.5	Use median survival time
LR-REG-ALL-AVE050-AVE	No	No	Estimated time	0.5	Use average survival time
LR-REG-ALL-AVE050-LIN	No	No	Estimated time	0.5	*linear regression*
LR-REG-ALL-AVE050-SVR	No	No	Estimated time	0.5	*support vector regression*
LR-REG-ALL-AVE050-RT	No	No	Estimated time	0.5	*regression trees*
LR-REG-ALL-AVE050-GAT	No	No	Estimated time	0.5	*gating regression*
LR-REG-ALL-CEN050-LINc	No	No	Censored Time	0.5	*linear regression for censored targets*
LR-REG-ALL-CEN050-SVRc	No	No	Censored Time	0.5	*support vector regression for censored targets*
LR-REG-ALL-AVE100-MED	No	No	Estimated time	1	Use median survival time
LR-REG-ALL-AVE100-AVE	No	No	Estimated time	1	Use average survival time
LR-REG-ALL-AVE100-LIN	No	No	Estimated time	1	*linear regression*
LR-REG-ALL-AVE100-SVR	No	No	Estimated time	1	*support vector regression*
LR-REG-ALL-AVE100-RT	No	No	Estimated time	1	*regression trees*
LR-REG-ALL-AVE100-GAT	No	No	Estimated time	1	*gating regression*
LR-REG-ALL-CEN100-LINc	No	No	Censored Time	1	*linear regression for censored targets*
LR-REG-ALL-CEN100-SVRc	No	No	Censored Time	1	*support vector regression for censored targets*
LR-REG-MCD-AVE050-MED	No	Yes	Estimated time	0.5	Use median survival time
LR-REG-MCD-AVE050-AVE	No	Yes	Estimated time	0.5	Use average survival time
LR-REG-MCD-AVE050-LIN	No	Yes	Estimated time	0.5	*linear regression*
LR-REG-MCD-AVE050-SVR	No	Yes	Estimated time	0.5	*support vector regression*
LR-REG-MCD-AVE050-RT	No	Yes	Estimated time	0.5	*regression trees*
LR-REG-MCD-AVE050-GAT	No	Yes	Estimated time	0.5	*gating regression*
LR-REG-MCD-CEN050-LINc	No	Yes	Censored Time	0.5	*linear regression for censored targets*
LR-REG-MCD-CEN050-SVRc	No	Yes	Censored Time	0.5	*support vector regression for censored targets*
LR-REG-MCD-AVE100-MED	No	Yes	Estimated time	1	Use median survival time
LR-REG-MCD-AVE100-AVE	No	Yes	Estimated time	1	Use average survival time
LR-REG-MCD-AVE100-LIN	No	Yes	Estimated time	1	*linear regression*
LR-REG-MCD-AVE100-SVR	No	Yes	Estimated time	1	*support vector regression*
LR-REG-MCD-AVE100-RT	No	Yes	Estimated time	1	*regression trees*
LR-REG-MCD-AVE100-GAT	No	Yes	Estimated time	1	*gating regression*
LR-REG-MCD-CEN100-LINc	No	Yes	Censored Time	1	*linear regression for censored targets*
LR-REG-MCD-CEN100-SVRc	No	Yes	Censored Time	1	*support vector regression for censored targets*
LR-LOG-ALL-AVE050-MED	Yes	No	Estimated time	0.5	Use median survival time
LR-LOG-ALL-AVE050-AVE	Yes	No	Estimated time	0.5	Use average survival time
LR-LOG-ALL-AVE050-LIN	Yes	No	Estimated time	0.5	*linear regression*
LR-LOG-ALL-AVE050-SVR	Yes	No	Estimated time	0.5	*support vector regression*
LR-LOG-ALL-AVE050-RT	Yes	No	Estimated time	0.5	*regression trees*
LR-LOG-ALL-AVE050-GAT	Yes	No	Estimated time	0.5	*gating regression*
LR-LOG-ALL-CEN050-LINc	Yes	No	Censored Time	0.5	*linear regression for censored targets*
LR-LOG-ALL-CEN050-SVRc	Yes	No	Censored Time	0.5	*support vector regression for censored targets*
LR-LOG-ALL-AVE100-MED	Yes	No	Estimated time	1	Use median survival time
LR-LOG-ALL-AVE100-AVE	Yes	No	Estimated time	1	Use average survival time
LR-LOG-ALL-AVE100-LIN	Yes	No	Estimated time	1	*linear regression*
LR-LOG-ALL-AVE100-SVR	Yes	No	Estimated time	1	*support vector regression*
LR-LOG-ALL-AVE100-RT	Yes	No	Estimated time	1	*regression trees*
LR-LOG-ALL-AVE100-GAT	Yes	No	Estimated time	1	*gating regression*
LR-LOG-ALL-CEN100-LINc	Yes	No	Censored Time	1	*linear regression for censored targets*

Continued on next page

Table B.2 – continued from previous page

Methods	Log-space	Outlier Detection	Class label of censored data	Censored Weights	Algorithm
LR-LOG-ALL-CEN100-SVRc	Yes	No	Censored Time	1	*support vector regression for censored targets*
LR-LOG-MCD-AVE050-MED	Yes	Yes	Estimated time	0.5	Use median survival time
LR-LOG-MCD-AVE050-AVE	Yes	Yes	Estimated time	0.5	Use average survival time
LR-LOG-MCD-AVE050-LIN	Yes	Yes	Estimated time	0.5	*linear regression*
LR-LOG-MCD-AVE050-SVR	Yes	Yes	Estimated time	0.5	*support vector regression*
LR-LOG-MCD-AVE050-RT	Yes	Yes	Estimated time	0.5	*regression trees*
LR-LOG-MCD-AVE050-GAT	Yes	Yes	Estimated time	0.5	*gating regression*
LR-LOG-MCD-CEN050-LINc	Yes	Yes	Censored Time	0.5	*linear regression for censored targets*
LR-LOG-MCD-CEN050-SVRc	Yes	Yes	Censored Time	0.5	*support vector regression for censored targets*
LR-LOG-MCD-AVE100-MED	Yes	Yes	Estimated time	1	Use median survival time
LR-LOG-MCD-AVE100-AVE	Yes	Yes	Estimated time	1	Use average survival time
LR-LOG-MCD-AVE100-LIN	Yes	Yes	Estimated time	1	*linear regression*
LR-LOG-MCD-AVE100-SVR	Yes	Yes	Estimated time	1	*support vector regression*
LR-LOG-MCD-AVE100-RT	Yes	Yes	Estimated time	1	*regression trees*
LR-LOG-MCD-AVE100-GAT	Yes	Yes	Estimated time	1	*gating regression*
LR-LOG-MCD-CEN100-LINc	Yes	Yes	Censored Time	1	*linear regression for censored targets*
LR-LOG-MCD-CEN100-SVRc	Yes	Yes	Censored Time	1	*support vector regression for censored targets*
Grouping Method: *CART* with *gain-ratio* as splitting criterion					
GR-REG-ALL-AVE050-MED	No	No	Estimated time	0.5	Use median survival time
GR-REG-ALL-AVE050-AVE	No	No	Estimated time	0.5	Use average survival time
GR-REG-ALL-AVE050-LIN	No	No	Estimated time	0.5	*linear regression*
GR-REG-ALL-AVE050-SVR	No	No	Estimated time	0.5	*support vector regression*
GR-REG-ALL-AVE050-RT	No	No	Estimated time	0.5	*regression trees*
GR-REG-ALL-AVE050-GAT	No	No	Estimated time	0.5	*gating regression*
GR-REG-ALL-CEN050-LINc	No	No	Censored Time	0.5	*linear regression for censored targets*
GR-REG-ALL-CEN050-SVRc	No	No	Censored Time	0.5	*support vector regression for censored targets*
GR-REG-ALL-AVE100-MED	No	No	Estimated time	1	Use median survival time
GR-REG-ALL-AVE100-AVE	No	No	Estimated time	1	Use average survival time
GR-REG-ALL-AVE100-LIN	No	No	Estimated time	1	*linear regression*
GR-REG-ALL-AVE100-SVR	No	No	Estimated time	1	*support vector regression*
GR-REG-ALL-AVE100-RT	No	No	Estimated time	1	*regression trees*
GR-REG-ALL-AVE100-GAT	No	No	Estimated time	1	*gating regression*
GR-REG-ALL-CEN100-LINc	No	No	Censored Time	1	*linear regression for censored targets*
GR-REG-ALL-CEN100-SVRc	No	No	Censored Time	1	*support vector regression for censored targets*
GR-REG-MCD-AVE050-MED	No	Yes	Estimated time	0.5	Use median survival time
GR-REG-MCD-AVE050-AVE	No	Yes	Estimated time	0.5	Use average survival time
GR-REG-MCD-AVE050-LIN	No	Yes	Estimated time	0.5	*linear regression*
GR-REG-MCD-AVE050-SVR	No	Yes	Estimated time	0.5	*support vector regression*
GR-REG-MCD-AVE050-RT	No	Yes	Estimated time	0.5	*regression trees*
GR-REG-MCD-AVE050-GAT	No	Yes	Estimated time	0.5	*gating regression*
GR-REG-MCD-CEN050-LINc	No	Yes	Censored Time	0.5	*linear regression for censored targets*
GR-REG-MCD-CEN050-SVRc	No	Yes	Censored Time	0.5	*support vector regression for censored targets*
GR-REG-MCD-AVE100-MED	No	Yes	Estimated time	1	Use median survival time
GR-REG-MCD-AVE100-AVE	No	Yes	Estimated time	1	Use average survival time
GR-REG-MCD-AVE100-LIN	No	Yes	Estimated time	1	*linear regression*
GR-REG-MCD-AVE100-SVR	No	Yes	Estimated time	1	*support vector regression*
GR-REG-MCD-AVE100-RT	No	Yes	Estimated time	1	*regression trees*
GR-REG-MCD-AVE100-GAT	No	Yes	Estimated time	1	*gating regression*
GR-REG-MCD-CEN100-LINc	No	Yes	Censored Time	1	*linear regression for censored targets*
GR-REG-MCD-CEN100-SVRc	No	Yes	Censored Time	1	*support vector regression for censored targets*
GR-LOG-ALL-AVE050-MED	Yes	No	Estimated time	0.5	Use median survival time
GR-LOG-ALL-AVE050-AVE	Yes	No	Estimated time	0.5	Use average survival time
GR-LOG-ALL-AVE050-LIN	Yes	No	Estimated time	0.5	*linear regression*
GR-LOG-ALL-AVE050-SVR	Yes	No	Estimated time	0.5	*support vector regression*
GR-LOG-ALL-AVE050-RT	Yes	No	Estimated time	0.5	*regression trees*
GR-LOG-ALL-AVE050-GAT	Yes	No	Estimated time	0.5	*gating regression*
GR-LOG-ALL-CEN050-LINc	Yes	No	Censored Time	0.5	*linear regression for censored targets*
GR-LOG-ALL-CEN050-SVRc	Yes	No	Censored Time	0.5	*support vector regression for censored targets*
GR-LOG-ALL-AVE100-MED	Yes	No	Estimated time	1	Use median survival time
GR-LOG-ALL-AVE100-AVE	Yes	No	Estimated time	1	Use average survival time
GR-LOG-ALL-AVE100-LIN	Yes	No	Estimated time	1	*linear regression*
GR-LOG-ALL-AVE100-SVR	Yes	No	Estimated time	1	*support vector regression*
GR-LOG-ALL-AVE100-RT	Yes	No	Estimated time	1	*regression trees*

Continued on next page

Table B.2 – continued from previous page

Methods	Log-space	Outlier Detection	Class label of censored data	Censored Weights	Algorithm
GR-LOG-ALL-AVE100-GAT	Yes	No	Estimated time	1	*gating regression*
GR-LOG-ALL-CEN100-LINc	Yes	No	Censored Time	1	*linear regression for censored targets*
GR-LOG-ALL-CEN100-SVRc	Yes	No	Censored Time	1	*support vector regression for censored targets*
GR-LOG-MCD-AVE050-MED	Yes	Yes	Estimated time	0.5	Use median survival time
GR-LOG-MCD-AVE050-AVE	Yes	Yes	Estimated time	0.5	Use average survival time
GR-LOG-MCD-AVE050-LIN	Yes	Yes	Estimated time	0.5	*linear regression*
GR-LOG-MCD-AVE050-SVR	Yes	Yes	Estimated time	0.5	*support vector regression*
GR-LOG-MCD-AVE050-RT	Yes	Yes	Estimated time	0.5	*regression trees*
GR-LOG-MCD-AVE050-GAT	Yes	Yes	Estimated time	0.5	*gating regression*
GR-LOG-MCD-CEN050-LINc	Yes	Yes	Censored Time	0.5	*linear regression for censored targets*
GR-LOG-MCD-CEN050-SVRc	Yes	Yes	Censored Time	0.5	*support vector regression for censored targets*
GR-LOG-MCD-AVE100-MED	Yes	Yes	Estimated time	1	Use median survival time
GR-LOG-MCD-AVE100-AVE	Yes	Yes	Estimated time	1	Use average survival time
GR-LOG-MCD-AVE100-LIN	Yes	Yes	Estimated time	1	*linear regression*
GR-LOG-MCD-AVE100-SVR	Yes	Yes	Estimated time	1	*support vector regression*
GR-LOG-MCD-AVE100-RT	Yes	Yes	Estimated time	1	*regression trees*
GR-LOG-MCD-AVE100-GAT	Yes	Yes	Estimated time	1	*gating regression*
GR-LOG-MCD-CEN100-LINc	Yes	Yes	Censored Time	1	*linear regression for censored targets*
GR-LOG-MCD-CEN100-SVRc	Yes	Yes	Censored Time	1	*support vector regression for censored targets*

Table B.3: Experimental Results

Methods	*RAE*	*CI*	RAE^{95}	$LI_{p<12}$	$RAE_{p>12}$
BASELINE-MED	0.5891±0.01	0.5000±0.00	0.5638±0.01	undefined	undefined
BASELINE-AVE	0.6081±0.00	0.5000±0.00	0.5875±0.00	undefined	undefined
HANDCRAFT-LIN	0.5942±0.01	0.5090±0.01	0.5649±0.01	4.8967±1.08	0.5863±0.01
HANDCRAFT-SVR	0.6067±0.00	0.5090±0.01	0.5859±0.00	9.9984±4.00	9.7233±4.55
NG-CEN100-ALL-REG-LIN	0.5592±0.02	0.7442±0.01	0.5220±0.02	4.2071±0.25	0.5403±0.01
NG-CEN100-ALL-REG-SVR	0.5582±0.03	0.7434±0.01	0.5287±0.03	4.9575±0.41	0.5434±0.02
NG-CEN100-ALL-REG-RT	0.5601±0.02	0.7483±0.02	0.5266±0.01	4.7212±0.81	0.5438±0.01
NG-REG-ALL-AVE050-LIN	0.5524±0.02	0.7498±0.01	0.5102±0.02	4.1004±0.22	0.5257±0.01
NG-REG-ALL-AVE050-SVR	0.5538±0.04	0.7493±0.01	0.4980±0.02	4.1903±0.39	0.5152±0.02
NG-REG-ALL-AVE050-RT	0.5630±0.04	0.7517±0.01	0.5225±0.02	5.1055±1.06	0.5281±0.02
NG-REG-ALL-AVE050-MED	0.5912±0.01	0.5000±0.00	0.5667±0.01	undefined	undefined
NG-REG-ALL-CEN050-LINc	0.6252±0.04	0.7497±0.01	0.5526±0.03	4.0587±0.39	0.5321±0.02
NG-REG-ALL-CEN050-SVRc	0.5890±0.03	0.7499±0.01	0.5513±0.02	4.5743±0.35	0.5661±0.02
NG-REG-ALL-AVE050-AVE	0.6114±0.00	0.5000±0.00	0.5914±0.01	undefined	undefined
NG-REG-ALL-AVE050-GAT	0.5622±0.02	0.7448±0.01	0.5152±0.02	4.5236±0.46	0.5274±0.02
NG-REG-ALL-AVE100-LIN	0.5621±0.02	0.7446±0.01	0.5278±0.02	4.3505±0.20	0.5424±0.01
NG-REG-ALL-AVE100-SVR	0.5468±0.03	0.7432±0.01	0.5092±0.02	3.9596±0.47	0.5267±0.02
NG-REG-ALL-AVE100-RT	0.5717±0.03	0.7452±0.02	0.5377±0.02	5.0832±0.89	0.5510±0.02
NG-REG-ALL-AVE100-MED	0.5912±0.01	0.5000±0.00	0.5667±0.01	undefined	undefined
NG-REG-ALL-CEN100-LINc	0.5589±0.02	0.7445±0.01	0.5215±0.02	4.1591±0.29	0.5405±0.01
NG-REG-ALL-CEN100-SVRc	0.5751±0.03	0.7499±0.01	0.5409±0.03	4.6095±0.54	0.5590±0.03
NG-REG-ALL-AVE100-AVE	0.6114±0.00	0.5000±0.00	0.5914±0.01	undefined	undefined
NG-REG-ALL-AVE100-GAT	0.5669±0.01	0.7456±0.01	0.5315±0.01	4.5456±0.51	0.5470±0.01
NG-REG-MCD-AVE050-LIN	0.5506±0.03	0.7498±0.01	0.5085±0.02	4.1940±0.43	0.5227±0.02
NG-REG-MCD-AVE050-SVR	0.5503±0.04	0.7501±0.01	0.4953±0.02	4.1312±0.40	0.5125±0.03
NG-REG-MCD-AVE050-RT	0.5619±0.03	0.7552±0.01	0.5180±0.02	5.2004±0.66	0.5175±0.02
NG-REG-MCD-AVE050-MED	0.5893±0.00	0.5000±0.00	0.5650±0.00	undefined	undefined
NG-REG-MCD-CEN050-LINc	0.6026±0.04	0.7575±0.01	0.5357±0.03	4.0014±0.36	0.5169±0.03
NG-REG-MCD-CEN050-SVRc	0.5510±0.03	0.7547±0.01	0.5137±0.03	4.1875±0.47	0.5235±0.03
NG-REG-MCD-AVE050-AVE	0.6091±0.01	0.5000±0.00	0.5888±0.01	undefined	undefined
NG-REG-MCD-AVE050-GAT	0.5556±0.02	0.7502±0.01	0.5107±0.02	4.9558±0.31	0.5125±0.02

Continued on next page

Table B.3 – continued from previous page

Methods	*RAE*	*CI*	RAE^{95}	$LI_{p<12}$	$RAE_{p>12}$
NG-REG-MCD-AVE100-LIN	0.5586±0.02	0.7421±0.01	0.5256±0.01	4.2028±0.38	0.5454±0.01
NG-REG-MCD-AVE100-SVR	0.5479±0.03	0.7418±0.01	0.5074±0.02	3.9782±0.43	0.5267±0.02
NG-REG-MCD-AVE100-RT	0.5688±0.03	0.7452±0.02	0.5348±0.02	4.5674±0.79	0.5545±0.02
NG-REG-MCD-AVE100-MED	0.5900±0.01	0.5000±0.00	0.5654±0.01	undefined	undefined
NG-REG-MCD-CEN100-LINc	0.5506±0.02	0.7518±0.01	0.5191±0.01	4.1870±0.34	0.5355±0.01
NG-REG-MCD-CEN100-SVRc	0.5766±0.02	0.7530±0.01	0.5477±0.01	4.8107±0.38	0.5621±0.01
NG-REG-MCD-AVE100-AVE	0.6114±0.01	0.5000±0.00	0.5915±0.01	undefined	undefined
NG-REG-MCD-AVE100-GAT	0.5618±0.01	0.7483±0.02	0.5296±0.01	4.6106±0.59	0.5427±0.02
NG-LOG-ALL-AVE050-LIN	0.5568±0.05	0.7698±0.01	0.5010±0.04	4.4688±0.51	0.4937±0.04
NG-LOG-ALL-AVE050-SVR	0.5535±0.04	0.7658±0.01	0.5001±0.03	4.3953±0.48	0.4904±0.03
NG-LOG-ALL-AVE050-RT	0.5587±0.04	0.7591±0.01	0.5039±0.03	4.4559±0.33	0.5064±0.05
NG-LOG-ALL-AVE050-MED	0.5956±0.00	0.5000±0.00	0.5727±0.00	undefined	undefined
NG-LOG-ALL-CEN050-LINc	0.6902±0.08	0.7666±0.01	0.5990±0.06	4.6232±0.68	0.5256±0.04
NG-LOG-ALL-CEN050-SVRc	0.5725±0.03	0.7629±0.01	0.5329±0.02	4.3343±0.38	0.5506±0.03
NG-LOG-ALL-AVE050-AVE	0.5841±0.01	0.5000±0.00	0.5531±0.01	undefined	undefined
NG-LOG-ALL-AVE050-GAT	0.5956±0.00	0.5000±0.00	0.5727±0.00	undefined	undefined
NG-LOG-ALL-AVE100-LIN	0.5380±0.04	0.7679±0.01	0.4929±0.03	4.4588±0.42	0.4955±0.03
NG-LOG-ALL-AVE100-SVR	0.5393±0.03	0.7641±0.01	0.4967±0.03	4.2782±0.32	0.5055±0.03
NG-LOG-ALL-AVE100-RT	0.5418±0.03	0.7580±0.01	0.4966±0.02	4.5625±0.36	0.5014±0.04
NG-LOG-ALL-AVE100-MED	0.5956±0.00	0.5000±0.00	0.5727±0.00	undefined	undefined
NG-LOG-ALL-CEN100-LINc	0.5462±0.04	0.7678±0.01	0.4968±0.03	4.4718±0.49	0.4909±0.04
NG-LOG-ALL-CEN100-SVRc	0.5758±0.03	0.7624±0.01	0.5388±0.02	4.3558±0.33	0.5580±0.03
NG-LOG-ALL-AVE100-AVE	0.5841±0.01	0.5000±0.00	0.5531±0.01	undefined	undefined
NG-LOG-ALL-AVE100-GAT	0.5392±0.03	0.7640±0.01	0.4952±0.02	4.2810±0.32	0.5049±0.03
NG-LOG-MCD-AVE050-LIN	0.5548±0.05	0.7687±0.01	0.5005±0.04	4.4726±0.60	0.4988±0.04
NG-LOG-MCD-AVE050-SVR	0.5513±0.04	0.7650±0.01	0.4979±0.03	4.3860±0.62	0.4950±0.03
NG-LOG-MCD-AVE050-RT	0.5682±0.05	0.7595±0.01	0.5153±0.04	4.6984±0.64	0.5105±0.03
NG-LOG-MCD-AVE050-MED	0.5939±0.01	0.5000±0.00	0.5707±0.01	undefined	undefined
NG-LOG-MCD-CEN050-LINc	0.6870±0.09	0.7645±0.01	0.5964±0.07	4.6198±0.75	0.5329±0.05
NG-LOG-MCD-CEN050-SVRc	0.5755±0.04	0.7614±0.01	0.5335±0.03	4.3568±0.40	0.5531±0.04
NG-LOG-MCD-AVE050-AVE	0.5843±0.01	0.5000±0.00	0.5530±0.01	undefined	undefined
NG-LOG-MCD-AVE050-GAT	0.5939±0.01	0.5000±0.00	0.5707±0.01	undefined	undefined
NG-LOG-MCD-AVE100-LIN	0.5385±0.04	0.7649±0.01	0.4926±0.03	4.5260±0.46	0.4968±0.04
NG-LOG-MCD-AVE100-SVR	0.5376±0.03	0.7612±0.01	0.4955±0.03	4.3355±0.40	0.5064±0.03
NG-LOG-MCD-AVE100-RT	0.5536±0.03	0.7536±0.01	0.5073±0.02	4.6130±0.39	0.5140±0.02
NG-LOG-MCD-AVE100-MED	0.5956±0.01	0.5000±0.00	0.5725±0.01	undefined	undefined
NG-LOG-MCD-CEN100-LINc	0.5468±0.05	0.7650±0.01	0.4964±0.04	4.3729±0.53	0.5013±0.04
NG-LOG-MCD-CEN100-SVRc	0.5755±0.04	0.7600±0.01	0.5347±0.03	4.3591±0.41	0.5555±0.03
NG-LOG-MCD-AVE100-AVE	0.5828±0.01	0.5000±0.00	0.5522±0.01	undefined	undefined
NG-LOG-MCD-AVE100-GAT	0.5376±0.03	0.7612±0.01	0.4955±0.03	4.3355±0.39	0.5063±0.03
LR-REG-ALL-AVE050-LIN	0.5598±0.02	0.7421±0.01	0.5168±0.02	4.1554±0.32	0.5307±0.01
LR-REG-ALL-AVE050-SVR	0.5746±0.04	0.7400±0.01	0.5107±0.01	4.3420±0.19	0.5157±0.03
LR-REG-ALL-AVE050-RT	0.5679±0.02	0.7470±0.02	0.5265±0.02	5.1883±0.60	0.5297±0.02
LR-REG-ALL-AVE050-MED	0.6019±0.01	0.5251±0.01	0.5693±0.01	6.1599±0.65	0.5871±0.01
LR-REG-ALL-CEN050-LINc	0.6359±0.02	0.7453±0.01	0.5650±0.02	4.2453±0.43	0.5358±0.01
LR-REG-ALL-CEN050-SVRc	0.5801±0.02	0.7426±0.01	0.5307±0.01	4.2794±0.48	0.5378±0.03
LR-REG-ALL-AVE050-AVE	0.6120±0.01	0.5252±0.01	0.5887±0.01	9.1863±3.76	7.4443±5.58
LR-REG-ALL-AVE050-GAT	0.5705±0.01	0.7476±0.02	0.5213±0.01	4.6896±0.24	0.5300±0.02
LR-REG-ALL-AVE100-LIN	0.5649±0.02	0.7378±0.02	0.5312±0.02	4.0842±0.35	0.5495±0.01
LR-REG-ALL-AVE100-SVR	0.5674±0.03	0.7360±0.01	0.5238±0.02	4.0757±0.39	0.5337±0.02
LR-REG-ALL-AVE100-RT	0.5765±0.01	0.7393±0.02	0.5436±0.01	4.8189±0.39	0.5568±0.01
LR-REG-ALL-AVE100-MED	0.6019±0.01	0.5251±0.01	0.5693±0.01	6.1599±0.65	0.5871±0.01
LR-REG-ALL-CEN100-LINc	0.5618±0.02	0.7410±0.01	0.5252±0.02	3.9586±0.41	0.5480±0.01
LR-REG-ALL-CEN100-SVRc	0.5907±0.03	0.7424±0.01	0.5404±0.02	4.4074±0.40	0.5465±0.02
LR-REG-ALL-AVE100-AVE	0.6120±0.01	0.5252±0.01	0.5887±0.01	9.1863±3.76	7.4443±5.58

Continued on next page

Table B.3 – continued from previous page

Methods	*RAE*	*CI*	RAE^{95}	$LI_{p<12}$	$RAE_{p>12}$
LR-REG-ALL-AVE100-GAT	0.5777±0.02	0.7462±0.02	0.5405±0.02	4.8822±0.31	0.5475±0.01
LR-REG-MCD-AVE050-LIN	0.5573±0.02	0.7408±0.01	0.5138±0.01	4.1938±0.38	0.5312±0.01
LR-REG-MCD-AVE050-SVR	0.5823±0.04	0.7380±0.01	0.5135±0.02	4.3787±0.32	0.5204±0.02
LR-REG-MCD-AVE050-RT	0.5659±0.02	0.7527±0.01	0.5224±0.02	5.3885±0.82	0.5253±0.01
LR-REG-MCD-AVE050-MED	0.5963±0.01	0.5203±0.01	0.5684±0.01	7.5716±2.23	2.8711±4.56
LR-REG-MCD-CEN050-LINc	0.6248±0.03	0.7499±0.01	0.5500±0.02	4.1771±0.43	0.5213±0.03
LR-REG-MCD-CEN050-SVRc	0.5917±0.04	0.7447±0.01	0.5404±0.02	4.6512±0.52	0.5465±0.03
LR-REG-MCD-AVE050-AVE	0.6095±0.01	0.5203±0.01	0.5895±0.01	undefined	undefined
LR-REG-MCD-AVE050-GAT	0.5662±0.02	0.7504±0.02	0.5189±0.01	4.9426±0.41	0.5264±0.02
LR-REG-MCD-AVE100-LIN	0.5660±0.02	0.7380±0.01	0.5308±0.02	4.1544±0.35	0.5469±0.01
LR-REG-MCD-AVE100-SVR	0.5675±0.04	0.7352±0.01	0.5179±0.02	4.0624±0.52	0.5328±0.02
LR-REG-MCD-AVE100-RT	0.5703±0.01	0.7412±0.02	0.5361±0.01	4.7079±0.60	0.5526±0.01
LR-REG-MCD-AVE100-MED	0.5955±0.01	0.5199±0.01	0.5670±0.00	7.5588±2.28	2.8694±4.57
LR-REG-MCD-CEN100-LINc	0.5595±0.02	0.7480±0.01	0.5223±0.01	4.1206±0.29	0.5364±0.01
LR-REG-MCD-CEN100-SVRc	0.5879±0.04	0.7467±0.01	0.5408±0.03	4.5890±0.67	0.5545±0.03
LR-REG-MCD-AVE100-AVE	0.6078±0.01	0.5198±0.01	0.5871±0.01	11.5172±0.97	9.7198±4.56
LR-REG-MCD-AVE100-GAT	0.5660±0.01	0.7493±0.02	0.5321±0.01	4.5930±0.72	0.5448±0.02
LR-LOG-ALL-AVE050-LIN	0.5677±0.05	0.7596±0.01	0.5087±0.04	4.4999±0.49	0.5007±0.04
LR-LOG-ALL-AVE050-SVR	0.5709±0.04	0.7540±0.02	0.5126±0.03	4.3872±0.33	0.5042±0.04
LR-LOG-ALL-AVE050-RT	0.5839±0.04	0.7575±0.01	0.5239±0.03	4.7084±0.52	0.5170±0.03
LR-LOG-ALL-AVE050-MED	0.6053±0.01	0.5251±0.01	0.5744±0.01	6.1516±0.66	0.5927±0.01
LR-LOG-ALL-CEN050-LINc	0.7093±0.08	0.7624±0.01	0.6135±0.07	4.7654±0.73	0.5297±0.03
LR-LOG-ALL-CEN050-SVRc	0.5844±0.03	0.7545±0.01	0.5400±0.02	4.4038±0.31	0.5414±0.03
LR-LOG-ALL-AVE050-AVE	0.5925±0.01	0.5253±0.01	0.5560±0.01	5.8892±0.62	0.5784±0.01
LR-LOG-ALL-AVE050-GAT	0.6016±0.01	0.5271±0.01	0.5720±0.01	4.9960±1.36	0.5924±0.01
LR-LOG-ALL-AVE100-LIN	0.5532±0.04	0.7588±0.01	0.5032±0.03	4.5052±0.44	0.5017±0.03
LR-LOG-ALL-AVE100-SVR	0.5607±0.03	0.7535±0.01	0.5095±0.02	4.3449±0.25	0.5158±0.03
LR-LOG-ALL-AVE100-RT	0.5666±0.03	0.7552±0.01	0.5145±0.02	4.7231±0.49	0.5162±0.03
LR-LOG-ALL-AVE100-MED	0.6053±0.01	0.5251±0.01	0.5744±0.01	6.1516±0.66	0.5927±0.01
LR-LOG-ALL-CEN100-LINc	0.5578±0.04	0.7635±0.01	0.5064±0.03	4.4927±0.47	0.4982±0.03
LR-LOG-ALL-CEN100-SVRc	0.5835±0.03	0.7547±0.01	0.5400±0.02	4.3538±0.28	0.5449±0.03
LR-LOG-ALL-AVE100-AVE	0.5925±0.01	0.5253±0.01	0.5560±0.01	5.8892±0.62	0.5784±0.01
LR-LOG-ALL-AVE100-GAT	0.5480±0.03	0.7591±0.01	0.5049±0.02	4.3366±0.25	0.5128±0.03
LR-LOG-MCD-AVE050-LIN	0.5676±0.05	0.7650±0.01	0.5085±0.04	4.5858±0.61	0.4992±0.04
LR-LOG-MCD-AVE050-SVR	0.5729±0.04	0.7582±0.01	0.5129±0.03	4.4462±0.37	0.5025±0.03
LR-LOG-MCD-AVE050-RT	0.5749±0.03	0.7583±0.01	0.5132±0.02	4.7314±0.29	0.5064±0.03
LR-LOG-MCD-AVE050-MED	0.6054±0.01	0.5180±0.01	0.5746±0.01	8.4429±4.26	0.5936±0.01
LR-LOG-MCD-CEN050-LINc	0.7219±0.08	0.7630±0.01	0.6175±0.06	4.9132±0.94	0.5288±0.02
LR-LOG-MCD-CEN050-SVRc	0.5952±0.04	0.7564±0.01	0.5486±0.04	4.3462±0.35	0.5584±0.04
LR-LOG-MCD-AVE050-AVE	0.5927±0.01	0.5180±0.01	0.5567±0.01	8.3889±4.48	0.5808±0.01
LR-LOG-MCD-AVE050-GAT	0.6006±0.01	0.5193±0.01	0.5700±0.01	7.8540±4.85	0.5914±0.01
LR-LOG-MCD-AVE100-LIN	0.5518±0.04	0.7581±0.01	0.4992±0.03	4.5228±0.49	0.4984±0.03
LR-LOG-MCD-AVE100-SVR	0.5626±0.04	0.7548±0.01	0.5057±0.02	4.4316±0.28	0.5108±0.03
LR-LOG-MCD-AVE100-RT	0.5616±0.03	0.7533±0.01	0.5106±0.02	4.9125±0.52	0.5116±0.02
LR-LOG-MCD-AVE100-MED	0.6007±0.01	0.5180±0.01	0.5721±0.01	7.8465±2.48	0.5917±0.00
LR-LOG-MCD-CEN100-LINc	0.5601±0.04	0.7603±0.01	0.5058±0.03	4.4828±0.44	0.5007±0.04
LR-LOG-MCD-CEN100-SVRc	0.6044±0.04	0.7530±0.01	0.5507±0.02	4.6020±0.44	0.5563±0.02
LR-LOG-MCD-AVE100-AVE	0.5878±0.01	0.5182±0.01	0.5531±0.01	7.4040±2.70	0.5772±0.01
LR-LOG-MCD-AVE100-GAT	0.5533±0.03	0.7577±0.01	0.5043±0.02	4.5273±0.23	0.5098±0.03
GR-REG-ALL-AVE050-LIN	0.5874±0.04	0.7296±0.02	0.5389±0.03	4.3272±0.81	0.5423±0.02
GR-REG-ALL-AVE050-SVR	0.6277±0.06	0.7237±0.01	0.5518±0.03	4.6823±0.85	0.5327±0.01
GR-REG-ALL-AVE050-RT	0.5689±0.02	0.7305±0.01	0.5263±0.02	5.1001±0.51	0.5312±0.01
GR-REG-ALL-AVE050-MED	0.5897±0.00	0.5576±0.02	0.5642±0.00	undefined	undefined
GR-REG-ALL-CEN050-LINc	0.6826±0.05	0.7249±0.02	0.5998±0.04	4.7162±0.70	0.5484±0.02
GR-REG-ALL-CEN050-SVRc	0.6259±0.07	0.7182±0.01	0.5665±0.04	4.6510±0.90	0.5667±0.01

Continued on next page

Table B.3 – continued from previous page

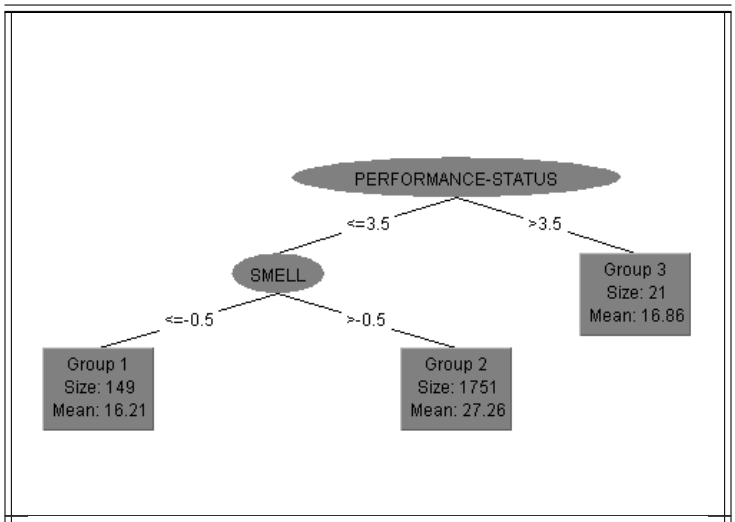
Methods	*RAE*	*CI*	RAE^{95}	$LI_{p<12}$	$RAE_{p>12}$
GR-REG-ALL-AVE050-AVE	0.6075±0.00	0.5823±0.01	0.5873±0.00	undefined	undefined
GR-REG-ALL-AVE050-GAT	0.5797±0.02	0.7266±0.02	0.5335±0.02	4.5583±0.44	0.5450±0.02
GR-REG-ALL-AVE100-LIN	0.5848±0.02	0.7267±0.02	0.5467±0.02	4.2784±0.54	0.5622±0.02
GR-REG-ALL-AVE100-SVR	0.6180±0.04	0.7183±0.01	0.5546±0.02	4.5905±0.53	0.5555±0.01
GR-REG-ALL-AVE100-RT	0.5827±0.01	0.7302±0.01	0.5464±0.01	5.2116±0.46	0.5554±0.01
GR-REG-ALL-AVE100-MED	0.5897±0.00	0.5576±0.02	0.5642±0.00	undefined	undefined
GR-REG-ALL-CEN100-LINc	0.5852±0.03	0.7269±0.02	0.5445±0.02	4.2130±0.71	0.5576±0.02
GR-REG-ALL-CEN100-SVRc	0.6240±0.05	0.7190±0.01	0.5570±0.02	4.6325±0.63	0.5506±0.01
GR-REG-ALL-AVE100-AVE	0.6075±0.00	0.5823±0.01	0.5873±0.00	undefined	undefined
GR-REG-ALL-AVE100-GAT	0.5822±0.02	0.7252±0.01	0.5412±0.02	4.3555±0.51	0.5630±0.01
GR-REG-MCD-AVE050-LIN	0.5811±0.03	0.7329±0.02	0.5341±0.03	4.2990±0.60	0.5405±0.02
GR-REG-MCD-AVE050-SVR	0.6260±0.07	0.7262±0.01	0.5471±0.04	4.4300±0.67	0.5358±0.02
GR-REG-MCD-AVE050-RT	0.5700±0.03	0.7309±0.01	0.5281±0.03	5.0089±0.41	0.5318±0.02
GR-REG-MCD-AVE050-MED	0.5896±0.00	0.5599±0.02	0.5643±0.00	undefined	undefined
GR-REG-MCD-CEN050-LINc	0.6514±0.05	0.7356±0.02	0.5774±0.04	4.4634±0.65	0.5368±0.02
GR-REG-MCD-CEN050-SVRc	0.6122±0.05	0.7241±0.01	0.5619±0.03	4.7127±0.82	0.5564±0.02
GR-REG-MCD-AVE050-AVE	0.6072±0.00	0.5737±0.02	0.5865±0.00	undefined	undefined
GR-REG-MCD-AVE050-GAT	0.5983±0.02	0.7178±0.01	0.5424±0.01	4.5478±0.60	0.5553±0.01
GR-REG-MCD-AVE100-LIN	0.5863±0.02	0.7247±0.01	0.5436±0.02	4.1399±0.60	0.5641±0.01
GR-REG-MCD-AVE100-SVR	0.6011±0.04	0.7194±0.01	0.5483±0.03	4.5700±0.76	0.5504±0.02
GR-REG-MCD-AVE100-RT	0.5829±0.01	0.7263±0.01	0.5486±0.01	5.2569±0.59	0.5577±0.01
GR-REG-MCD-AVE100-MED	0.5902±0.00	0.5595±0.02	0.5648±0.00	undefined	undefined
GR-REG-MCD-CEN100-LINc	0.5859±0.04	0.7329±0.01	0.5381±0.03	4.3004±0.88	0.5558±0.02
GR-REG-MCD-CEN100-SVRc	0.6303±0.10	0.7216±0.01	0.5565±0.07	4.4155±0.96	0.5482±0.02
GR-REG-MCD-AVE100-AVE	0.6089±0.01	0.5783±0.02	0.5888±0.01	undefined	undefined
GR-REG-MCD-AVE100-GAT	0.5935±0.02	0.7249±0.01	0.5485±0.01	4.4011±0.51	0.5625±0.01
GR-LOG-ALL-AVE050-LIN	0.5864±0.06	0.7545±0.02	0.5194±0.04	4.6709±0.50	0.5128±0.03
GR-LOG-ALL-AVE050-SVR	0.6022±0.07	0.7478±0.02	0.5325±0.04	4.7410±0.46	0.5132±0.03
GR-LOG-ALL-AVE050-RT	0.5804±0.05	0.7452±0.02	0.5207±0.04	4.6454±0.49	0.5170±0.03
GR-LOG-ALL-AVE050-MED	0.5918±0.00	0.5554±0.02	0.5679±0.00	undefined	undefined
GR-LOG-ALL-CEN050-LINc	0.7333±0.09	0.7430±0.02	0.6355±0.07	4.9521±0.66	0.5530±0.04
GR-LOG-ALL-CEN050-SVRc	0.6470±0.08	0.7357±0.02	0.5784±0.05	4.7968±0.73	0.5678±0.03
GR-LOG-ALL-AVE050-AVE	0.5845±0.01	0.5863±0.03	0.5524±0.01	undefined	undefined
GR-LOG-ALL-AVE050-GAT	0.5918±0.00	0.5554±0.02	0.5679±0.00	undefined	undefined
GR-LOG-ALL-AVE100-LIN	0.5657±0.05	0.7535±0.02	0.5127±0.04	4.5285±0.48	0.5178±0.03
GR-LOG-ALL-AVE100-SVR	0.5829±0.05	0.7458±0.02	0.5258±0.03	4.5575±0.51	0.5311±0.03
GR-LOG-ALL-AVE100-RT	0.5697±0.03	0.7424±0.02	0.5179±0.03	4.7563±0.37	0.5134±0.02
GR-LOG-ALL-AVE100-MED	0.5918±0.00	0.5554±0.02	0.5679±0.00	undefined	undefined
GR-LOG-ALL-CEN100-LINc	0.5776±0.05	0.7532±0.02	0.5167±0.04	4.6082±0.46	0.5172±0.03
GR-LOG-ALL-CEN100-SVRc	0.5946±0.06	0.7460±0.02	0.5299±0.04	4.6517±0.59	0.5250±0.03
GR-LOG-ALL-AVE100-AVE	0.5845±0.01	0.5863±0.03	0.5524±0.01	undefined	undefined
GR-LOG-ALL-AVE100-GAT	0.6071±0.03	0.6718±0.04	0.5528±0.01	5.0383±0.42	0.5631±0.02
GR-LOG-MCD-AVE050-LIN	0.5838±0.06	0.7561±0.02	0.5190±0.04	4.7824±0.72	0.5047±0.02
GR-LOG-MCD-AVE050-SVR	0.5978±0.06	0.7449±0.01	0.5288±0.04	4.8247±0.50	0.5042±0.03
GR-LOG-MCD-AVE050-RT	0.5872±0.04	0.7430±0.02	0.5241±0.04	4.8171±0.54	0.5152±0.04
GR-LOG-MCD-AVE050-MED	0.5909±0.00	0.5596±0.02	0.5671±0.01	undefined	undefined
GR-LOG-MCD-CEN050-LINc	0.7387±0.09	0.7479±0.02	0.6375±0.07	4.9236±0.83	0.5589±0.04
GR-LOG-MCD-CEN050-SVRc	0.6246±0.07	0.7387±0.02	0.5674±0.05	4.6752±0.55	0.5617±0.04
GR-LOG-MCD-AVE050-AVE	0.5826±0.01	0.5871±0.03	0.5504±0.01	undefined	undefined
GR-LOG-MCD-AVE050-GAT	0.5909±0.00	0.5596±0.02	0.5671±0.01	undefined	undefined
GR-LOG-MCD-AVE100-LIN	0.5593±0.04	0.7460±0.02	0.5056±0.04	4.5079±0.65	0.5196±0.03
GR-LOG-MCD-AVE100-SVR	0.5751±0.04	0.7388±0.02	0.5221±0.03	4.6645±0.64	0.5334±0.02
GR-LOG-MCD-AVE100-RT	0.5599±0.03	0.7390±0.02	0.5055±0.03	4.7115±0.47	0.5165±0.02
GR-LOG-MCD-AVE100-MED	0.5840±0.00	0.5629±0.03	0.5595±0.01	undefined	undefined
GR-LOG-MCD-CEN100-LINc	0.5709±0.05	0.7468±0.02	0.5085±0.04	4.7050±0.65	0.5139±0.02
GR-LOG-MCD-CEN100-SVRc	0.5890±0.05	0.7382±0.02	0.5274±0.04	4.6331±0.58	0.5324±0.03

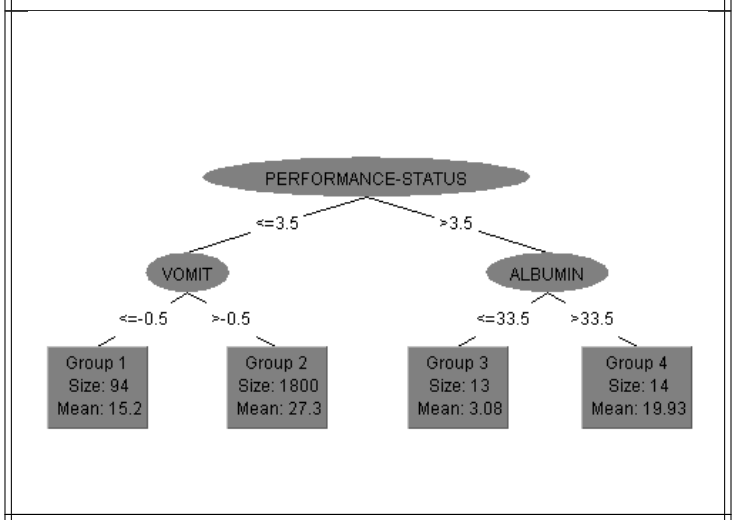
Continued on next page

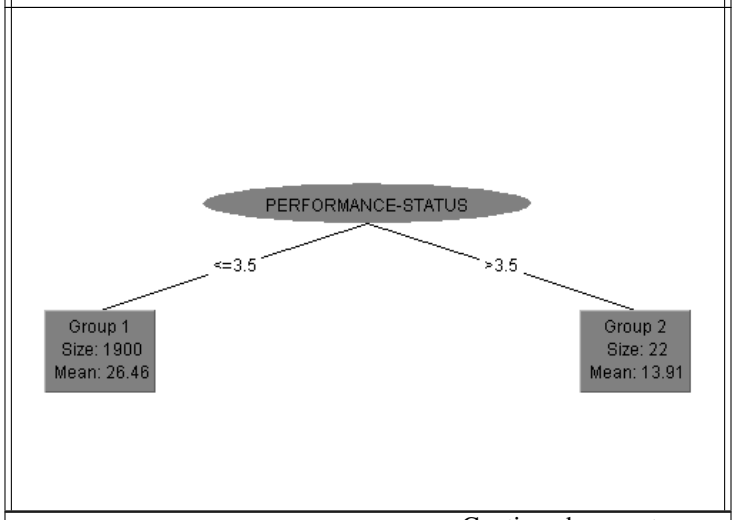
Table B.3 – continued from previous page

Methods	*RAE*	*CI*	RAE^{95}	$LI_{p<12}$	$RAE_{p>12}$
GR-LOG-MCD-AVE100-AVE	0.5776±0.00	0.5989±0.04	0.5453±0.01	undefined	undefined
GR-LOG-MCD-AVE100-GAT	0.5856±0.03	0.6689±0.04	0.5419±0.02	4.7328±0.62	0.5582±0.02

Table B.4: A Visualization of a decision tree

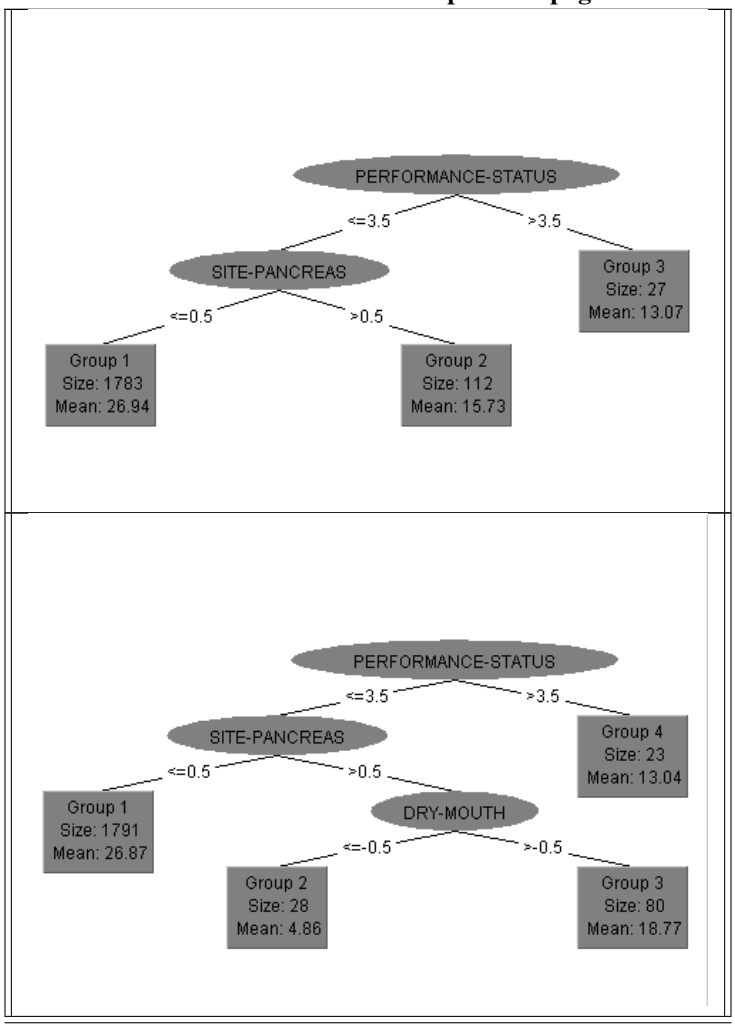






Continued on next page

Table B.4 – continued from previous page



B.3 Best Results

Table B.5: Top 10 models

Statistical Results of the Top 10 models					
Regressor	*RAE*	*CI*	RAE^{95}	$LI_{p<12}$	$RAE_{p>12}$
NG-LOG-MCD-AVE100-GAT	0.5376±0.03	0.7612±0.01	0.4955±0.03	4.3355±0.39	0.5063±0.03
NG-LOG-MCD-AVE100-SVR	0.5376±0.03	0.7612±0.01	0.4955±0.03	4.3355±0.40	0.5064±0.03
NG-LOG-ALL-AVE100-LIN	0.5380±0.04	0.7679±0.01	0.4929±0.03	4.4588±0.42	0.4955±0.03
NG-LOG-MCD-AVE100-LIN	0.5385±0.04	0.7649±0.01	0.4926±0.03	4.5260±0.46	0.5068±0.04
NG-LOG-ALL-AVE100-GAT	0.5392±0.03	0.7640±0.01	0.4952±0.02	4.2810±0.32	0.5049±0.03
NG-LOG-ALL-AVE100-SVR	0.5393±0.03	0.7641±0.01	0.4967±0.03	4.2782±0.32	0.5055±0.03
NG-LOG-ALL-AVE100-RT	0.5418±0.03	0.7580±0.01	0.4966±0.02	4.5625±0.36	0.5014±0.04
NG-LOG-ALL-CEN100-LINc	0.5462±0.04	0.7678±0.01	0.4968±0.03	4.4718±0.49	0.4909±0.04
NG-LOG-MCD-AVE100-LINc	0.5468±0.05	0.7650±0.01	0.4964±0.04	4.3729±0.53	0.5013±0.04
NG-REG-ALL-AVE100-SVR	0.5468±0.03	0.7432±0.01	0.5092±0.02	3.9596±0.47	0.5267±0.02

Generalization of the Best Combination
Grouping Method: No grouping
Log-space Transformation: Yes
Outliers Detection: MCD (*Mahalanobis Distance* with *MCD estimator*)
Handling Censored Data: take average survival time of uncensored patients in the risk set
Learning Algorithm: *gating regression*, *support vector regression*, or *linear regression*

Table B.7: Parameters

β	X
	Linear Regression
2.34	GENDER=FEMALE
0.65	BOX2-SCORE
-1.73	PERFORMANCE-STATUS-2
-2.87	PERFORMANCE-STATUS-3
-4.99	PERFORMANCE-STATUS-4
0.19	BMI
-1.46	NO-PROBLEM
1.5	NO-APPETITE
1.64	PAIN
2.19	DRY-MOUTH
-0.39	WEIGHT-CHANGEPOINT
-8.68	SITE-BRUNCHUS-LUNG
2.52	SITE-COLORECTAL
2.56	SITE-HEAD-AND-NECK
-4.73	SITE-PANCREAS
16.71	STAGE-1
17.89	STAGE-2
12.77	STAGE-3
-0.13	AGE
-0.8	GRANULOCYTES
0	LDH-SERUM
0.15	WBC-COUNT
0.04	HGB
0.36	ALBUMIN
8.59	
	Support Vector Regression
0.08	(standardized) GENDER
-0.04	(standardized) BOX1-SCORE
0.02	(standardized) BOX2-SCORE
-0.02	(standardized) PERFORMANCE-STATUS-1
-0.02	(standardized) PERFORMANCE-STATUS-2
-0.05	(standardized) PERFORMANCE-STATUS-3
-0.03	(standardized) PERFORMANCE-STATUS-4
0.06	(standardized) BMI
-0.06	(standardized) NO-PROBLEM
0.06	(standardized) NO-APPETITE
-0.01	(standardized) NAUSEA
-0.03	(standardized) CONSTIPATION
0	(standardized) SORE-MOUTH
0	(standardized) TASTE-FUNNY
0.03	(standardized) SMELL
0	(standardized) SWALLOW
-0.02	(standardized) FEEL-FULL
0.05	(standardized) PAIN
-0.01	(standardized) OTHER
0.01	(standardized) VOMIT
0.01	(standardized) DIARRHEA
0.04	(standardized) DRY-MOUTH

Continued on next page

Table B.7 – continued from previous page

β	X
0.04	(standardized) DENTAL-PROBLEM
-0.01	(standardized) AGE65
0.03	(standardized) WEIGHT-CHANGEPOINT
-0.16	(standardized) SITE-BRUNCHUS-LUNG
0.14	(standardized) SITE-COLORECTAL
0.1	(standardized) SITE-HEAD-AND-NECK
0	(standardized) SITE-ESOPHAGUS
-0.07	(standardized) SITE-PANCREAS
-0.01	(standardized) SITE-STOMACH
0	(standardized) SITE-OTHER-DIGESTIVE
0	(standardized) MISC
0.15	(standardized) STAGE-1
0.31	(standardized) STAGE-2
0.25	(standardized) STAGE-3
-0.07	(standardized) AGE
-0.12	(standardized) GRANULOCYTES
-0.09	(standardized) LDH-SERUM
-0.02	(standardized) LYMPHOCYTES
0.01	(standardized) PLATELET
0.01	(standardized) WBC-COUNT
-0.01	(standardized) CALCIUM-SERUM
0.03	(standardized) HGB
0	(standardized) CREATININE-SERUM
0.09	(standardized) ALBUMIN
-0.11	

Regression Trees	
G1: SITE-COLORECTAL == 0	
-0.47	BOX1-SCORE
0.65	BOX2-SCORE
0	BMI
-3.53	NO-PROBLEM
3.48	DRY-MOUTH
-0.53	WEIGHT-CHANGEPOINT
0.13	SITE-COLORECTAL
6.34	SITE-HEAD-AND-NECK
0.24	STAGE-2
0.15	STAGE-3
-0.13	AGE
-0.7	GRANULOCYTES
0	LDH-SERUM
1.54	LYMPHOCYTES
0.45	ALBUMIN
10.71	
G2: SITE-COLORECTAL == 1	
-0.01	BOX1-SCORE
-1.06	BOX2-SCORE
0.34	BMI
-0.05	NO-PROBLEM
3.49	FEEL-FULL
0.06	DRY-MOUTH
0.18	SITE-COLORECTAL

Continued on next page

Table B.7 – continued from previous page

β	X
0.16	SITE-HEAD-AND-NECK
23.03	STAGE-2
22.32	STAGE-3
-0.12	AGE
-0.84	GRANULOCYTES
0	LDH-SERUM
0.01	LYMPHOCYTES
-5.62	CALCIUM-SERUM
0.05	HGB
0.43	ALBUMIN
15.8	

Linear Regression for Censored Targets	
3.91	GENDER=FEMALE
-2.34	PERFORMANCE-STATUS-2
-4.01	PERFORMANCE-STATUS-3
0.15	BMI
-2.99	NO-PROBLEM
2.5	PAIN
2.94	DRY-MOUTH
3.21	DENTAL-PROBLEM
-10.48	SITE-BRUNCHUS-LUNG
2	SITE-COLORECTAL
4.79	SITE-HEAD-AND-NECK
-9.27	SITE-PANCREAS
21.98	STAGE-1
19.34	STAGE-2
13.15	STAGE-3
-0.18	AGE +
-1.59	GRANULOCYTES
0	LDH-SERUM
0.46	WBC-COUNT
0.07	HGB
0.65	ALBUMIN
0.03	

Linear Regression for Censored Targets	
0.11	(standardized) GENDER
-0.09	(standardized) BOX1-SCORE
0.01	(standardized) BOX2-SCORE
-0.03	(standardized) PERFORMANCE-STATUS-1
-0.03	(standardized) PERFORMANCE-STATUS-2
-0.06	(standardized) PERFORMANCE-STATUS-3
-0.02	(standardized) PERFORMANCE-STATUS-4
0.08	(standardized) BMI
-0.06	(standardized) NO-PROBLEM
0.03	(standardized) NO-APPETITE
-0.01	(standardized) NAUSEA
-0.03	(standardized) CONSTIPATION
0	(standardized) SORE-MOUTH
0.01	(standardized) TASTE-FUNNY
0.02	(standardized) SMELL

Continued on next page

Table B.7 – continued from previous page

β	X
-0.02	(standardized) SWALLOW
-0.01	(standardized) FEEL-FULL
0.07	(standardized) PAIN
0	(standardized) OTHER
-0.01	(standardized) VOMIT
-0.01	(standardized) DIARRHEA
0.04	(standardized) DRY-MOUTH
0.03	(standardized) DENTAL-PROBLEM
0	(standardized) AGE65
0.08	(standardized) WEIGHT-CHANGEPOINT
-0.17	(standardized) SITE-BRUNCHUS-LUNG
0.12	(standardized) SITE-COLORECTAL
0.13	(standardized) SITE-HEAD-AND-NECK
0	(standardized) SITE-ESOPHAGUS
-0.07	(standardized) SITE-PANCREAS
-0.02	(standardized) SITE-STOMACH
0	(standardized) SITE-OTHER-DIGESTIVE
0.03	(standardized) MISC
0.16	(standardized) STAGE-1
0.35	(standardized) STAGE-2
0.21	(standardized) STAGE-3
-0.08	(standardized) AGE
-0.13	(standardized) GRANULOCYTES
-0.09	(standardized) LDH-SERUM
-0.01	(standardized) LYMPHOCYTES
-0.02	(standardized) PLATELET
0.03	(standardized) WBC-COUNT
0	(standardized) CALCIUM-SERUM
0.04	(standardized) HGB
-0.01	(standardized) CREATININE-SERUM
0.12	(standardized) ALBUMIN
0.41	
Gating Regression (*linear regression* was selected	
2.34	GENDER=FEMALE
0.65	BOX2-SCORE
-1.73	PERFORMANCE-STATUS-2
-2.87	PERFORMANCE-STATUS-3
-4.99	PERFORMANCE-STATUS-4
0.19	BMI
-1.46	NO-PROBLEM
1.5	NO-APPETITE
1.64	PAIN
2.19	DRY-MOUTH
-0.39	WEIGHT-CHANGEPOINT
-8.68	SITE-BRUNCHUS-LUNG
2.52	SITE-COLORECTAL
2.56	SITE-HEAD-AND-NECK
-4.73	SITE-PANCREAS
16.71	STAGE-1
17.89	STAGE-2
12.77	STAGE-3

Continued on next page

Table B.7 – continued from previous page

β	X
-0.13	AGE
-0.8	GRANULOCYTES
0	LDH-SERUM
0.15	WBC-COUNT
0.04	HGB
0.36	ALBUMIN
8.59	

Table B.6: Visualizations of predicted survival time versus actual survival time of the final model

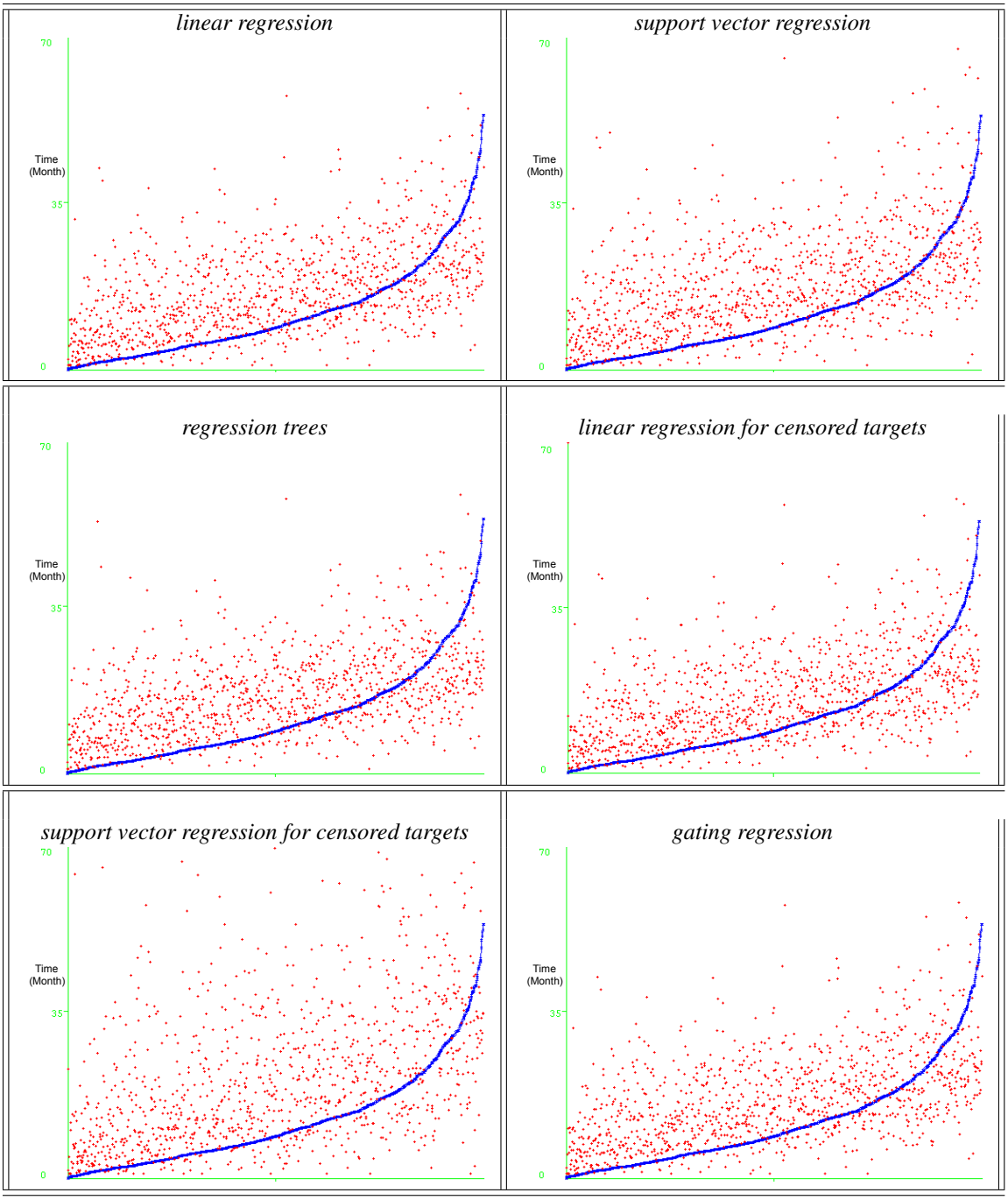


Table B.8: Confusion matrices of classification

Methods	Confusion Matrix			
Baseline				
Method: Median survival time			Actual	
			$<$ median	$>$ median
	Predicted	$<$ median	185	255
		$>$ median	762	1033
Method: Average survival time			Actual	
			$<$ median	$>$ median
	Predicted	$<$ median	0	0
		$>$ median	947	1258
Conventional Classification Methods				
Method: Naive Bayesian Network			Actual	
			$<$ median	$>$ median
	Predicted	$<$ median	522	185
		$>$ median	438	1038
Method: Bayesian Network			Actual	
			$<$ median	$>$ median
	Predicted	$<$ median	598	257
		$>$ median	362	966
Method: Neural Network			Actual	
			$<$ median	$>$ median
	Predicted	$<$ median	589	339
		$>$ median	371	884
Method: Decision Trees			Actual	
			$<$ median	$>$ median
	Predicted	$<$ median	587	373
		$>$ median	322	901
Method: Logistic Regression			Actual	
			$<$ median	$>$ median
	Predicted	$<$ median	632	237
		$>$ median	328	986
Our Final Models				
Method: NG-LOG-MCD-AVE100-LIN			Actual	
			$<$ median	$>$ median
	Predicted	$<$ median	791	423
		$>$ median	180	757
Method: NG-LOG-MCD-AVE100-SVR			Actual	
			$<$ median	$>$ median
	Predicted	$<$ median	745	336
		$>$ median	226	844

Continued on next page

Table B.8 – continued from previous page

Methods	Confusion Matrix			
Method: NG-LOG-MCD-AVE100-RT			Actual	
			$<$ median	$>$ median
	Predicted	$<$ median	785	469
		$>$ median	186	711
Method: NG-LOG-MCD-AVE100-LINc			Actual	
			$<$ median	$>$ median
	Predicted	$<$ median	808	450
		$>$ median	163	730
Method: NG-LOG-MCD-AVE100-SVRc			Actual	
			$<$ median	$>$ median
	Predicted	$<$ median	622	216
		$>$ median	349	964
Method: NG-LOG-MCD-AVE100-GAT			Actual	
			$<$ median	$>$ median
	Predicted	$<$ median	744	334
		$>$ median	227	846