Estimating patient's life expectancy after a successful kidney transplant using machine learning methods

Odhad délky života pacienta po úspěšné transplantaci ledviny pomocí metod strojového učení

Bachelor's Degree Project

Author:	**Kyrylo Stadniuk**
Supervisor:	**Ing. Tomáš Kouřim**
Consultant:	**Ing. Pavel Strachota, Ph.D.**
Language advisor:	**PaedDr. Eliška Rafajová**
Academic year:	2022/2023

Katedra: matematiky

Akademický rok: 2022/2023

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student:	Kyrylo Stadniuk
Studijní program:	Aplikovaná informatika
Název práce (česky):	Odhad délky života pacienta po úspěšné transplantaci ledviny pomocí metod strojového učení
Název práce (anglicky):	Estimating patient's life expectancy after a successful kidney transplant using machine learning methods

Pokyny pro vypracování:

1) Prozkoumejte současný přístup k transplantacím ledvin, jeho problémy a výzvy. /
   Investigate the current approach to kidney transplantation, its problems and challenges.

2) Prozkoumejte příslušné metody strojového učení a metody pro hodnocení přesnosti modelu. /
   Explore applicable machine learning methods and model accuracy evaluation methods.

3) Vyčistěte, předzpracujte a rozšiřte stávající datovou sadu. /
   Clean, preprocess and extend the existing dataset.

4) Vytvořte prediktivní model strojového učení pro odhad délky života pacienta a ohodnoťte jeho přesnost. /
   Create a predictive machine learning model estimating a patient's life expectancy and evaluate its accuracy.

5) Navrhněte úpravy skórovacího algoritmu pro transplantace ledvin na základě výsledků prediktivního modelu. /
   Design an updated kidney matching compatibility scoring algorithm based on the prediction model.

6) Prozkoumejte možnost integrace dosažených výsledků do nástroje pro správu transplantací TX Matching. /
   Evaluate the possibility of integrating achieved results into kidney transplantation management tool TX Matching.

Doporučená literatura:

1) P. Bruce, A. Bruce, P. Gedeck, Practical Statistics for Data Scientists, O´Reilly, 2020.

2) I. H. Witten, E. Frank, M. A. Hall, Ch. J. Pal, Data Mining : Practical Machine Learning Tools and Techniques. Morgan Kaufman, 2017.

3) A. Géron, Hands-On Machine Learning with Scikit-Learn, Keras, and Tensorflow: Concepts, Tools, and Techniques to Build Intelligent Systems. O'Reilly Media, 2019.

4) J. J. Kim, S. V. Fuggle, S. D. Marks, Does HLA matching matter in the modern era of renal transplantation? Pediatr Nephrol 36, 2021, 31–40.

5) R. Reindl-Schwaighofer, A. Heinzel, A. Kainz, et al., Contribution of non-HLA incompatibility between donor and recipient to kidney allograft survival: genome-wide analysis in a prospective cohort. The Lancet 393, 10174, 2019, 910-917.

6) M. Wohlfahrtová, O. Viklický, R. Lischke a kolektiv, Transplantace orgánů v klinické praxi. Grada, 2021.

Jméno a pracoviště vedoucího bakalářské práce:

Ing. Tomáš Kouřim
Mild Blue, s.r.o., Plzeňská 27, Praha 5
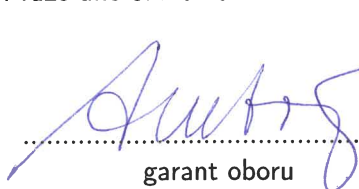
Jméno a pracoviště konzultanta:

Ing. Pavel Strachota, Ph.D.
Katedra matematiky, Fakulta jaderná a fyzikálně inženýrská, České vysoké učení technické v Praze, Trojanova 13, 120 00 Praha 2

Datum zadání bakalářské práce: 31.10.2022
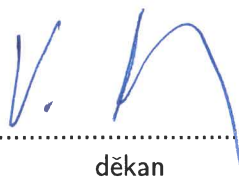
Datum odevzdání bakalářské práce: 2.8.2023

Doba platnosti zadání je dva roky od data zadání.

V Praze dne 31.10.2022

....
garant oboru

....
vedoucí katedry

....
děkan

Author's declaration:

I declare that this Bachelor's Degree Project is entirely my own work and I have listed all the used sources in the bibliography.

Prague, August 2, 2023   Kyrylo Stadniuk

Název práce:

Odhad délky života pacienta po úspěšné transplantaci ledviny pomocí metod strojového učení

Autor: Kyrylo Stadniuk

Obor: Aplikovaná Informatika

Druh práce: Bakalářská práce

Vedoucí práce: Ing. Tomás Kourim Mild Blue, s.r.o., Plzenská 27, Praha 5

Konzultant: Ing. Pavel Strachota, Ph.D. Katedra matematiky, Fakulta jaderna a fyzikálne inzenyrska, Ceské vysoké udeni technické v Praze, Trojanova 13, 120 00 Praha 2

Abstrakt: Abstrakt max. na 10 řádků.

Klíčová slova: klíčová slova (nebo výrazy) seřazená podle abecedy a oddělená čárkou

Title:

Estimating patient's life expectancy after a successful kidney transplant using machine learning methods

Author: Kyrylo Stadniuk

Abstract: Max. 10 lines of English abstract text.

Key words: keywords in alphabetical order separated by commas

Contents

Chapter 1

Introduction

Kidney transplantation (KT) is the gold standard treatment for kidney failure that provides an increased quality of life and a longer life expectancy compared to dialysis. Prior to transplantation, donor-recipient compatibility is maximized in hopes of better outcomes. However, there is a scarcity of tools capable of accurately forecasting long-term post-transplant outcomes. This thesis attempts to bridge this gap. In this thesis, we will train several machine learning models to predict the post-transplant outcomes in terms of survival function. Leveraging trained machine learning models, we aim to develop an application to aid clinicians in decision-making.

In Chapter 2, we dive into the medical aspects of transplantation to better understand the domain. Specifically, we will review the characteristics of the kidney, common causes of kidney failure, clinical aspects of kidney transplantation, history of kidney transplantation, and immunology of kidney transplantation. We aim to provide a comprehensive description of the relevant medical concepts and principles related to kidney transplantation.

In Chapter 3, we explore the fields of machine learning and survival analysis. We review the main branches of ML, including but not limited to supervised and unsupervised learning, and touch upon the most prevalent algorithms used in each. Additionally, we discuss approaches to data preparation, model training, and hyperparameter tuning in machine learning. Lastly, we provide a comprehensive overview of survival analysis, classic survival analysis methods, and machine-learning-based survival analysis techniques.

Chapter 4 is devoted to data. In this chapter, we intend to perform a comprehensive data analysis, including examining value distributions and the influence of various features on survival. In addition to that, we describe the data acquisition process, data loading, and preprocessing pipeline, along with the feature engineering process.

In Chapter 5, we present the results of our study. It includes the models we trained for the deceased and living donor subgroups and their thorough evaluation. As part of this, we examine their performance metrics and analyze how they evolve over time. Moreover, we analyze the feature importance to gain a better understanding of the factors that drive the models' predictions. Furthermore, we compare the models based on their training experience and practicality.

In Chapter 6, we present KidneyLife, an inference application designed to facilitate the user-friendly use of machine learning models for survival analysis. The application aims to aid physicians in the decision-making process by providing a comprehensive estimate of a kidney transplant recipient's life expectancy. We will explain its architecture and design considerations, and discuss existing software for KT.

Chapter 2

Medical Background

2.1 Kidney — An Overview

The kidneys are reddish-brown, bean-shaped organs located on both sides of the spine beneath the lower ribs. They serve several critical functions, such as filtering blood, regulating blood pressure, aiding red blood cell production, and maintaining the fluid balance. Structurally, a kidney consists of about 1 million *nephrons*, each consisting of a small filter called a *glomerulus*, attached to a tubule, responsible for filtering waste products from the blood and excreting them with small amounts of fluid as urine [33].

Each kidney performs 50% of the normal kidney function. However, if one kidney is lost, the remaining kidney can adapt, increasing its capacity to as much as 75% of normal function. Kidney function is measured with *glomerular filtration rate (GFR)* [33]. There are two types of GFR: *measured GFR (mGFR)* and *estimated GFR (eGFR)*. Measuring mGFR is a problematic and lengthy process, so healthcare practitioners usually use a formula to calculate eGFR. An eGFR of 90 and above is the normal range. An eGFR of 15-89 is a range for different stages of chronic kidney disease. Values below 15 might indicate renal failure [38].

Impairment in kidney function can have serious health consequences. There are many reasons why kidneys may become dysfunctional, such as end-stage chronic kidney disease, heavy metal poisoning, polycystic kidney disease, an infection, nephrotoxic drugs, lupus, physical injury, etc [36, 37]. Among these conditions, chronic kidney disease is the most prevalent — about 10% of the population worldwide is affected by some form of chronic kidney disease [35]. *Chronic kidney disease (CKD)* is an incurable condition characterized by a gradual reduction in kidney function over time [34, 35]. Diabetes and high blood pressure are responsible for the majority of cases of chronic kidney disease, with high blood sugar damaging the glomeruli and high blood pressure damaging the kidney blood vessels [34, 37]. The latter interaction is associated with a negative feedback loop, where it is unclear whether kidney damage causes high blood pressure or vice versa [35].

CKD progresses in 5 stages, ultimately culminating in *end-stage renal disease (ESRD)*, where kidneys fail. Treatment options include *dialysis*, a mechanical process that substitutes kidney function but cannot fully replicate it. It leads to a reduced life expectancy of, on average, 5 to 10 years. Alternatively, *kidney transplantation* offers a more permanent solution but requires a compatible donor, waiting for which can take many months or years, and the use of immunosuppressants after the transplantation [36].

In conclusion, the kidneys play an essential role in maintaining overall health with their ability to filter waste, regulate blood pressure, and balance fluids. Kidney failure has serious health implications and there are multiple reasons for it. Chronic kidney disease, resulting from various causes such as diabetes and hypertension, progressively impairs kidney function, leading to end-stage renal disease. While dialysis offers temporary relief, it cannot fully substitute for a healthy kidney, which brings us to

the crucial role of kidney transplantation. Kidney transplantation offers a more conclusive and sustainable solution for patients with ESRD, albeit accompanied by challenges related to donor compatibility and life-long immunosuppression. Both of which will be discussed in the subsequent section.

2.2   Clinical Aspects of Kidney Transplantation

Kidney transplantation stands as the most commonly performed organ transplant worldwide. This prevalence is partly due to the high incidence of diseases such as diabetes and chronic kidney disease, which often lead to chronic renal failure. Unlike organs such as the liver and heart, kidney transplantation is relatively straightforward, contributing to its frequency of transplantation. Kidneys can be obtained from both deceased and living donors - relatives or altruistic volunteers. Living donors can donate one kidney and continue to lead normal and healthy lives, significantly expanding the potential donor pool [32].

ABO blood group and histocompatibility (defined in 2.5) matchings for kidney transplantation are of utmost importance. Proper matching substantially reduces the risk of rejection and improves transplant outcomes. Compared to organs such as the liver or bone marrow, kidneys do not present additional challenges such as graft versus host disease (GVHD), simplifying the transplantation process [32].

However, kidney transplantation, despite its advances, faces significant challenges. Two critical issues confront potential recipients: the scarcity of available organs and the risk of sensitization following a failed first transplant, decreasing the pool of available donors even further. In such cases, the immune system develops antibodies to the graft alloantigens, and any further transplantation of organs with similar antigens will result in hyperacute rejection (many of these termes are explained in 2.5). This fact often leaves many patients unable to find a compatible donor after one or two rejection episodes [32].

Recipients of kidney transplants typically require life-long immunosuppression. While immunosuppressants are essential to prevent organ rejection, they are associated with severe side effects, such as increased risk of cancer, hypertension, and metabolic bone disease [32].

Given the high incidence of renal failure in diabetic patients - affecting roughly 30% of individuals with advanced diabetes - simultaneous kidney and pancreas transplants are sometimes performed. This approach addresses renal failure and underlying diabetes, offering a comprehensive solution for these patients [32].

In summary, kidney transplantation has become a vital and frequent medical procedure, offering improved quality of life to many suffering from kidney failure. The procedure, while simpler than other organ transplants, has its own set of challenges. The following section will delve into the history of kidney transplantation, showing its path from the initial experiments to the most frequently performed organ transplantation today. The historical perspective will provide insights into the advancements in surgical techniques, immunosuppressive therapy, and donor-recipient matching that have shaped the current state of kidney transplantation.

2.3   The History of Kidney Transplantation

Early Animal Experiments

Advancements in surgical methods at the beginning of the 20th century eventually led to experiments with organ transplantation. One of the first recorded transplantations was an *autograft*, where the donor and recipient are the same individual, performed by Emerich Ullmann on March 1, 1902, at the Vienna Medical School. Ullmann successfully connected the dog's kidney to the vessels of its neck, which

resulted in the urine production. The success of this experiment was notable enough to be presented to the Vienna Medical Society, sparking considerable interest.

That year, other experiments followed. Alfred von Decastello performed a dog-to-dog kidney *allograft*, a transplant between two individuals of the same species, at the Institute of Experimental Pathology in Vienna. Although initially, the transplanted kidney produced urine, it eventually ceased. Later, Ullman performed a dog-to-goat kidney *xenograft*, a transplantation between individuals of different species, which resulted in brief function time as well.

At the same time, in Lyon, Alexis Carrel and his colleagues were working on vascular suturing methods. Carrel's technique, known as Carrel's seam, was a considerable improvement over existing methods, addressing the common issues of thrombosis, hemorrhage, stricture, and embolism [41]. His consecutive move to The Rockefeller Institute for Medical Research in the United States led to further refinements of his method and the documentation of organ rejection. For his contributions, Carrel received the Nobel Prize in Medicine in 1912 [41].

These early experiments, although varied in their outcomes, were defining in shaping the future of the organ transplantation. They not only illustrated the possibility of organ transplantation, but also highlighted the challenges, such as rejection, that would direct the course of future research. The insights gained in animal experiments laid the groundwork for the next big milestone in transplantation medicine: the advent of human organ transplantation. This transition from animals to humans marked the beginning of a new era in medical history.

Early Human Transplantation

Two first recorded human renal *xenografts* are credited to Mathieu Jaboulay in 1906. It involved a pig and a goat as donor animals. One kidney was transplanted to the arm and the second to the thigh. Although each kidney functioned only for an hour, these efforts marked the beginning of human transplantation attempts [?, 4]. Ernst Unger's xenografts in 1909 gained more attention. His first attempt, involving the transplantation of a kidney from a stillborn baby to a baboon, resulted in a lack of kidney function despite a successful connection of vessels. Inspired by a successful surgery, Unger attempted a monkey-to-human xenograft, which also resulted in failure.

These experiments demonstrated the technical feasibility of kidney transplantation but also exposed the challenge of graft rejection. Alexis Carrel, in a 1914 lecture, mentioned J. B. Murphy's work on irradiation and benzol treatment, suggesting their potential to improve graft survival [41]. Inspired by these findings, Carrel conducted his own experiments with irradiation, achieving prolonged graft survival. However, these findings were never formally published [41].

The period of the 1930s and 1940s was stagnant compared to the beginning of the century. European surgical centers that studied transplantology before were in decline. Meanwhile, the Mayo Clinic in the US was conducting some cautious experiments without considering Carrol's works and attempts at immunosuppression [41].

However, it was during that period that a significant milestone was achieved by Yurii Voronyi. On March 3, 1933, in Kherson, Ukraine, Voronyi performed the first human kidney allograft on a woman suffering from acute renal failure due to mercury chloride poisoning. Given the ethical concerns surrounding living donors and previous failures of xenografts, Voronyi considered a cadaveric transplant the only viable option. Although there was initial urine production, the transplant failed 48 hours postsurgery due to blood group incompatibility and prolonged warm ischemia, triggering an immune reaction. Despite this setback, Voronyi continued to perform similar transplantations. He viewed these transplants as temporary measures to bridge the gap until the recipient's own kidneys could recover. Out of the six transplants he performed, two patients experienced a complete recovery, regaining normal kidney function [41].

The pioneering work of Jaboulay, Unger, and Voronyi not only demonstrated the technical feasibility of human organ transplantation but also highlighted its main challenge of graft rejection. Despite the progress, the field has yet to witness success, largely due to the lack of effective immunosuppression. In the next section, we will explore important moments of 1950s that transformed human renal transplantation from a daring experiment to feasible medical procedure.

First Successes

In 1946, at the Peter Bent Brigham Hospital in Boston, a group of surgeons: Hufnagel, Hume, and Landsteiner, performed kidney transplantation under local anesthetic on the arm vessels. The short period of kidney functioning may have helped the patient recover from acute renal failure, igniting the hospital's interest in renal transplantation.

Meanwhile, European surgeons were making significant advancements. Notably, Simonsen in Denmark, Dempster in London, and Küss in Paris concluded that it is preferable to place the kidney in the pelvis. Furthermore, both Simonsen and Dempster deduced that the immune response was responsible for graft failure and hypothesized that the humoral mechanism of rejection was probable.

The early 1950s marked a period of active experimentation. In Paris, Jean Hamburger reported the first live-related kidney transplant between a mother and her child, achieving the immediate function of the transplanted kidney for 22 days until it was rejected. Meanwhile, in Boston, a series of nine transplantations with the thigh position of the allograft was closely studied. Moreover, in 1953, David Hume introduced the pre-transplant use of hemodialysis. Although some success was achieved with the administration of the adrenocorticotropic hormone (more known as cortisone), it was deemed clinically insignificant. Hume's further research suggested the potential benefits of prior blood transfusions, blood group compatibility, and removal of host's both kidneys for transplant success - insights later validated with further research. On December 23, 1954, in Boston, Joseph Murray performed kidney allograft from one identical twin to another, bypassing the rejection barrier. From that time, many similar surgeries were performed in Boston [?, 3].

To conclude, the early successes during the 1950s marked a transformative period in the history of kidney transplantation. Works of Hume, Murray, and others underlined the critical role of immune response in graft survival, starting the quest for effective immunosuppression, to which the subsequent section is dedicated.

Attempts in Immunosuppression

The exploration of immunosuppression in transplantation began as early as the late 1940s. At the Mayo Clinic in 1948, patients with rheumatoid arthritis were administered cortisone, an adrenal cortical hormone with mild immunosuppressive properties, which provided temporary relief. Although initially praised, its effects were deemed clinically insignificant for transplantation purposes. It led researchers to revisit earlier experiments with irradiation. Experiments on mice by Joan Main and Richmond Prehn showed promising results, inspiring human trials in Boston and Paris [42].

In 1958, Murray's team in Boston employed radiation in human transplantation, achieving a significant breakthrough with a kidney transplant between non-identical twins that lasted 20 years. Similarly, in Paris, Jean Hamburger's team accomplished a 26-year functioning transplant using radiation [42].

The quest for safer immunosuppressive methods led to the anticancer drug 6-mercaptopurine (6-MP). In 1959, Schwarz and Damesehek published a paper that described how 6-MP lowered immune response to foreign proteins in rabbits. Inspired by their work, Roy Calne performed his own dog experiments, showing promising results backed by Charles Zukoski and David Hume. Despite initial setbacks, Küss and others reported prolonged graft survival from non-related donors using total body irradiation (TBI)

complemented by 6-MP. The introduction of azathioprine in 1959, a derivative of 6-MP, by Gertrude Elion and George Hitchings, further improved results. Their groundbreaking work earned them the Nobel Prize, and by 1961, azathioprine was available for human use [42].

In 1963, a conference held by the National Research Council revealed a bleak outlook for kidney transplantation, with less than 10% survival beyond three months. It changed when Tom Starzl presented a protocol combining 6-MP with prednisone, leading to over one-year graft survival in 70% of cases. His results revolutionized the field, resulting in 50 new US transplantation programs and setting a 20-year standard in post-transplant immunosuppression [42].

In summary, the late 1940s to the early 1960s was a period of discoveries and experimentation. The use of cortisone in 1948, the administration of 6-MP for immunosuppression, and the invention of azathioprine in 1959 led to better immunosuppression, culminating in Tom Starzl's protocol, which became the standard for the next two decades. The next section is dedicated to a phase of steady developments, in literature referred to as plateau, that would solidify the practice of kidney transplantation.

Plateau

From 1964 to 1980, kidney transplantation saw gradual progress. Dialysis, developed during WWII, finally became available for chronic renal failure as a result of the invention of Teflon arteriovenous conduits in the 1960s. Acceptance of brain death expanded donor pools. Organ preservation techniques also advanced: total body hypothermia was replaced by targeted cold solutions infusions that preserved organs better. By the mid-60s, longer preservation times allowed organ exchanges between centers. Concerns about equitable distribution of organs led to the National Transplant Act in 1984, establishing the United Network of Organ Sharing (UNOS) for oversight [42].

As techniques were improving and donor pools expanding in the period from 1964 to 1980, so was the understanding of the nuances of immune compatibility. The following section describes the developments in tissue typing that would further expand the field of kidney transplantation.

Tissue Typing

The concept of tissue typing, suggested by Alexis Carrel in the early 20th century, remained unproven until Jean Dausset discovered the first human leukocyte antigen (HLA) in 1958 (more on HLA in 2.5) . It wasn't until 1964, with Paul Terasaki's development of the microcytotoxicity assay, that testing for antibodies became reliable. The test involved mixing the donor's lymphocytes with the recipient's serum and swiftly became the standard, known as the *crossmatch test* [42].

For several years, Terasaki performed typing for most U.S. transplant centers and found a couple of observations: 1) Positive crossmatch test predicts hyperacute rejection. The test is performed by mixing recipient's serum with donor's blood cells, if there is a reaction, the test is considered to be positive and the transplantation cannot be performed. 2) matching can reliably identify optimal donors within a family, and it was assumed that the same principle would apply to non-related recipients.

However, in 1970, Terasaki's review of his extensive database of cadaver renal allografts revealed no correlation with the typing. It raised much agitation in the tissue typing community, and his grant was temporarily suspended until others didn't report the same. Since then, many crucial histocompatibility antigens have been discovered (Class II locus: D and DR, more on antigens in Section 2.5). Histocompatibility matching remains essential in bone marrow transplantation and in selecting family donors [42].

While tissue typing and matching have played a crucial role in improving transplantation outcomes, the evolution of immunosuppressive drugs has been equally, if not more, important. The subsequent section tells more about new developments in immunosuppressive drugs in upcoming years.

Advancements in Immunosuppression

The discovery of cyclosporine in 1976 by Jean-François Borel marked a significant milestone in the realm of transplantation. As a fungal derivative with potent immunosuppressive properties, cyclosporine drastically improved the outcomes of both renal and extra-renal transplants, surpassing the efficacy of the previously used drug, azathioprine. Similar to 6-MP, to achieve the best results, it had to be combined with prednisone. This protocol remained standard until 1989, when Tacrolimus, an even more powerful immunosuppressive agent, was introduced. Tacrolimus proved to be effective in cases where the combination of cyclosporine and prednisone was insufficient, effectively replacing cyclosporine as a usual baseline agent [42].

In conclusion, the history of renal transplantation has been marked by groundbreaking discoveries and persistent challenges. From the pioneering attempts of xenografts in the early 20th century to the technical advancements and immunological breakthroughs of the mid-century — each milestone has been essential in shaping the current landscape of organ transplantation As we explored the history of kidney transplantation, a deeper understanding of the immune system has become crucial. This sets the stage for the next section, which examines the fundamentals of immunology, laying the foundation for understanding the interplay between the immune system and the transplanted organ. .

2.4   The Introduction to Immunology

The *immune system* is a sophisticated defense mechanism that evolved to protect multicellular organisms from pathogens such as bacteria, fungi, viruses, and parasites. It is a network of many cells and tissues that compose a complex system that detects, evaluates, and responds to the invader. It is essential to understand these mechanisms, as the immune response plays a crucial role in graft acceptance or rejection [6].

Innate and adaptive immunity are the two interconnected systems of immune response. *Innate immunity (*also *natural* or *native)* includes primitive built-in cellular and molecular mechanisms that provide rapid, albeit non-specific responses to common pathogens. In contrast, *adaptive (specific* or *acquired) immunity* is slower to respond but capable of providing more targeted responses [6].

Physical barriers, including epithelia and mucous membranes, constitute the host's first line of defense and are a part of innate immunity. This branch of the immune system not only prevents invaders from infiltrating the host but also quickly destroys many microbes that succeed in breaching these barriers. Innate immunity provides the necessary protection until adaptive immunity is activated. It also communicates to the adaptive immunity how to best respond to the invader. Furthermore, innate immunity plays an important role in the clearance of dead tissue and the initiation of repair after the tissue is damaged [6].

Adaptive immunity is broadly categorized into *humoral* and *cell-mediated immunity*. Central to both humoral and cell-mediated immune responses are *lymphocytes*, also known as *white blood cells*. *B lymphocytes (B cells)* mediate humoral response by producing antigen-specific antibodies on encounter with the antigen. Produced antibodies then bind themselves to *antigens* — foreign molecular structures identified by common molecular patterns known as *pathogen-associated molecular patterns* (*PAMPs*) — to mark them for destruction. The immune system uses *pathogen recognition receptors* (*PRRs*), found on the surface of T cells, in conjunction with antibodies to detect and categorize these PAMPs, which can take the form of molecules on the surface of a pathogen or its by-products. PRRs bind to PAMPs and initiate a targeted cascade of events that culminate in the pathogen's elimination [6].

T lymphocytes, on the other hand, when encountering an antigen, start to proliferate, forming an army of T cells that will eliminate the invader and will form long-term memory about the pathogen.

Activated T cells are divided into the following categories: helper T cells ($CD4^+$), that help B cells to produce antibodies and help kill ingested microbes; *cytotoxic T cells* ($CD8^+$) that target and kill infected cells; *regulatory T lymphocytes* that prevent or limit immune responses; and *memory cells*, that remain in the body long-term to provide faster and stronger immune response if the same antigen is encountered in the future [6].

Pathogen-host interaction is a continuous arms race, as pathogens usually have a short life cycle and can modify their DNA to elude the host's recognition systems. The immune system counters this with the generation of host-tolerant lymphocytes with diverse PRRs during the development in bone marrow. Cells that react to the host's own cells are eliminated, ensuring that only non-self-reactive cells are allowed in circulation. The principle of recognizing self vs. non-self is called tolerance [6].

In conclusion, the immune system is a complex network of molecules, cells, tissues, and organs that cooperate in protecting the organism from pathogens. The system can be divided into two main branches: innate and adaptive, which cooperate in protecting the host from infections while developing long-term immunity to specific pathogens. Understanding the mechanisms of the immune system is essential to understanding the domain of kidney transplantation.

2.5   Immunology of Kidney Transplant

The degree to which the immune system responds to a graft depends on the type of graft. There are four types of grafts:

- **Autograft** is body tissue transfer from one body site to the other in the same individual.

- **Isograft** is a tissue transplanted between two genetically identical individuals. In humans, it is usually homozygotic (identical) twins.

- **Allograft** is a tissue transplanted between two genetically non-identical individuals of the same species.

- **Xenograft** is a tissue transplanted between individuals of different species.

Autografts and isografts are generally accepted because they are genetically identical. Allografts, being genetically different, are typically identified as foreign by the immune system and as a result rejected. Xenografts, which have the most significant genetic differences, are the most vigorously rejected by the immune system [32].

2.5.1   Immune response to the graft

There are three reasons why the immune system may react to the allograft: damage due to ischemia, reaction to the incompatible blood group antigens, and reaction to major histocompatibility complex (MHC) and minor histocompatibility (miH) antigens.

Ischemia Reperfusion Injury   The transplantation process inevitably includes termination of blood flow and, as a result, oxygenation. This interruption in blood supply inhibits the cell's ability to generate sufficient energy to maintain homeostasis, leading to cellular damage or death. If this happens, the kidney is said to experience *ischemia-reperfusion injury (IRI)*. IRI is a significant factor in the success of kidney transplantation. When the blood flow to the ischemic kidney is restored, dead cells trigger an innate immune response. It is associated with the release of *danger-associated molecular patterns (DAMP)* from the compromised cells, serving as a critical alert mechanism. These DAMPs are recognized by

Table 2.1: MHC class division

MHC class I	MHC class II
HLA-A	HLA-DR
HLA-B	HLA-DP
HLA-C	HLA-DQ

both innate and adaptive immunity, although the former is predominately activated. Such an immune response might damage the graft even more and contribute to acute rejection [41]. Because of that, the duration of ischemia is crucial in predicting the graft (and patient) survival.

Blood Group Compatibility   ABO blood group antigens, expressed by most cell types, play a crucial role in the compatibility of organ transplants. If the transplantation between incompatible pairs were performed, it would result in a severe and often immediate reaction known as hyperacute antibody-mediated rejection (AMR or ABMR). The rejection risk is so high that the ABO blood group compatibility is the first thing checked before performing the transplant.

There are four primary blood groups: A, B, O, and AB. Within this system, individuals with blood group O are so-called "universal donors" – organs from them can be transplanted to recipients with any ABO blood group. Whereas recipients in the AB group can safely receive organs from recipients with any ABO blood group and are called "universal recipients". In clinical practice, however, particularly with deceased donors, the preference is to match organ recipients with ABO-identical organs to prevent potential inequities in organ allocation and access. In the case of living donors, the principle of ABO-identity is somewhat more flexible, where organs from ABO-compatible donors are considered acceptable for transplantation [41].

Histocompatibility   Genetically similar tissues, known as *histocompatible*, are less likely to trigger an immune response post-transplant, as the immune system does not recognize the transplanted tissue as foreign. Conversely, *histoincompatible* tissues, characterized by genetic differences, typically trigger an immune response. Although more than 40 distinct loci encode the antigens responsible for histocompatibility, the most vigorous allograft-rejection responses are attributed to loci within the *major histocompatibility complex (MHC)*. The organization of MHC in humans is called *human leukocyte antigen (HLA)* [32].

Histocompatibility antigens, genetically encoded antigens that cover cell surfaces, play a crucial role in recognizing self versus non-self. In all vertebrates, histocompatibility antigens are categorized into a single *major histocompatibility complex (MHC)* and numerous *minor histocompatibility (miH)* systems. Mismatches in either MHC or miH result in an immune reaction, most severe in the case of MHC. Rejection in MHC-compatible donor-recipient pair is usually delayed, in some cases forever. However, the miH mismatch might be so drastic that it would be comparable to the MHC mismatch [41].

The MHC itself is divided into class I and class II antigens. MHC class I antigens cover the surfaces of most cells and can activate cytotoxic CD8 cells. MHC class II antigens, found on specific immune cells, play an essential role in immune response coordination [41]. In humans, each MHC class is further divided into three subgroups, as illustrated in Table 2.1.

It is important to note that class I and II mismatches differ in their impact on graft survival. A mismatch of one or two class I antigens has little effect on graft survival, whereas a single mismatch in class II antigen is equivalent to a mismatch of 3 or 4 class I antigens. When there are mismatches in both class I and II antigens, the risk and severity of the rejection are notably increased [32].

HLA Typing and Matching   In clinical practice, clinicians asses and try to match donors and recipients according to the number of HLA-A, -B, and -DR mismatches, ranging from zero mismatches (0-0-0) to a maximum of 6 mismatches (2-2-2). Generally, more emphasis is placed on DR loci due to the capability of CD4 T cell activation, which might trigger both humoral and cellular adaptive immune responses [41].

HLA typing of potential donors and recipient is conducted with a *microcytotoxicity test*. The subject's white blood cells are distributed into various wells on a microtiter plate, after which specific antibodies for various class I and class II MHC alleles are added to different wells. Following incubation, complement is introduced to the wells, and cytotoxicity is assessed by the cells' absorption of various dyes. If a cell has an MHC allele for which a particular antibody is specific, then the cell will die upon the addition of a complement, and these dead cells will take up a dye [32].

Even when a fully HLA-compatible donor is not available, the transplantation still can be successful. In this situation, a one-way mixed-lymphocyte reaction (MLR) test can be used to assess the degree of class II MHC compatibility. The irradiated (or treated with mitomycin C) potential donor's lymphocytes play the role of stimulator, and the unaltered recipient's lymphocytes take a responder role. Then, the proliferation of recipient cells is measured. The intense recipient leukocyte proliferation indicates a poor prognosis for graft survival. The MLR gives a better understanding of the degree of CD4 cell activation. However, it takes several days to run the assay, often making it less suitable in the case of cadaver transplantation compared to a quicker microcytotoxicity test, which takes only a couple of hours. Notably, even perfect HLA matching does not guarantee rejection-free transplantation. Minor histocompatibility loci mismatches can still lead to graft rejection, necessitating at least some level of immunosuppression even in HLA-identical pairings. It is particularly critical in kidney and bone marrow transplants, where HLA matching is vital, whereas heart and liver transplantation can withstand higher levels of mismatching [32].

Mechanisms of Graft Rejection   Graft rejection is mainly due to cell-mediated immune response to alloantigens, predominantly MHC antigens, present on the graft cells. The process of cell-mediated graft rejection is divided into two phases. The first is known as the sensitization phase, which involves the recognition of MHC and miH alloantigens by helper CD4 and killer CD8 cells, leading to their proliferation. The second stage, the effector stage, is where the actual destruction of the graft occurs [32].

A range of effector mechanisms are involved in graft rejection. The most prevalent are cell-mediated reactions involving delayed-type hypersensitivity and cytotoxic T lymphocyte-mediated cytotoxicity. Less common mechanisms are antibody-plus-complement lysis (cell disintegration) and destruction by antibody-dependent cell-mediated cytotoxicity (ADCC) [32].

2.5.2   Clinical Manifestations of Graft Rejection

The onset and severity of graft rejection varies depending on the organ transplanted and the underlying immune response mechanisms involved. Rejection can be categorized into three types: hyperacute, acute, and chronic rejections.

Hyperacute rejections occur within the first 24 hours post-transplant. This type of rejection occurs due to the prior sensitization to graft antigens, with specific antibodies already present in the bloodstream. It may happen due to several reasons: recipients of repeated blood transfusions sometimes develop significant amounts of antibodies to MHC antigens present on white blood cells of the transfused blood; women, through repeated pregnancies, can become sensitized to paternal alloantigens of the fetus; individuals with previous grafts may have antibodies against alloantigens of that graft; and

naturally occurring antibodies to blood group antigens are always present, although pre-transplant blood group matching has made rejections of this nature rare [32].

Acute rejections typically happen within the first few weeks post-transplant. This type of rejection is primarily mediated by adaptive immunity via T cell responses [32].

Chronic rejections, on the other hand, happen from months to years after the transplantation and are mediated by both humoral and cell-mediated mechanisms. While the advancements in immunosuppression and tissue typing significantly improve short-term survival, little progress has been made in terms of long-term survival. Chronic rejections are hard to manage with immunosuppressants and may necessitate another transplantation [32].

2.5.3 Immunosuppression

Most of the available immunosuppression methods, while essential in preventing graft rejections, have the disadvantage of being non-specific, meaning they suppress immune reactions to all antigens, not only ones of the graft, placing the recipient at the risk of infection and, in some cases, more severe complications. For instance, some drugs target the speed of proliferation of activated lymphocytes. In doing so, they slow down the proliferation of all cells in the body. It can be particularly problematic for rapidly dividing cells, such as those of gut epithelia or bone marrow hematopoietic stem cells, placing patients at risk of severe or even life-threatening complications. Moreover, long-term use of immunosuppressive agents puts patients at risk of cancer, hypertension, and metabolic bone disease. In this section, we will delve into commonly used immunosuppressive methods [32].

A class of drugs known as *mitotic inhibitors* is used to inhibit rapid cell division across all cells. These are administered just before and after the transplantation to stop T-cell proliferation. The most commonly used mitotic inhibitors include *Azathioprine*, *Cyclophosphamide*, and *Methotrexate*. However, their broad action on cell division can lead to significant adverse effects [32].

Corticosteroids, another class of drugs, are often used in conjunction with mitotic inhibitors to prevent acute rejection. They have anti-inflammatory properties and act on different levels of immune response. Commonly used corticosteroids are *prednisone* and *dexamethasone* [32].

Fungal metabolites, such as *Cyclosporine A (CsA)*, *Tacrolimus*, and *rapamycin* are also widely used due to their potent immunosuppressive properties in heart, liver, kidney, and bone marrow transplants. A substantial drawback is their nephrotoxicity. While CsA has been widely used, Tacrolimus and rapamycin, being much more potent, might be administered at much lower doses, minimizing the side effects [32].

Radiation, known for its effectiveness in destroying lymphocytes, is another lymphocyte elimination method employed just before transplantation. In one such procedure, lymph nodes, spleen, and thymus are irradiated. Later, newer, more tolerant to the graft alloantigens lymphocytes will emerge, as the bone marrow was unaffected [32].

Ideal immunosuppression would be antigen-specific, targeting only alloantigens of the graft, preserving the recipient's ability to respond to infections. Such specific immunosuppression has not been achieved in humans yet, but animal models suggest its feasibility.

2.5.4 Transplant Tolerance

Taking into account the detrimental effect of long-term immunosuppression, one of the primary objectives in transplantation is the induction of immunologic non-responsiveness (tolerance) to an allograft. There are several pathways of immune non-responsiveness generation described in the literature. However, it has not gone further than animal models yet [41]. Tolerance is usually induced by prior exposure to donor antigens in a way that causes immune tolerance rather than sensitization in the recipient [32].

Chapter 3

Machine Learning Background

Machine learning is a subfield of computer science that consists of building algorithms capable of processing large amounts of data, finding patterns, and performing actions such as predictions or generating new data. It is an intersection of many fields of science, such as statistics, theory of probability, linear algebra, calculus, and certainly, computer science.

Machine learning excels in problems that are either overly complex or have no known algorithm.[9] It can help us generate knowledge. We can extract previously unknown correlations from the data and build knowledge. It might make fewer errors in decision-making than humans.

Based on the problem and, therefore, on our approach to building a dataset and the model, machine learning can be divided into four subfields: *supervised*, *semi-supervised*, *unsupervised*, and *reinforcement learning*. *Supervised learning* means that data is labeled, and we want to predict labels for the unlabeled data. The term labeled data is explained in the following section dedicated to supervised learning. Unsupervised learning deals with unlabeled data.

Semi-supervised learning deals with partially labeled data, and we need to label it fully either manually, or using techniques such as *clustering*.

In *reinforcement learning*, we create an environment, set up rewards for performing certain actions and punishment for others, and let the machine (actor) perform actions that produce the highest reward.

Every field is affected by human errors, and medicine is no exception. Machine learning also makes mistakes, but if we manage to get at least 1% fewer errors than humans make, this will be a substantial achievement.The human body is a complex system, where it is very difficult to comprehend all processes and and how they relate to each other. In addition, machine learning can help us gain insight into them through accumulated data and discover new relations between them.

In this chapter, we will cover all theoretical backgrounds that might prove useful for solving our problem, including classical machine learning, statistical survival analysis, basic steps that are required to **create machine learning system**s, and data preprocessing. We will begin by exploring supervised learning.

3.1 Supervised Learning

Supervised learning is the process of training a model on data where the outcome is known, to make predictions for data where the outcome is not known[12]. *Classification* and *regression* are common supervised learning tasks. In this section we will define these problems and the necessary terminology, and describe commonly used algorithms that are used to solve these types of problems.

In supervised learning the *dataset* is the collection of labeled examples $\{(\overline{x}_i, y_i)\}_{i=1}^{N}$, where each individual $\overline{x}_i$ is called a *feature vector*. A feature vector is a vector that in each its dimension $j = 1, ..., D$

contains a value that describes an example in some way. This value is called a *feature* and is denoted as $x^{(j)}$. The *label* y^i might be either a finite set of classes $\{1, 2, ..., C\}$, in case of a classification task, or a real number, a vector, a matrix or graph, in case of a regression. The goal of supervised learning algorithm is to create a model using the dataset that will take the feature vector as an input and produce a label or a more complex structure as an output.

Classification is a problem of assigning a label to an unlabeled example. This problem is solved by a classification learning algorithm that takes a labeled set of examples as input and produces a model that takes an unlabeled example as input and outputs a label. If the set of labels has only two classes we talk about *binary classification*. Consequently, if the set of labels has three or more classes, it is a *multiclass classification*. Some algorithms are binary classifiers by definition while others are multiclass classifiers. It is possible to create an *ensemble* out of binary classifiers that will be able to perform multiclass classification. An ensemble is a combination of algorithms that are connected to perform one task.

Regression is a problem of predicting a *target value* given an unlabeled example. The problem is solved by a regression learning algorithm that takes a set of labeled examples as input, and produces a model that takes an unlabeled example as input and outputs a target value.

Classification and regression tasks are similar in many ways and often for each classifier there is an equivalent regressor, and vice versa. In the following subsections we are going to explore some techniques for supervised learning.

3.1.1 Linear Regression

Linear regression is a popular regression learning algorithm. The model produced is a linear combination of all features.

The problem formulation we are trying to solve is as follows: Given a collection of labeled examples $\{(\overline{x}_i, y_i)\}_{i=1}^{N}$, create a model

$$f_{\overline{w},b}(\overline{x}) = \overline{wx} + b, \tag{3.1}$$

where N is the size of the collection, $\overline{x}_i$ is a *feature vector* of D dimensions of example $i = 1, \ldots, N$, every feature $x_i^{(J)} \epsilon \mathbb{R}$, $y_i \epsilon \mathbb{R}$ is the target value. $\overline{w}$ is a D-dimensional vector of parameters and $b \epsilon \mathbb{R}$. Notation $f_{\overline{w},b}(\overline{x})$ means that f is parametrized by $\overline{w}$ and b.

To train the linear regression means to find optimal values $(\overline{w}^*, b^*)$ of parameters $\overline{w}$ and b so that the model makes as accurate predictions as possible. In graphical terms, it means finding such a hyperplane that fits data points from the training set as well as possible, as shown in image3.1.

To find optimal parameters we need to minimize the following expression:

$$\frac{1}{N} \sum_{i=1...N} (f_{\overline{w},b}(\overline{x}_i) - y_i)^2. \tag{3.2}$$

It is called *mean squared error (MSE),* the *loss function* that comprises of *squared error loss* $(f_{\overline{w},b}(\overline{x}_i) - y_i)^2$, another loss function that evaluates individual predictions. The loss function measures the model's overall performance (MSE) or evaluates each prediction (square error loss).

There is a *closed-form solution* for finding optimal values $(\overline{w}^*, b^*)$. A closed-form solution is a simple algebraic expression that gives the result directly. In case of linear regression, it is the *normal equation*, and it looks like the following:

$$\overline{\mathbf{w}}^* = (\mathbf{x^T x}) \mathbf{x^T y}. \tag{3.3}$$
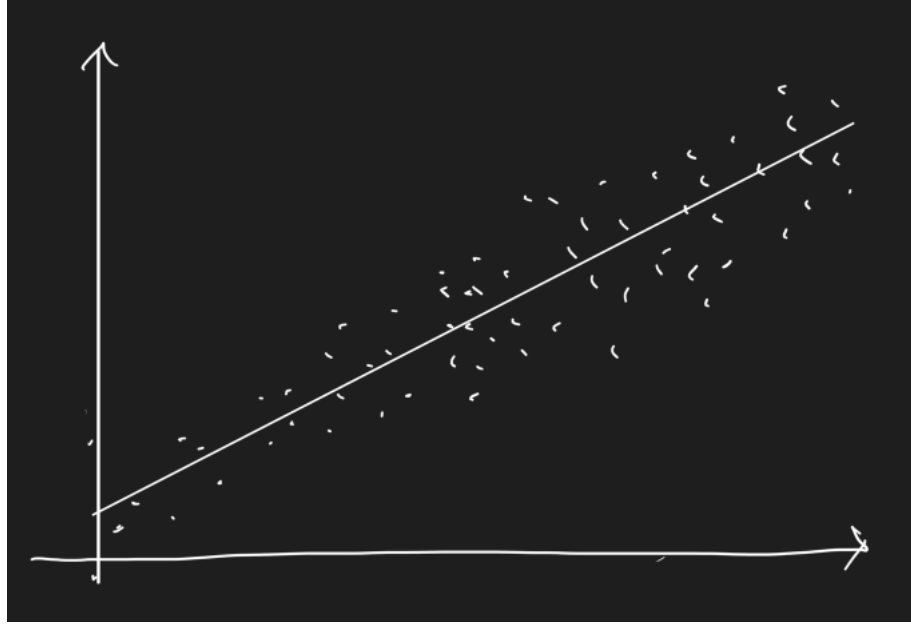
Where x^T means transposed feature matrix x.

Figure 3.1: Linear regression for two-dimensional data

We could select another loss function, but according to Andriy Burkov, it would be a different algorithm. For example, we could take the absolute difference between $f(x_i)$ and y_i but that would create problems as the derivative of absolute value is not continuous. Therefore the function is not smooth, which might create unnecessary complications during the optimization process.

Linear models are usually resilient to overfitting because they are simple. The model overfits when it learns the intricacies of the training dataset so well that it remembers actual values instead of learning the underlying pattern. Such model is unable to make accurate predictions when confronted with unseen data. More on overfitting in section 3.6.

3.1.2 Logistic Regression

Logistic regression is a binary classifier that estimates the probability of an example belonging to a particular class. If the predicted probability of the instance belonging to a class is greater than 50%, then the model concludes that it belongs to the class (referred to as positive class and labeled as 1). Otherwise, it predicts that the example does not belong to that class (but belongs to the negative class, labeled 0). Logistic regression comes from statistics where its mathematical formulation is similar to a regression, hence the name. Multiclass classification is available in softmax regression, a multiclass variant of logistic regression.

As with linear regression, in logistic regression, we want to model y as a linear combination of $\overline{x}$, but in this case, it is not that straightforward.

The logistic regression model looks like the following:

$$f_{\overline{w},b}(\overline{x}) \stackrel{def}{=} \frac{1}{1 + e^{-(wx+b)}}. \tag{3.4}$$

Similar to linear regression, our task is to find optimal values $(\overline{w}^*, b^*)$ for parameters $\overline{w}$ and b.

Once we found $(\overline{w}^*, b^*)$ for the 3.4, in other words, we trained the model, we can apply the model 3.4 on features x_i from an example (x_i, y_i). The output value lies in the range $0 < p < 1$. If y_i is the positive
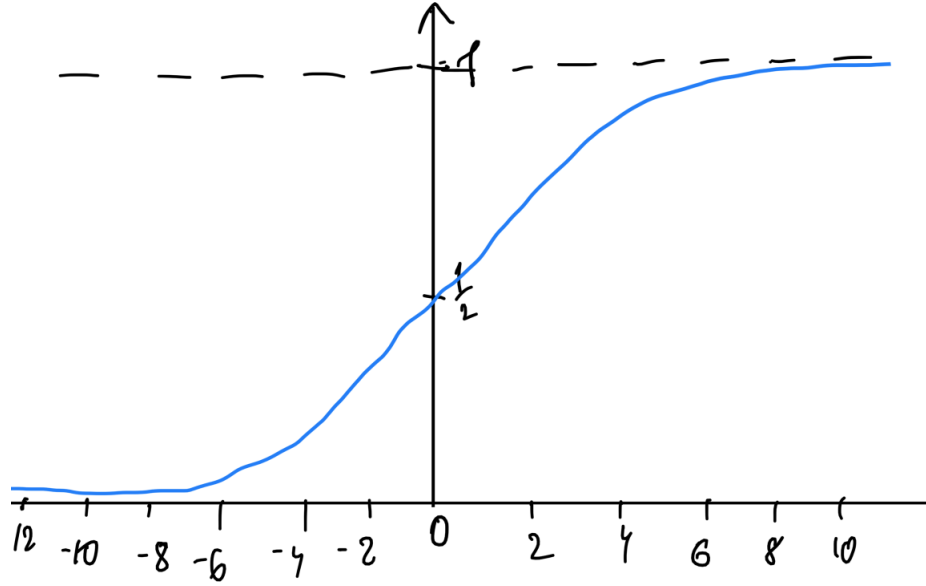
Figure 3.2: Logistic function

class, the likelihood of y_i being a positive class is given by p. Consequently, if y_i is the negative class, the likelihood for it being the negative class is given by $1 - p$.

In the figure we can see that if the y has a value lower than $\frac{1}{2}$, it has negative x values and will be marked as a negative class. If y is greater than $\frac{1}{2}$, it is positive. Although, depending on the context, the threshold may be different.

In logistic regression, instead of *minimizing* MSE we are trying to *maximize* the *likelihood function*. In statistics, the likelihood function tells how likely the example is according to our model. The objective function in logistic regression is called *maximum likelihood*. It looks like the following:

$$L_{\bar{w},b} \stackrel{def}{=} \prod_{i=1...N} f_{\bar{w},b}(\bar{x}_i)^{y_i}(1 - f_{\bar{w},b}(\bar{x}_i))^{(1-y_i)}. \tag{3.5}$$

On the other hand, due to the exponential function in equation 3.5, it is better to use the *log-likelihood* instead, to make calculations easier. As *Log* is a strictly increasing function, maximizing it is the same as maximizing its argument. The solution to this optimization problem is the same as the solution to the original problem. The log-likelihood function looks like the following:

$$LogL_{\bar{w},b} \stackrel{def}{=} ln(L_{\overline{w},b}(\overline{x})) \sum_{i=1}^{N} y_i ln f_{\bar{w},b}(\overline{x}) + (1 - y_i)ln(1 - f_{\bar{w},b}(\overline{x})). \tag{3.6}$$

Unfortunately, there is no closed-form solution for this optimization problem. Nonetheless, the function is convex, hence gradient descent (or any other optimization algorithm) pretty much guarantees the finding of the global minimum, provided that the learning rate is not too large and enough time is given.
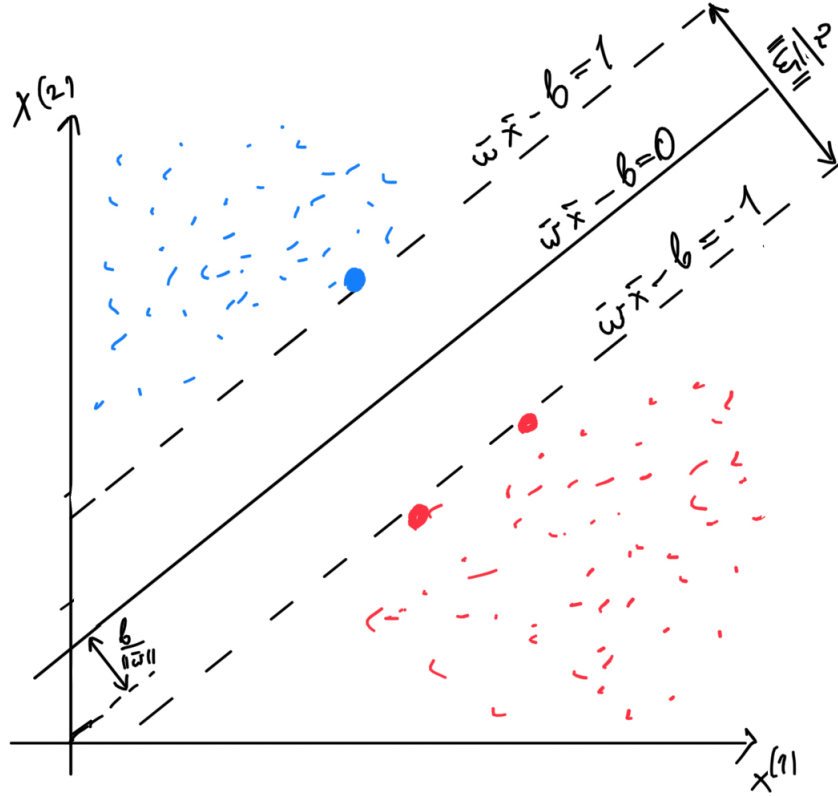
Figure 3.3: SVM demonstration for two-dimensional dataset

3.1.3   Support Vector Machines

Support vector machine (SVM) is a widely-used and powerful machine learning algorithm that can perform a wide range of tasks, including linear and nonlinear classification, regression, and outlier detection on small- to medium-sized datasets.

Linear SVM

In its classical formulation, the support vector machine is a binary classifier. Classes are called positive and negative and are labeled +1 and -1, respectively.

The model is described by the equation

$$f(x) = sign(\overline{w}x - b).$$

The function *sign* returns +1 if the input is positive, and -1 if it is negative. To train the SVM means to find optimal values $(\overline{w}^*, b^*)$ of parameters $\overline{w}$ and b so that the model makes as accurate predictions as possible. The process of finding $(\overline{w}^*, b^*)$ is called training.

The concept behind support vector machines is demonstrated in Figure 3.3. The image consists of two classes represented by red and blue dots, divided by a solid line termed the *decision boundary* $\overline{w}x - b = 0$, with two dashed lines by its sides known as *support vectors* $\overline{w}x - b = 1$ and $\overline{w}x - b = -1$. Support vectors are defined by the closest instances of a class to the decision boundary. These instances are emphasized in the figure.

The distance between the closest instances of two classes is called *margin* and is equal to $\frac{2}{\|\overline{w}\|}$, where $\|\overline{w}\|$ is the Euclidean norm and $\overline{w}$ is a parameter vector of the same dimensionality as the feature vector.
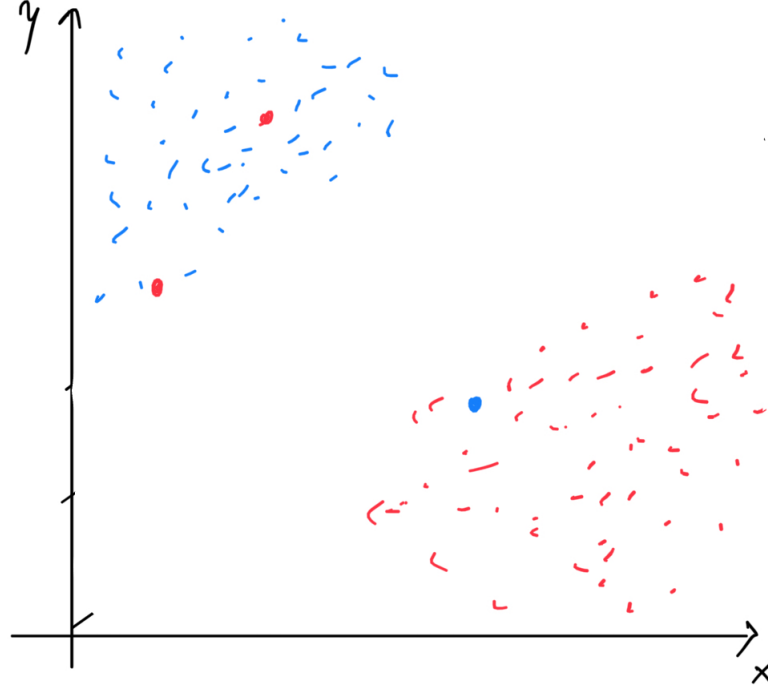
Figure 3.4: Linearly non separable dataset

Thus, the smaller the norm, the larger the margin. The larger the margin, the better the model's generalization. The primary objective of the model is to find the largest possible margin $\frac{2}{\|\overline{w}\|}$, so, to do that we need to *minimize* the Euclidian norm defined by the expression

$$\|\overline{w}\| = \sqrt{\sum_{j=1}^{D} (w^{(j)})^2}.$$

The fundamental assumption of support vector machines is that classes are linearly separable, implying their instances can be separated by a hyperplane (decision boundary) with no examples of one class lying among the ones of the opposite class. It is illustrated in the figure 3.4. In this case, the algorithm won't be able to find an optimal solution with no instances lying between the support vectors and the decision boundary. Consequently, the model is highly sensitive to outliers.

Every optimization problem requires constraints, and for the support vector machine, they are the following:

1. $\overline{wx_i} - b \geqslant +1$ if $y_i = +1$

2. $\overline{wx_i} - b \leqslant -1$ if $y_i = -1$.

These two equations can be reduced to one $y_i(\overline{wx} - b) \geqslant 1$.

The optimization problem we want to solve is the following: Minimize $\|\overline{w}\|$ subject to constraint $y_i(\overline{wx_i} - b) \geqslant 1$ for $i = 1, ..., N$, where N is the number of features. This problem can be modified so that the quadratic programming techniques could be used in the optimization process. The modified formula is $\frac{1}{2}\|\overline{w}\|^2$, and minimization of it would also mean minimization of $\|\overline{w}\|$. The updated optimization problem looks like this:

$$min \frac{1}{2}\|\overline{w}\|^2 \text{such that } y_i(\overline{wx_i} - b) \geqslant 1, i = 1, ..., N \tag{3.7}$$

Handling Noise

To introduce the ability of SVM to handle nonlinearly separable data (but not to the extreme), we define the hinge loss function: $max(0, 1 - y_i(\overline{wx_i} - b))$. It is zero if the constraints 1 and 2 are satisfied. If it is not, the data point does not lie on the right side of the decision boundary. The function value is proportional to the distance from the decision boundary. The resulting cost function looks like the following:

$$C\|\overline{w}\|^2 + \frac{1}{N} \sum_{j=1}^{N} max(0, 1 - y_i(\overline{wx_i} - b)), \tag{3.8}$$

where C is the hyperparameter that determines the trade-off between increasing the size of the decision boundary and ensuring that each x_i lies on the correct side of the decision boundary. Its value is chosen experimentally. C handles the trade-off between classifying the training data well and classifying future examples well (generalization). For higher values of C, the misclassification error will be almost negligible, so the algorithm will try to find the highest margin without considering it. For lower values of C, the algorithm will try to make fewer mistakes by sacrificing the margin size. (A larger margin is better for the generalization.) Lower values lead to wider streets and more margin violations, higher values lead to narrower streets and fewer margin violations.

SVM with the hinge loss function is called *soft-margin SVM* while the original formulation that optimizes the Euclidian norm is referred to as *hard-margin SVM*. *Soft margin classification* tries to mitigate the downsides of the *hard margin classification* by trying to find a balance between keeping the margin as large as possible and mitigating the margin outliers (instances that lie on the margin or on the opposite side).

Handling Non-linearity

We can adapt SVM to work with nonlinearly separable datasets by applying the kernel trick. The kernel trick means transforming the original space to a higher dimensional one during the cost function optimization with the hope that, in higher dimensional space, it will become linearly separable. In mathematical language: the kernel trick is mapping $\varphi : \overline{x} \to \varphi(\overline{x})$, where $\varphi(\overline{x})$ is a vector of higher dimensionality than $\overline{x}$. The kernel trick allows us to save a lot of non-necessary computations.

There are multiple kernel functions. The most widely used are linear, polynomial, radial basis function (RBF),

3.2   Unsupervised Learning

Unsupervised learning deals with a dataset that does not have labels. There are three main branches of unsupervised learning: clustering, dimensionality reduction and anomaly detection. *Clustering* is a method that identifies similar instances and groups them into sets. It has applications in data analysis, namely, *exploratory data analysis (EDA),* customer segmentation, dimensionality reduction, and anomaly detection. Clustering might be either soft, where an instance has a score of belonging to a particular cluster, or hard, where an instance belongs to only one class. The score might be the distance from the cluster centroid or an affinity (similarity score).

Dimensionality reduction is useful for visualization and for the acceleration of learning. Datasets often have a lot of redundant data or the task requires a lot of features. Many algorithms, such as linear models, SVMs, decision trees, might have their performances compromised due to high-dimensional data. So called *curse of dimensionality* states that high dimensional data can cause slow learning and prevent us from getting an optimal model. Consequently, the reduction of the data dimensionality might be a good idea. However, it is worth noting that a dimensionality reduction algorithm might lose some useful information. A lot of modern algorithms, such as neural networks or ensemble algorithms, handle high dimensional data very well, and dimensionality reduction techniques are used less than in the past. However, they are still used for data visualization and cases when we need to build an interpretable model while we are limited in theb number of algorithms we can use.

Anomaly (outlier) detection involves the detection of instances strongly deviating from the norm. These instances are called *outliers* or anomalies while regular ones are referred to as *inliers*. Anomaly detection has many applications. For example, it can be used as a data preprocessing step to remove outliers from the dataset, which might improve the performance of the resulting model. In addition, it is used in the *fraud detection* task and the detection of faulty products in manufacturing facilities.

Novelty detection is closely related to anomaly detection. The only difference is that novelty detection assumes that the training dataset was not contaminated by outliers while anomaly detection does not make this assumption.

3.2.1 Principal Component Analysis (PCA)

Principal components are vectors that define a new coordinate system. The first vector goes in the direction of the highest variance. The second vector is orthogonal to the first one and goes in the direction of the second highest variance, and so on. If we were to reduce dimensionality to $D_{new} < D$, we would pick D_{new} largest principal components and *project* instances onto them.

It is not advised to choose the number of dimensions arbitrarily. It is recommended to choose a number of dimensions that preserves a large amount of variance (e.g. 95%), or in case of visualization to reduce the number of dimensions down to two or three. There are different versions of PCA; kernel PCA, Incremental PCA (online or batch PCA), and Randomized PCA.

3.2.2 Gaussian Mixtures

Gaussian mixtures is a common algorithm that can be used for anomaly detection. Gaussian mixtures assume that the dataset is generated by several Gaussian distributions. Any instance lying in a region of low density is an anomaly. The density threshold has to be specified. If one gets too many false positives (good products labeled as faulty) they need to decrease the threshold. Consequently, if we get too many false negatives (faulty products labeled as good) the threshold has to be increased. Gaussian mixtures belong to soft clustering. Gaussian mixtures require the number of clusters to be specified. It needs to be run a couple of times to avoid suboptimal solutions.

3.3 Data Preparation

Due to factors such as curse of dimensionality and inherent noise, we cannot load raw data to an algorithm and expect good performance. Most often, the raw data has too many features and most of them have very little predictive power. We need to build a dataset first. *Feature engineering* is responsible for transforming raw data into a dataset. It is a labor-demanding process that requires creativity and, most importantly, domain knowledge.

The objective of this stage is to create *informative* features or features with *high predictive power*. For example, in our task of predicting survival time, donor-recipient blood group compatibility or recipient's age is likely to have much higher predictive power than the donor's or recipient's citizenship.

Moreover, it is possible to create new features with higher predictive power out of those with low predictive power. For example, the calculation of *estimated Glomerular Filtration Rate (eGFR)*, the metric of kidney function estimated on a patient's age, gender, and serum creatinine level, could potentially give more information to the learning algorithm than all features separately.

In the following subsections, we will cover some popular feature engineering techniques.

3.3.1 Handling Categorical Features

The majority of machine learning algorithms primarily operate with numerical features. To handle categorical features (the ones with only a few possible values), such as the age group or a blood group, we can use *one-hot encoding* to convert them to several binary ones. For instance, let's consider a blood group feature comprised of four primary blood groups: A, B, AB, and O. We can convert each blood group into a vector of four numerical values:

$$A = [1, 0, 0, 0]$$
$$B = [0, 1, 0, 0]$$
$$AB = [0, 0, 1, 0]$$
$$O = [0, 0, 0, 1]$$

This technique will increase the dimensionality of the dataset but this is a trade-off we have to make because if we were to assign a number to each group (1 to A, 2 to B, etc.), that would imply gradation or ranking among these categories, while there is none.

However, if the categorical feature does suggest some gradation, for example, university marks as "fail", "average", "good", or "excellent", an enumeration of each value will be appropriate. This practice of assigning a number to categories that have ranking is called *ordinal encoding*.

Binning (or *bucketing*) is the technique used for converting numerical values into multiple binary features called *bins* or *buckets*. For example, a patient's age can be transformed into age-range bins: 0 to 18 years old, 18 to 25 y.o., 25 to 40 years old, and so on. This technique might help an a laearning algorithm learn better, particularly with smaller datasets.

3.3.2 Feature Scaling

Different ranges of feature values might pose a problem to some machine learning algorithms as they do not handle them very well. It might result in a slower training time or a poorer performance. This problem is solved by *normalization* and *standardization* scaling techniques.

Normalization (also known as *min-max scaling*) is a technique of converting an actual range of numerical feature values into a standard range of values: $[-1, 1]$ or $[0, 1]$ without losing any information. The normalization formula for value $x^{(j)}$ for feature j, looks like the following:

$$\bar{x}^{(j)} = \frac{x^{(j)} - min(j)}{max(j) - min(j)},$$

where $min(j)$ and $max(j)$ are minimal and maximal values of feature j.

Standardization is a scaling technique that scales numerical data in such a way that after scaling, it has properties of the *standard normal distribution* with the mean μ=0 (average value) and the standard

deviation from the mean $\sigma = 1$. The standardization formula for value $x^{(j)}$ for feature j, looks like the following:

$$\hat{x}^{(j)} = \frac{x^{(j)} - \mu^{(j)}}{\sigma^{(j)}}.$$

Typically, standardization is used for supervised learning, in case feature values are formed by standard distribution (bell curve) or a feature has outliers. In other cases, the normalization is preferred.

3.3.3 Handling Missing Feature Values

Datasets frequently have missing values and to handle them, we have one of the following options:

1. **Removal of rows with missing values**. The most direct and straightforward approach to managing missing data. If missing values are sparse or the dataset is large enough, the usage of this technique would be appropriate.

2. **Feature removal**. If the dataset has a feature with an excessive amount of missing values relative to its size, it is better to remove the feature.

3. **Regression imputation**. This technique implies the filling in a missing feature value with predictions of a machine learning regression algorithm.

4. **Mean/median imputation**. This method involves the filling of missing feature values with their mean or median value.

5. **Constant value imputation**. This technique entails the filling the missing values with clearly too high or too low values. The motivation is for the algorithm to discern the value as an outlier while considering other features. This method is not recommended as it can introduce bias.

It is often impossible to tell which data imputation method would work the best and therefore, it should be checked experimentally.

3.4 Model Training and Hyperparameter Tuning

It is a common practice to divide a dataset into three parts

- Training set (70% of the dataset)

- Validation set (15% of the dataset)

- Test set (15% of the dataset)

The training set, being the largest of them, is employed to train the machine learning model. Validation and test sets, which are of identical sizes and often called hold-out sets, are used in subsequent stages of model evaluation.

The rationale behind the use of separate training and validation sets is to prevent overfitting - a situation when the model performs well on the training data but poorly on the unseen data. Overfitting can occur if the model is tested and evaluated on the same dataset. As a result, the model may memorize the training examples and fail to make accurate predictions on the unseen data. To alleviate this, we use the validation set to fine-tune the model, and the test set to assess its performance before deploying it to production.

A typical workflow involves training the model on the training set, validation on the validation set using the selected metric, then adjusting the model's parameters to improve its performance. This process is repeated until no substantial improvement is observed. Finally, the model's performance is assessed on the test set. This iterative process is referred to as hyperparameter tuning.

An alternative to the three-set technique is *k-fold cross-validation*. This technique involves splitting the dataset into k subsets, or folds, of equal size. One fold is used as a validation set, while the other k-1 folds constitute a training set. The model is trained exactly k times, with each fold serving as a validation set only once. The only drawback is that it is highly computationally demanding, particularly with a high k value and larger datasets, as the model will be trained k times.

A *hyperparameter* is a parameter specified before model training, in contrast to regular parameters that are calculated during training. Each model possesses a different set of hyperparameters and they profoundly influence the model's performance. The number of trees in Random Forest and the C hyperparameter in Support Vector Machines are examples of hyperparameters. The task of finding the optimal combination of hyperparameters is called hyperparameter tuning. One strategy might be to select hyperparameters manually and observe their impact on the performance. However, utilizing the grid search is a better way.

Grid search is a standard way of performing hyperparameter fine-tuning. It includes defining hyperparameters to experiment with, providing values for each hyperparameter to be tested, and training a model for each possible combination of hyperparameters. The performance of each individual model is assessed using k-fold cross-validation and the best combination of hyperparameters is selected. This approach is used in scikit-learn's implementation - GridSearchCV.

Grid search proves to be effective when dealing with relatively few hyperparameter combinations. However, with larger number of hyperparameter combinations, it is advisable to use RandomizedSearch (RandomizedSearchCV in scikit-learn). This method is very similar to grid search but instead of trying every possible combination of provided values, it tests only a specified number of randomly selected hyperparameter combinations. The primary advantage of this method over grid search lies in more control over computational power and the time dedicated to hyperparameter tuning.

3.5 Survival Analysis

Survival analysis, often referred to as *time-to-event analysis,* is a statistical technique employed to analyze and predict the time until an event of interest occurs. Its name originates from clinical and biological research where these methods are used to analyze survival time. These methods, however, found their use in areas far beyond clinical settings: in business to predict the time until the customer "churns" from a subscription; in engineering to estimate the product longevity or the longevity of its parts; in social sciences to estimate the longevity of a marriage; or to estimate a student dropout rate in an academic setting.

In the context of survival analysis, *time* (also *survival time*) refers to the duration from the start of an individual's follow-up to the occurrence of an event. It is measured in days, weeks, months, or years [13]. The term *event* (also referred to as *deat*h or *failure*) encompasses any occurrence that permanently changes the state of the subject. It can be death, the onset of a disease, a relapse from remission, a recovery, or any other specified experience of interest that an individual might encounter [13, 30]. Usually, only one event of interest is considered. When evaluating multiple events, the problem is categorized as *competing risks* or *recurrent events* problem [13].

In this section, we will delve into fundamental survival analysis terminology, such as censoring and survival function. We will explore the classification of the survival analysis methods, highlighting the
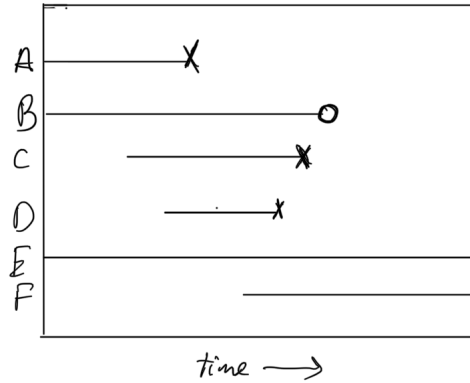
Figure 3.5: Censoring illustration

most popular statistical methods, as well as one machine learning method tailored to the needs of survival analysis. Additionally, we will discuss ways to evaluate survival models effectively.

3.5.1 Basic Terminology

In this subsection, we are going to cover the basic terminology required for survival analysis such as censoring, censoring assumptions, survival, and hazard functions.

Censoring

The most distinct feature of survival analysis methods is the ability to handle censored data. *Censoring* refers to a circumstance when the information about survival time is only partially known. For example, the dataset utilized in our research has 370,000 censored instances out of 500,000 performed transplantations. These patients were either still alive at the last date of observation or were lost to follow-up. This lack of complete information indicates *censoring* in survival analysis. Censoring makes the application of standard statistical and machine learning approaches to survival data impractical [38].

Look at the Figure 3.5. On the y-axis, we can see individual patients, while the x-axis corresponds to the study timeline (the right side is the end of the study). Cross (X) denotes an occurrence of the event, and circle (O) corresponds to the patient's exit from the study.

There are three types of censoring: *left*, *right*, and *interval* censoring. *Right censoring*, which is more common, occurs when we are sure that the event did not happen by a specific time and we don't know when it will happen. The situation arises when the patient drops out of a study, or the study ends when they are still alive, as illustrated with patients B, E, and F in Figure 3.5.

Left censoring is less common and happens when the event occurs before the study begins or before the initial observation. We know the event happened before a specific time, but the exact time is unknown. This type is typical in cases where a patient has already experienced the event (e.g., developed a disease) before enrolling in the study.

Interval censoring happens when the event occurs within a particular timeframe, but the exact time is unknown. It can be the case in studies involving periodic patient follow-up, where the event can happen at any point between two visits.

Understanding right, left, and interval censoring is essential in survival analysis. We will next turn our attention to the assumptions associated with these censoring types. These assumptions are inherent to many survival analysis methods and are critical in selecting an appropriate technique.

Censoring Assumptions

There are three types of censoring assumptions: *random*, *independent*, and *non-informative*. Each shares certain similarities but also possesses unique distinctions which we will explore in detail. Censoring assumptions is the way censoring is managed.

1. **Random Censoring**: Subjects censored at time *t* are assumed to have the same failure rate as remaining subjects provided the same survival experience. Subjects that were censored are selected randomly, meaning the study does not influence or bias which participants are censored.

2. **Independent Censoring**: Independent censoring occurs when the censoring is random within certain subgroups, defined by specific covariates. If no covariates are present, it defaults to random censoring. This distinction might not be apparent when examining a single subgroup.

3. **Non-Informative Censoring**: Non-informative censoring occurs when censored instances do not provide any information on their survival prospects. In other words, whether or not the patient is censored, has no influence on experiencing the event.

Generally, it is safe to assume *non-informative* censoring when censoring is *independent* and/or *random*. However, these assumptions are not equivalent. To better understand these concepts, let us look into some examples.

Consider a three-year disease occurrence study with 100 subjects at risk (group A). By the end of the study, 20 of them contracted the disease, giving a 20% three-year disease risk. Suppose we want to extend the study for another two years on the remaining 80 individuals. However, 40 refuse to continue in the study and, are, therefore, lost to follow-up (censored). Of the remaining 40, 5 contracted the disease. Assuming those who left were representative of the remaining subjects (random and independent censoring), another 5 among the censored would have contracted it. Consequently, the five-year risk is 30% and the five-year survival is 70% under random and independent censoring assumptions. In this case, random and independent censoring are the same, as no predictor variables are considered.

To illustrate the difference between random and independent censoring, let us introduce another group to the study: group B with 100 individuals. In the first three years, 40 contracted the disease, and 10 left the study. So, the calculated three-year risk for group B is 40%. In the next two years, 10 out of 50 get the disease, yielding 20% risk for years between 3 and 5. Under the independent censoring assumption, we assume that out of 10 censored, 2 contracted the disease. The five-year risk for group B is 52% with 48% survival under independent censoring assumptions.

As we can see, the five-year risk in the two groups differs significantly (30% against 52%), and the censoring proportion is also very different (50% against 17%). Hence, the overall censoring is not random. However, it is random within each group, so the censoring is independent. On other hand, if in group B, 30 subjects out of 60 were censored at the three-year mark, the censoring proportion would be the same in both groups, and the overall censoring would be random as those censored would be the representatives of those who remained at risk.

To best illustrate what non-informative censoring is, let us demonstrate informative censoring. Let us take a group of subjects under random and independent censoring assumptions. Every time subject A gets an event, subject B leaves the study (e.g., B is A's relative). If the censored subjects are representative of subjects at risk it would be random and independent censoring. Here, the censoring mechanism is directly related to the event occurrence, so the censoring is informative.
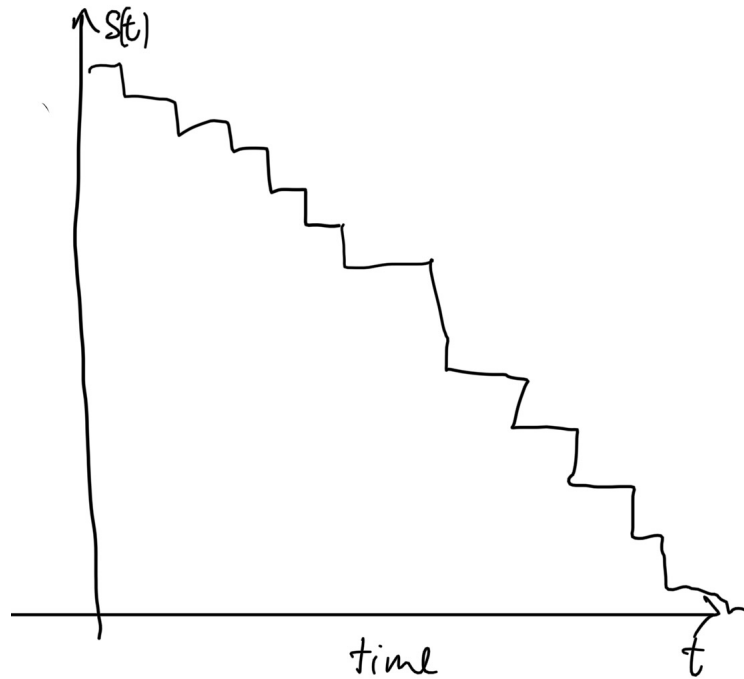
Figure 3.6: Survival function

Survival Function

The *survival function*, also known as the *survivor function*, denoted by $S(t)$, represents the probability that the patient survives, in other words, does not experience the event of interest beyond the given time t. Mathematically, it is denoted by:

$$S(t) = P(T > t), \tag{3.9}$$

where P represents the probability, t is any specific time of interest, and T is the random variable for the subject's survival time. For instance, if we want to know the likelihood that a patient is going to live for more than five years after a kidney transplant, we set t equal to 5, and we evaluate $S(t)$ to determine the probability that T, actual survival time, is greater than 5 years.

The survival function has several key characteristics:

1. It continually decreases or maintains its value over time, theoretically extending from 0 to infinity. That is, if the study lasted indefinitely, the survival function would eventually fall to 0. In practical research scenarios, studies do not last forever, and not every patient experiences an event by the end of the study.

2. As it represents a probability, the function value ranges from 0 to 1. At the beginning of the observation period it starts at 1, indicating 100% survival probability, and declines over time, potentially reaching 0 as the probability of survival decreases.

3. Although the graph of the survival function is smooth by definition, in reality it is a step function. This stepwise representation is due to the nature of real-world data, where events are recorded at specific, discrete time points rather than continuously [13].

Survival function $S(t)$ gives a comprehensive understanding of the probability of an individual surviving beyond a specific time point. It presents a declining or constant trend starting at 1 and potentially declining to 0. While understanding the probability of not failing at every time point is important, knowing the instantaneous risk might also be useful, leading us to the next tool: the hazard function.

Hazard Function

The survival function provides the probability of an individual surviving at each given point in time. This function is often preferred over the *hazard function* due to its more intuitive appeal: it directly communicates the chance of survival. However, there are scenarios where knowing the risk at each point in time is necessary, entailing the use of the hazard function.

The *hazard function*, denoted as $h(t)$, represents the instantaneous potential per unit of time for the event to occur, provided the subject's survival up to time t. It is expressed by the following equation:

$$h(t) = \lim_{\Delta t \to 0} \frac{P(t \leq T < t + \Delta t | T \geq t)}{\Delta t}. \tag{3.10}$$

Here, Δt represents an infinitesimally small increment of time. The function $h(t)$ is equal to the limit, as Δt approaches zero, of a conditional probability, divided by Δt. The conditional probability statement gives the probability that a person's survival time T will lie in the time interval between t and $t + \Delta t$, given that the survival time is greater than or equal to t.

Occasionally, due to its structure, the hazard function is referred to as a *conditional failure rate*. It is a rate because it represents a conditional probability per unit of time Δt, and it is conditional on the subject surviving until time t. Unlike probability, this rate has a scale from 0 to infinity — depending on the measure of time in days, weeks, or years. Essentially, by considering the limit as the Δt approaches zero we basically get the instantaneous potential of failing at time t, given survival until that moment.

To best illustrate the concept of instantaneous potential, let us refer to the concept of velocity. Velocity gives us the speed at a specific point in time. For instance, the velocity of 80 km/h, means that maintaining the same speed for an hour would result in traveling 80 kilometers. However, it doesn't predict how much the car will travel in reality. The same works with the instantaneous potential: it might be high at one point but low at another, reflecting that both are measurements at an instant in time rather than an interval [13].

The cumulative hazard function further extends our understanding by quantifying the accumulated risk over time. It is the area under the hazard function that allows us to say which group has a greater risk. It is defined by the following function:

$$H(t) \overset{def}{=} \int_0^t h(u)du, \text{ where } t > 0. \tag{3.11}$$

Where $h(u)$ represents a hazard function. This integral measure enables a comprehensive comparison of risk between groups, showing which has a greater risk from the perspective of accumulated potential for the event to occur over time.

The Relationship Between the Survival and Hazard Functions

Some models, such as Cox Proportional Hazards, are written in terms of the hazard function, and it is necessary to be able to estimate the survival function out of the hazard function to make the model more flexible and accessible to a larger audience. Fortunately, there are ways to convert one into the other and vice versa. In this subsection, we are going to describe techniques for these conversions.
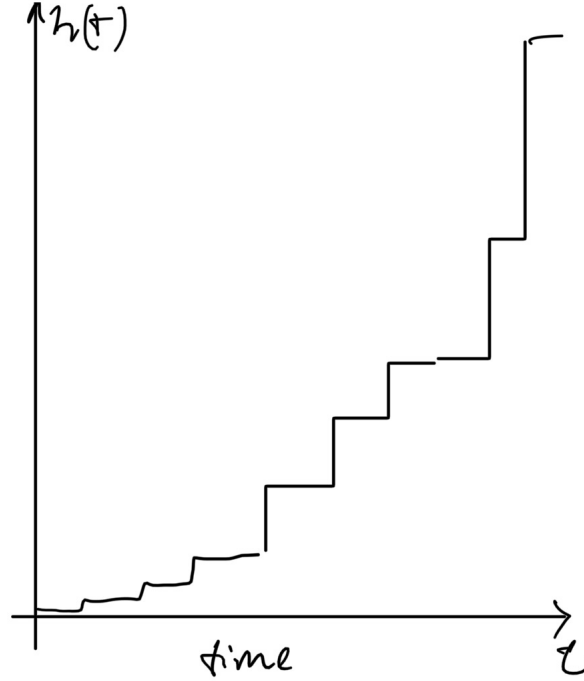
Figure 3.7: The example of a hazard function. Note that it may significantly differ.

Let us start by considering how to obtain the survival function $S(t)$ from the hazard function $h(t)$. The relationship is captured in the following equation:

$$S(t) = exp\left[-\int_0^t h(u)du\right].$$ (3.12)

Equation 3.12 illustrates that the survival function $S(t)$ is equal to the exponential of the negative cumulative hazard function from zero to t. As we can see, the integral in the equation is the $H(t)$ from the 3.11. And we can simplify our equation to:

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

On the other hand, to obtain the hazard function from the survival function, we can utilize the following relationship:

$$h(t) = -\left[\frac{dS(t)/dt}{S(t)}\right].$$ (3.13)

Here, Equation 3.13 denotes that the hazard function $h(t)$ is the negative derivative of the survival function $S(t)$ with respect to time t divided by $S(t)$.

Considering the fact that the survival function describes the probability of a patient surviving up to time t, and the hazard function shows the instantaneous risk of a person dying at a specific instant t, we can say that they provide complementary information about survival and risk over time [13]. Of the two discussed functions, the survival function is used more often as it is more intuitive. In the practical part, we will estimate the survival function as well. Fortunately, we will not convert them manually. The *scikit-survival* library (more on it in the Section 3.6) will do that for us.

3.5.2 Taxonomy of Survival Analysis Methods

Survival analysis methods can be broadly categorized into *statistical* and *machine learning based* methods. Both aim to estimate the survival time and the survival probability at the estimated survival time [36]. Statistical methods primarily characterize distributions of the event times and the statistical properties of the parameter estimation by estimating the survival curves. Typically, statistical methods are employed for low-dimensional data [36].

On the other hand, machine learning methods focus on the prediction of the event occurrence at a given time. They harness the strengths of traditional survival analysis while integrating different machine learning techniques, leading to more potent algorithms. Machine learning methods are mostly used with high-dimensional data [36].

Statistical methods can be further subdivided based on their assumptions and parameter usage into parametric, non-parametric, and semi-parametric methods. Machine learning methods, encompassing methods like survival trees, ensembles (random survival forests), neural networks, and support vector machines form a distinct category. Advanced machine learning techniques, such as active learning, transfer learning, and multitask learning are included as well [36].

In the following sections, we will cover selected statistical methods and one machine learning technique. Machine learning techniques related to neural networks will be covered in Section **??**, dedicated to these advanced survival analysis techniques.

3.5.3 Statistical Methods

This section provides a concise overview of statistical techniques. Here, we introduce three different types of statistical methods that are commonly used to estimate the survival and hazard functions: *non-parametric*, *semi-parametric*, and *parametric methods*.

Non-parametric methods are preferred in situations where the event time does not adhere to any known distribution or when the proportional hazards assumption is not met [36]. There are three main non-parametric methods: the Kaplan-Meier (KM) method, the Nelson-Aalen (NA) estimator, and the Life-Table (LT) method. In the next section, we will cover Kaplan-Meier in more detail. The Life-Table method is more convenient than Kaplan-Meier for the estimation of survival curves when data subjects are segmented into distinct time intervals when dealing with an extensive number of subjects or a broad population scope. On the other hand, the Nelson-Aalen method is used for the estimation of hazard functions [36].

Semi-parametric models offer a middle ground between fully parametric models, which make specific distributional assumptions, and non-parametric models, which make very few assumptions. Among semi-parametric methods, the Cox model is the most frequently employed model for survival regression analysis. It is semi-parametric as the distribution of the outcome is unknown. Unlike other approaches, this method is based on the proportional hazards assumption and uses partial likelihood for parameter estimation (more on that in 3.5.3.2). There are a couple of variants of the basic Cox model: the penalized Cox model, which will be used in the practical part of this work, the CoxBoost algorithm, and the Time-Dependent Cox model [36].

Parametric methods shine in their accuracy and efficiency when the time to the event conforms to a known distribution that can be specified in terms of certain parameters. With parametric models, estimating the time to the event is straightforward, whereas the Cox model can make this task somewhat cumbersome or unfeasible. In the domain of parametric models, linear regression is central. However, the Tobit model, Buckley-James regression, and penalized regression are the most favored. Beyond this, other parametric models like the Accelerated Failure Time (AFT) have gained traction. The AFT model represents survival time as a function of covariates [36].
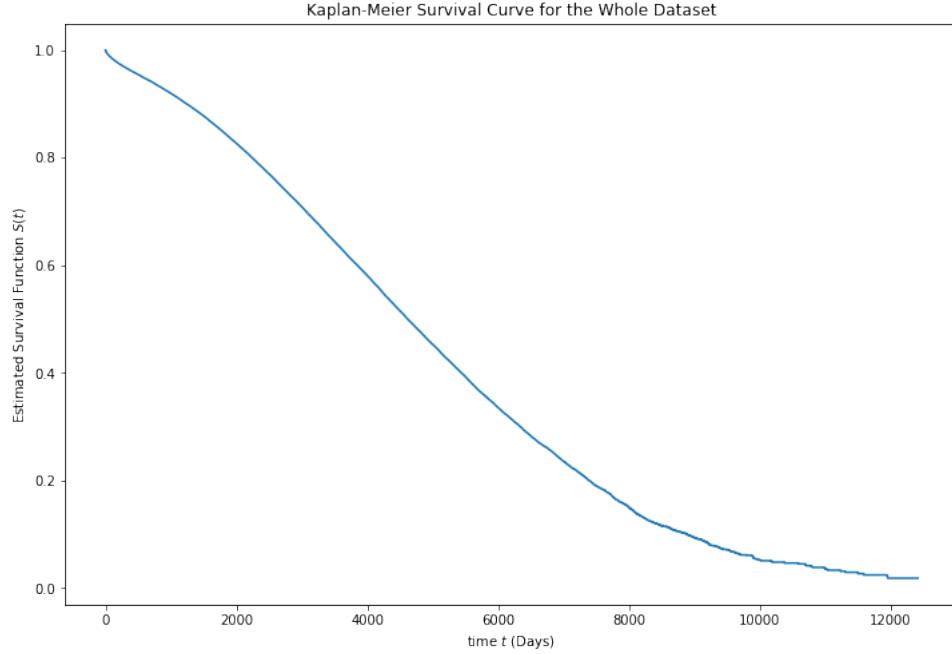
Figure 3.8: Survival curve estimated with KM

To conclude, statistical methods in survival analysis can be broadly categorized into non-parametric, semi-parametric, and parametric techniques, each with its unique strengths and applications. Among non-parametric methods, Kaplan-Meier stands out as particularly significant. It provides a robust and intuitive way to estimate survival functions. The following section delves deeper into the Kaplan-Meier method.

3.5.3.1 Kaplan-Meier Survival Curves

Kaplan-Meier is a non-parametric method of survival function creation. It is *non-parametric* because it does not take into account any covariates, or parameters, and requires only the survival time and the censoring indicator. It works under an independent censoring assumption. The general Kaplan-Meier formula for plotting the survival function is the following:

$$\hat{S}(t_{(f)}) = \hat{S}(t_{(f-1)}) \times \hat{P}(T > t_{(f)}|T \geq f_{(f)}). \tag{3.14}$$

That can be read as the survival function $\hat{S}$ of time $t_{(f)}$ is equal to the probability of surviving past the previous time point $t_{(f-1)}$ times the conditional probability of surviving past the time $t_{(f)}$ [13]. This function can also be expressed as a product limit:

$$\hat{S}(t_{(f)}) = \prod_{i=1}^{f-1} \hat{P}(T > t_{(i)}|T \geq f_{(i)}).$$

In the Equation 3.14, we replaced $\hat{S}(t_{(f-1)})$ with the product of all fractions estimating the conditional probabilities for failure times t_{f-1} and those preceding it [13]. Because of that the Kaplan-Meier estimator is sometimes referred to as the *product-limit method [36]*. Figure 3.8 shows the survival curve created with the KM method on the dataset used in this work.

These survival curves are often compared using the log-rank test. The *log-rank test* is a way to compare two survival functions, that is often used in studies, where there is a target group and a placebo (control) group to assess the efficacy of the treatment or intervention in the study by comparing the survival curves of the two groups [13].

To summarize, the Kaplan-Meier method provides a robust non-parametric approach for estimating survival functions, emphasizing its independence from covariates. While the log-rank test offers a mechanism to compare the survival curves and assess treatment efficacy, there are still situations that require taking covariates into consideration. That leads us to the Cox proportional hazards model, which inherently handles covariates.

3.5.3.2 Cox Proportional Hazards Method

The *Cox proportional hazards* (also Cox PH) model is widely used in survival analysis semi-parametric model. This section explores its formulation, key properties, and reasons for its widespread use in research.

The Cox proportional hazards model is defined in terms of a hazard at time t for a subject with a given vector of explanatory variables $\mathbf{X}$:

$$h(t, \mathbf{X}) = h_0(t) \times exp\left[\sum_{i=1}^{p} \beta_i X_i\right],\tag{3.15}$$

where $h_o(t)$ stands for the *baseline hazard function*. Coefficients β are the parameters of interest in the model. Since the exponential function has no t, $\mathbf{X}$ is called *time-independent*. The exponential function ensures that the function is non-negative, satisfying the definition of the hazard function[13].

Even though Equation 3.15 contains the baseline hazard function, the function is not specified. Fortunately, we can calculate the hazard ratio, a measure of effect, without having to estimate the baseline hazard function. Similarly, the hazard function $h(t, \mathbf{X})$ and the survival function $S(t, \mathbf{X})$ can be estimated without the baseline function. So, with minimal assumptions, we can estimate everything we need (h, S, and HR).

The *hazard ratio* (HR) is a measure of the influence of an intervention on the outcome. A hazard ratio is defined as the hazard for one individual divided by the hazard for the other, as illustrated with the following formula:

$$\hat{HR} = \frac{h(t, X^*)}{h(t, X)} = exp\left[\sum_{i=1}^{p} \hat{\beta}_i(X^* - X)\right]\tag{3.16}$$

As can be seen, the equation does not contain t and the basic hazard function, as they are canceled out, making it a *proportional hazard assumption*.

Like logistic regression, the CoxPH uses the *maximum likelihood* function 3.5 to calculate its parameters ($\hat{\beta}_i$). However, since the maximal likelihood considers only a part of patients, namely those who experienced an event, the formula is called ***partial** likelihood* [13].

Let us define the partial likelihood. The subject's survival can be defined by $h(t, X)dt$ with $dt \to 0$. Consider J (where $J \le N$) as the total number of events of interest observed for N instances. $T_1 < T_2 < \cdots < T_J$ represents the unique and sequentially ordered times to the event of interest. Let X_j be the vector of covariates for a subject who experiences the event at time T_j. R_j is the set of risk subjects at T_j. Given that the event happens at time T_j, the individual probability associated with X_j can be described as follows:

$$\frac{h(T_j, X_j)dt}{\sum_{i \epsilon R_j} h(T_j, X_i)dt}.$$

By taking the product across all subject's probabilities we get the partial likelihood. Based on Cox assumption and the presence of censoring, partial likelihood is defined as follows:

$$PL(\beta) = \prod_{j=1}^{N} \left[\frac{exp(X_j\beta)}{\sum_{i \epsilon R_j} exp(X_i\beta)} \right]^{\delta_J}. \qquad (3.17)$$

δ_j is the indicator of censoring of a given instance (1 for event occurrence, 0 for censoring). Therefore, if the subject is censored, $\delta = 0$, the individual probability is equal to 1 and the subject does not affect the result. The vector of coefficients $\hat{\beta}$ is estimated by either maximizing partial likelihood, defined above, or maximizing the negative *log-partial likelihood* to improve the efficiency:

$$LL(\beta) = - \sum_{j=1}^{N} \delta_j \{X_j\beta - log[\sum_{i \epsilon R_j} exp(X_i\beta)]\}.$$

[36]

Being a semi-parametric, this model is a safe choice because it consistently delivers a sufficiently reliable result. The risk of choosing the wrong model as often happens with parametric models, is practically non-existent. However, if one is sure that a parametric model suits the problem, one should use the parametric model [13].

3.5.3.3   Penalized Cox Models

The Cox proportional hazards model is often chosen, since its coefficients can be interpreted in terms of hazard ratios, offering meaningful insights. Yet, when estimating coefficients for many features, the standard Cox model collapses due to matrix inversion getting disrupted by correlations between features. Feature (column) correlations make a matrix singular, i.e. impossible to revert. In this section, we aim to explore the *Ridge regression*, *LASSO*, and *Elastic Net* methods as extensions or modifications to the Cox model. These techniques address the inherent challenges in the standard Cox model, offering solutions for regularization and feature selection.

Ridge   By incorporating an l_2 penalty term on the coefficients, shrinking them to zero, we can avoid the problem of the inability to revert singular matrix. Consequently, our objective has the following form:

$$arg \max_{\beta} \log PL(\beta) - \frac{\alpha}{2} \sum_{j=1}^{p} \beta_j^2.$$

In the equation, $PL(\beta)$ denotes the partial likelihood of the Cox model (3.17), terms $\beta_1, \ldots, \beta_p$ represent the coefficients corresponding to p features, $\alpha \geq 0$ is a hyper-parameter that controls the amount shrinkage. The resulting objective is referred to as *ridge regression*. If α is set to zero, we get the regular Cox model [37].

LASSO   While the l_2 ridge penalty solves the mathematical problem of fitting the Cox model, we would still need to take into account all features, no matter how many there are. Preferably, we would like to select a small subset of the most predictive features and ignore the rest, as too many features might result in overfitting. *LASSO* (Least Absolute Shrinkage and Selection Operator) does exactly that. Rather

than merely shrinking the coefficients to zero, it performs feature selection as a part of the optimization process, where a subset of coefficients is set to zero and is, therefore, excluded, reducing the number of features we would need for prediction. Mathematically, we would replace the l_2 penalty with the l_1 penalty, leading to the following optimization problem:

$$arg \max_{\beta} \log PL(\beta) - \alpha \sum_{j=1}^{p} |\beta_j|.$$

The main drawback is that we cannot directly control the number of features selected. However, the value of α essentially determines the number of features selected. To achieve a refined model that requires fewer features, we need a data-driven way to determine the appropriate α. This can be accomplished by first determining the α that would ignore all features (coefficients are set to zero) and then gradually decreasing its value, possibly down to 1% of its starting value. Fortunately, it is implemented in scikit-survival's *sksurv.linear_model.CoxnetSurvivalAnalysis*. We would need to set *l1_ratio=1.0* and *alpha_min_ratio=0.01* to search for 100 α values up to 1% of the estimated maximum [37].

Elastic Net   The LASSO method is effective for selecting a subset of discriminative features. However, it is not without its shortcomings. The first is that LASSO is unable to select more features than there are instances in the training data set. The second is its tendency to randomly select only one feature out of a set of highly correlated ones. The *Elastic Net* alleviates these issues by incorporating the l_1 and l_2 penalties in a weighted manner, as is shown in the following optimization problem:

$$arg \max_{\beta} \log PL(\beta) - \alpha \left(r \sum_{j=1}^{p} |\beta_j| + \frac{1-r}{2} \sum_{j=1}^{p} \beta_j^2 \right),$$

where $r\epsilon[0; 1]$ is the relative weight of the l_1 and l_2 penalty ($r = 1$ is a LASSO penalty, $r = 0$ is a ridge penalty). The Elastic Net penalty combines the LASSO's feature selection capability and Ridge's regularization power. As a result, it is more stable than LASSO, and in a situation of highly correlated features, it would select them all, while LASSO would randomly select only one. Usually, it is sufficient to give the l_2 penalty only a small weight to improve the stability of the LASSO, for example, $r = 0.9$ might suffice.

Similar to LASSO, the weight α inherently dictates the size of the chosen subset. Its optimal value is typically estimated in a data-driven way [37]. More on that in the practical part, where we will choose the best α for our data set.

To conclude, Ridge, LASSO, and Elastic Net offer powerful extensions of the Cox PH model, especially when dealing with high-dimensional datasets and highly correlated features. They help to regularize the Cox model and select only the most significant features, avoiding overfitting and assuring a more stable and interpretable model. Nevertheless, it is important to acknowledge their limitations. As linear models, they inherently capture only linear relationships, and non-linear relationships remain uncaptured. That leads us to the next section dedicated to the Random Survival Forests, a machine learning approach. Machine learning techniques are known for their ability to catch complex non-linear relationships, often outperforming traditional statistical methods in predictive accuracy.

3.5.4   Random Survival Forest

Random Survival Forest (RSF) is an ensemble machine learning method tailored for survival analysis. It is derived from the original Random Forest method developed by [28]. Random Forests have gained popularity in the machine learning community due to their effectiveness in classification and regression

machine learning tasks. Random Forest is built from multiple decision trees, averaging their predictions allows for more accurate predictions than any single tree can provide. Similar to Random Forest, Random Survival Forest leverages the power of multiple survival trees making its predictions more robust.

Random Survival Forest is comprised of *Survival Trees*. A survival tree is essentially a binary tree. It is developed by the iterative splitting of children nodes into two nodes until a certain criterion is met. An optimal node split is the one maximizing the survival difference between the children nodes. It is done by iterating through all features and finding such a value for a given feature that maximizes the survival difference [39].

Random Survival Forest training is comprised of the following steps:

1. Randomly draw L bootstrap samples of size n with replacement from the training dataset. n is about two-thirds of the original data. The remaining instances, about one-third, are termed out-of-bag (OOB) observations and are not included in the bootstrap sample.

2. For each sample, develop a full-grown survival tree based on the selected splitting criterion. At each node, randomly select $\sqrt{p}$ (or other number) covariates, where p is the total number of covariates. Stop developing when a certain condition is met (for example, when the terminal node has fewer observations than a predetermined threshold or the node reaches purity).

3. For each tree, calculate the cumulative hazard function with the Nelson-Aalen estimator. Find a mean of all trees to find ensemble CHF.

4. Use OOB data to determine the prediction error of the ensemble [38].

In summary, the Random Survival Forest is a robust and powerful approach to survival analysis that harnesses the collective power of multiple survival trees. Having explored several survival analysis methods, it is crucial to be able to evaluate them properly, since the standard machine learning performance metrics are not always applicable to survival analysis due to the presence of censoring. The next section will introduce essential criteria for the evaluation of survival analysis models and the primary performance metrics created for this purpose.

3.5.5 Performance Metrics

Survival prediction models play a vital role in healthcare. They are often used to estimate the risk of developing a particular disease and are crucial in guiding the clinical management of patients. It is, therefore, essential to assess their performance accurately. Similar to machine learning, this process of model evaluation is referred to as model validation. There are three aspects we can asses our model on:

1. **Overall performance,** which is the distance between the predicted and observed survival time.

2. **Discrimination,** or the model's ability to distinguish between high- and low-risk patients.

3. **Calibration** is the agreement between the observed and predicted survival times [34].

The absence of bias in a situation when the validation set contains censored instances is a sign of a good performance measure. Otherwise, in the presence of high levels of censoring, the evaluation would be unreasonably optimistic [34].

In this section, we will cover three measures of discrimination and one measure that assesses both discrimination and calibration (overall performance).

3.5.5.1 Harrel's and Uno's Concordance Indices

One way to measure discrimination is through *concordance*. A pair of patients is *concordant* if the subject who experiences an event earlier has a greater estimated risk [30]. *Measures of concordance* quantify the rank correlation between the predicted risk and the observed survival times. Typically, their values range between 0.5 and 1. A value of 0.5 indicates no discrimination, while 1 corresponds to the ideal discrimination [34].

The *concordance index*, or *C-index*, is the most widely used performance metric in survival analysis. It is defined through the concordance probability. The *concordance probability* is the probability that from the arbitrarily selected pair of patients (i, j), the one with a shorter survival time T, has the higher predicted risk M [34]. Mathematically, this is expressed as:

$$C = P(M_i > M_j | T_i < T_j).$$

Harrell's concordance index is the most widely used implementation of concordance index. To compute Harrel's concordance index C_H, we consider every comparable pair of patients where the one with the shorter time failed. Pair is "comparable" if we can determine which of them experienced the event first [30]. C_H is estimated as the proportion of these pairs in which the subject with the shorter survival time has a higher estimated risk. A modified version of this estimator, $C_H(\tau)$, only considers patients with $T_i < \tau$ and may provide more stable estimates [34].

Mathematically, Harrell's C-index is defined as a ratio between the number of concordant and comparable pairs:

$$\hat{C} = \frac{\sum_{i=1}^{N} \Delta_i \sum_{j=i+1}^{N} \left[I(T_i^{obs} < T_j^{obs}) + (1 - \Delta_j) I(T_i^{obs} = T_j^{obs}) \right] \left[I(M_i > M_j) + \frac{1}{2} I(M_i = M_j) \right]}{\sum_{i=1}^{N} \Delta_i \sum_{j=i+1}^{N} \left[I(T_i^{obs} < T_j^{obs}) + (1 - \Delta_j) I(T_i^{obs} = T_j^{obs}) \right]}. \quad (3.18)$$

where $I(\cdot)$ is the indicator function. It is equal to 1 if its argument is true, and 0 if it is not. T^{obs} is the observation time. Δ_i is a binary variable and $\Delta_i = 1$ if the subject i experienced an event during the time of observation and $\Delta_i = 0$ if they did not.

According to the Scikit-survival's documentation [35] and Rahman et. al. [34], Harrell's concordance index becomes biased in the presence of censoring. The bias increases with the level of censoring. Uno et. al. [31] introduced a modified concordance index, $C_U(\tau)$, which incorporates weights based on the probability of being censored. While their estimator proved robust to the choice of τ, they noted that the error of the estimate might be quite large if there are too few instances beyond this time point [21, 22].

Having explored the concordance index and its modifications, it is evident that it provides valuable insight into the model's discriminatory abilities. Yet, it offers a singular value that characterizes a model's performance across different time points without considering fluctuations in concordance that tend to happen over time. To enhance our evaluation of model performance in terms of discrimination across varying time points, we turn our attention to another popular metric in survival analysis: the ROC AUC. The subsequent section explores it in detail.

3.5.5.2 Time-dependent Area under the ROC

The *area under the receiver operating characteristic curve* (ROC AUC) is a popular performance measure for binary classification tasks. In survival analysis, it is used to determine how well estimated risk scores can distinguish diseased patients from healthy ones [35].

In binary classification, the *receiver operating characteristic (ROC)* is a curve that plots the *true positive rate (TPR or sensitivity)* against the *false positive rate (FPR)*. The FPR is the ratio of negative

instances that are falsely classified as positive. It is equal to 1− the *true negative rate* (the ratio of negative instances that are correctly classified, often referred to as *specificity*). TPR, or sensitivity, represents the ratio of positive instances classified as positive [9]

In survival analysis, we extend the ROC to continuous outcomes, where a patient is alive at the start of the observation, but might experience an event at some point later. Specificity and sensitivity, therefore, become time-dependent measures. It is specifically important, as model accuracy tends to be different at different points in time. Here we consider *cumulative cases* and *dynamic controls* at any given point in time t. *Cumulative cases* are all subjects who experienced an event prior to or at time t, while *dynamic controls* are those who are yet to experience the event after time t. By calculating the ROC AUC for any given time point t, we assess the model's ability to differentiate between patients. Specifically, how well the model can distinguish patients who fail by a given time $t_i < t$ from subjects who fail after this time $t_i > t$. The time-dependent ROC AUC is especially useful when we want to predict an event happening in a period up to time t, rather than at a specific time-point t [35].

While the time-dependent ROC AUC provides a valuable measure of a model's discrimination ability at various time points, it falls short in providing insight into the accuracy of individual predictions – calibration. To address this limitation and provide a more comprehensive model assessment, we turn our attention to the time-dependent Brier score.

3.5.5.3 Time-dependent Brier Score

Time-dependent ROC AUC and concordance index are great for assessing the overall discrimination among all time points (mean AUC and c-index) and the discrimination at any individual time point (the ROC graph), but they tell us nothing about the accuracy of individual predictions [35]. A metric analogous to regression performance measures used in machine learning would be ideal. Fortunately, such a metric exists. *Time-dependent Brier score* is a modification of mean squared error (MSE) that handles right censored data.

While the concordance index and time-dependent ROC AUC measure only discrimination, the time-dependent Brier score measures both discrimination and calibration, making it a metric of "overall performance". It is defined by the following equation:

$$BS^c(t) = \frac{1}{n} \sum_{i=1}^{n} I(y_i \leq t \wedge \delta_i = 1)\frac{(0 - \hat{\pi}(t|\mathbf{x}_i))^2}{\hat{G}(y_i)} + I(y_i > t)\frac{(1 - \hat{\pi}(t|\mathbf{x}_i))^2}{\hat{G}(t)}$$

where $\pi(\hat{t}|\mathbf{x})$ is a model's predicted probability of remaining event-free up to the time point t for feature vector $\mathbf{x}$, and $\frac{1}{\hat{G}(t)}$ is the inverse probability of censoring weight [35]. $I(\cdot)$ is the indicator function. y_i is the subject's survival time, δ_i is the event/censoring indicator.

The limitation of the time-dependent Brier Score is its applicability exclusively to the models that are capable of estimating the survival function. *Integrated Brier Score* provides a scalar value for general model evaluation. It is beneficial for model comparison, as time-dependent measures are a bit harder to compare than scalar values, and for the model fine-tuning process.

Survival analysis provides a suite of techniques for the prediction of a time to event. As we explored its methods and evaluation metrics, it is evident that the traditional SA provides robust tools for handling censored data, evaluating risk factors, and making predictions over time. However, as data grows in both complexity and volume, there is a growing need for more advanced modeling techniques, leading us to advanced machine learning: deep learning. Deep Learning, with its ability to handle large and complex datasets, promises to expand traditional survival analysis methods. In the following section, we will explore how deep learning can be adapted for survival analysis to harness the power of neural networks for more precise predictions.

3.6 Overview of Machine Learning Libraries and Tools

Chapter 4

Data Preparation and Analysis

In this chapter we are going to look into the dataset used in the study. Explore the most important features and their relationship with each other. Look into survival time...

The dataset consists of 993 806 of records for both transplanted patients and ones from the waiting list, and 450 features comprised of waiting list data and already transplanted patients for kidney and pancreas transplant from October 1, 1987 to the present. Kidney transplants account for 490 172 records.

4.1   Data Acquisition

The dataset provided by the IKEM (Institute of Clinical and Experimental Medicine in Prague) that we initially had was not suitable for any meaningful analysis. It had very few records, an insufficient amount of features, a lot of missing values, and quite outdated follow-ups. That is why it was decided to look for data elsewhere.

A number of institutions were contacted, including the Scientific Registry of Transplant Recipients (SRTR), the Australia and New Zealand Dialysis and Transplant Registry (ANZDATA), the National Health Service (NHS), the Center for Research in Transplantation and Translational Immunology (CR2TI), and other organizations from Spain, Germany and Canada. However, we faced the following obstacles:

- High cost of acquiring the data

- The need for association with local research group

- The need for local citizenship.

Fortunately, United Network for Organ Sharing (UNOS) agreed to provide their data for free upon the signing the data use agreement. The database consists of 1,108,884 records for all kinds of transplants. The table that contained data crucial for our analysis encompassed both kidney and pancreas transplants, had 993 806 records and 457 columns for both transplanted patients and ones from the waiting list. Data span from October 1, 1987 to the September, 2022. Kidney transplants account for 490 172 records.

4.2   Data Loading

The data were provided as a MongoDB database dump. Unfortunately, the database dump alone was insufficient for data analysis. It was necessary to run the database, import the database dump, and export data in the appropriate format. We set up the MongoDB database in a Docker container (Docker

explained in 6.2.2) running locally on our Mac, as we weren't allowed to install Docker in our university cluster. The corresponding table was then exported to CSV, compressed into a zip archive, and uploaded to the cluster.

The process of loading the data using the read_csv() method of the Pandas DataFrame was taking too long (about 5 minutes) due to the large size of the CSV file (80GB). The nature of data science is quite iterative, so we had to optimize this inefficiency. As a result, it was decided to store data in a Parquet file. The DataFrame.to_parquet() method was used to dump the pandas DataFrame into the Parquet database file. Parquet is an efficient cloud computing format that works on the principles of databases, making it possible to load data more efficiently. This significantly reduced the loading time of the entire dataset to just 38 seconds. Moreover, it allows for specifying the columns to load, which further reduces the data loading time to a mere 21 seconds. Therefore, using this technology has significantly improved the workflow.

4.3 Data Preprocessing Pipeline

In this section we will describe the data pipeline that is used to create the dataset out of the raw data. The pipeline can be found in github repository of this paper: survival_pipeline.py.

The work with the pipeline is pretty straightforward: we initialize the class and call the *load()* method. As is shown in the following block of python code:

```
1  from surv_data_pipeline.survival_pipline import
       ScikitSurvivalDataLoader
2
3  loader = ScikitSurvivalDataLoader()
4  X, y = loader.load()
```

Two main class constants are categorical_values and numerical_values, whose names speak for themselves. The primary class method is *load()*. This method loads the data into the pandas DataFrame, processes them, and returns *X* and *y*. *X* represents data transformed into numbers, while *y* is a tuple consisting of two elements: PSTATUS and PTIME. PSTATUS is the boolean censoring indicator (True - the event happened, False - otherwise), PTIME is the number of days survived. This format of target value is required by all Scikit-survival estimators.

The first step is to load the data into Pandas DataFrame from the parquet file. It is done by the Pandas method *read_parquet(path, engine, columns)*. In the *path* property, we need to specify the path to the parquet file; the *engine* parameter specifies the data loading scheme. We use *'auto'*, which tries PyArrow first; if that does not work, it uses FastParquet. PyArrow and FastParquet are just interfaces for reading data from Parquet files. In *columns*, we need to specify the columns we want to load.

Now, we need to filter out all records that might add noise to the data. Firstly, we remove patients who died of unrelated causes , such as accidents. Next, we filter out all recipients from the pediatric group, as the transplantation at that age differs significantly from that of adults. Then, we delete rows with all "Unknown" categories. Finally, we will include only living or deceased patients, depending on the donor type we are training a model for.

The next step is to handle missing (NaN) values. It is done with _handle_nan() method. We tried using mean and median imputation techniques from 3.3.3, but that only led to data leakage and suboptimal performance, so we just dropped all rows with empty columns. Fortunately, the size of the dataset allowed us to do that.
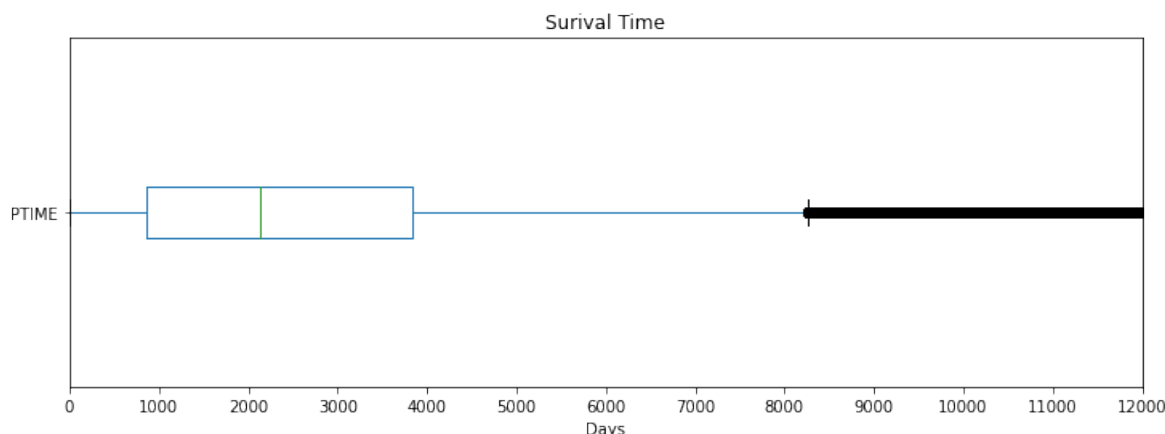
Figure 4.1: Box plot for the survival time

At this point, data were pickled and downloaded, and the next step was carried out locally, as we faced issues while trying to install the latest versions of packages on the cluster. The latest versions are needed to ensure a seamless run of models and transformer pipelines in a production environment.

After the NaN handling step, the training set is sent to the method *_get_X_y()* (or the same steps are done locally) where the numerical columns are scaled (with *StandardScaler()*) and categorical is one-hot encoded (with *OneHotEncoder()*) in the Scikit-learn transformer pipeline. The pipeline is pickled for later use in the application to transform the user input into data digestible by the model. Numerical and categorical values then comprise the *X* set. The target value set is constructed with the *Surv.from_arrays()* utility that accepts event and survival time and builds the y value acceptable to scikit-survival algorithms.

And now we have the dataset to work with.

4.4   Exploratory Data Analysis

In this section we are going to cover all the most significant features. The data were not adjusted to limit the influence of other factors, so the correlations we are trying to make here might not be fully correct, however some are confirmed in literature.

4.4.1   General Data

Here we will cover data for the whole DS with no division on living and deceased. We will see how some of these factors influence survival.

In the Table 4.1 you can see all features are used in this work

Survival Data

In this subsection, we will explore the y-axis that will be used for the survival estimator training. As mentioned earlier, the *y* value consists of boolean censoring status (PSTATUS) and time (PTIME), a numerical value representing the survival time or the time of censoring. The *y* value is called the *survival data*.

Please look at Figure 4.1, which shows the survival time box plot (PTIME column). The first quantile (Q_1) is equal to 867 days (2.4 years), the median is 2136 days (5.85 years), and the third quantile (Q_3) is equal to 3828 days (10.5 years). The interquartile range (*IQR*) is 2961 days, which forms the box. The

Table 4.1: Feature Table

Feature	Description	Type
ON_DIALYSIS	The recipient's pre-transplant dialysis status.	Categorical
PRE_TX_TXFUS	The recipient's history of pre-transplant bloo...	Categorical
GENDER	Recipient gender	Categorical
ETHCAT	Recipient ethnicity	Categorical
DIABETES_DON	The presence or absence of diabetes in the donor.	Categorical
DIAB	The presence or absence of diabetes in the rec...	Categorical
HCV_SEROSTATUS	The presence or absence of Hepatitis C in the ...	Categorical
PRE_TX_TXFUS	The recipient's history of pre-transplant bloo...	Categorical
GENDER	Recipient gender	Categorical
ON_DIALYSIS	The recipient's pre-transplant dialysis status.	Categorical
ABO_MAT	Donor recipient blood type match	Categorical
ETHCAT	Recipient ethnicity	Categorical
ETHCAT_DON	Donor ethnicity	Categorical
HBV_CORE	Whether or not the recipient has Hepatitis B. ...	Categorical
DIAB	The presence or absence of diabetes in the rec...	Categorical
HCV_SEROSTATUS	The presence or absence of Hepatitis C in the ...	Categorical
LIV_DON_TY	Donor type	Categorical
AGE	Recipient age at transplant	Numerical
BMI_CALC	Recipient BMI at transplant	Numerical
AGE_DON	Donor age at transplant	Numerical
CREAT_TRR	Recipient creatinine at transplant	Numerical
NPKID	Number of previous transplants	Numerical
COLD_ISCH_KI	The duration kidneys are preserved at low temp...	Numerical
KI_CREAT_PREOP	Living donor preoperative serum creatinine (mg...	Numerical
SERUM_CREAT	Recipient creatinine at discharge (mg/dL)	Numerical
NPKID	Number of previous transplants	Numerical
AGE	Recipient age at transplant	Numerical
HGT_CM_CALC	Recipient height in centimeters	Numerical
BMI_DON_CALC	Donor BMI	Numerical
AGE_DON	Donor age at transplant	Numerical

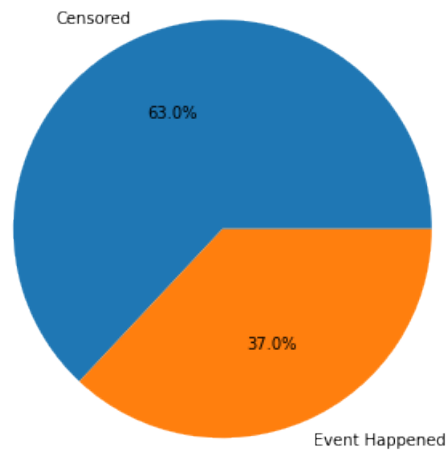The Distribution of Censored and Non-Censored Patients

Figure 4.2: Pie chart of censored/non-censored values.

left whisker extends from 0 days to the first quartile of 867 days. The right whisker is longer and extends from the third quartile to the $Q_3 + 1.5 * IQR$, which in our case is 8269,5 days. Any value above 8269,5 can be considered an outlier.

Look at the figure 4.2, where you can see the pie chart of the distribution of the PSTATUS column values. As can be seen, the percentage of censoring is 63%, which is pretty high.

Donor Type

In this subsection, we will explore the influence of donor type (living or diseased) on survival. It is a well-established fact that recipients who receive kidneys from living donors have higher life expectancy[29] [30]. Let us confirm that on the data.

Let us look at the Figure 4.3a, where are plotted two Kaplan-Meier survival curves for all patients from the dataset. On the graph, we can see that the survival probability of the living donor transplant is indeed significantly higher than the survival probability of the deceased.

This is the case because often there is no time to make full, in-depth HLA screening, allowing for some HLA mismatches. Additionally, deceased transplants may suffer from mild kidney damage due to the delay in transplantation. Living donor transplants are most often performed between blood relatives who share similar HLA, resulting in better compatibility.

Please, take a look at Figure 4.5a. There you can see the distribution of donor types. As can be seen, the deceased donor transplantation is much more prevalent.

In Figure 4.4 you can see the number of deceased/living transplantations by year. Even though the number of living transplantations grows each year, deceased transplantations still prevail.

Gender

In this subsection, we will explore the gender distribution in our dataset and its influence on survival. Let us explore the distribution of gender in our dataset. Please look at Figure 4.5b. As you can see, there are more males than females. Even though chronic kidney disease is more common in women, end-stage kidney failure and, therefore, kidney transplantation is more common in men[32].
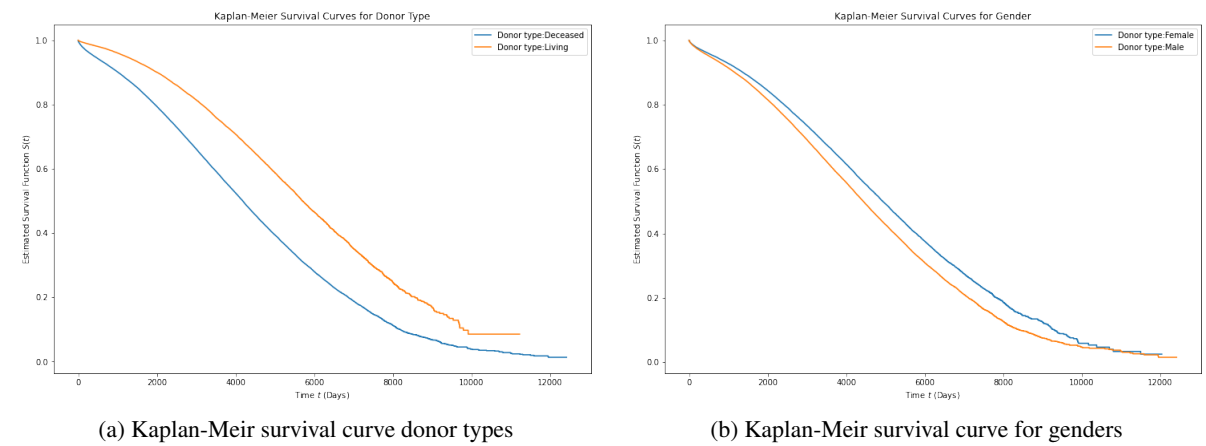
(a) Kaplan-Meir survival curve donor types   (b) Kaplan-Meir survival curve for genders

Figure 4.3: Survival curves for dialysis and gender
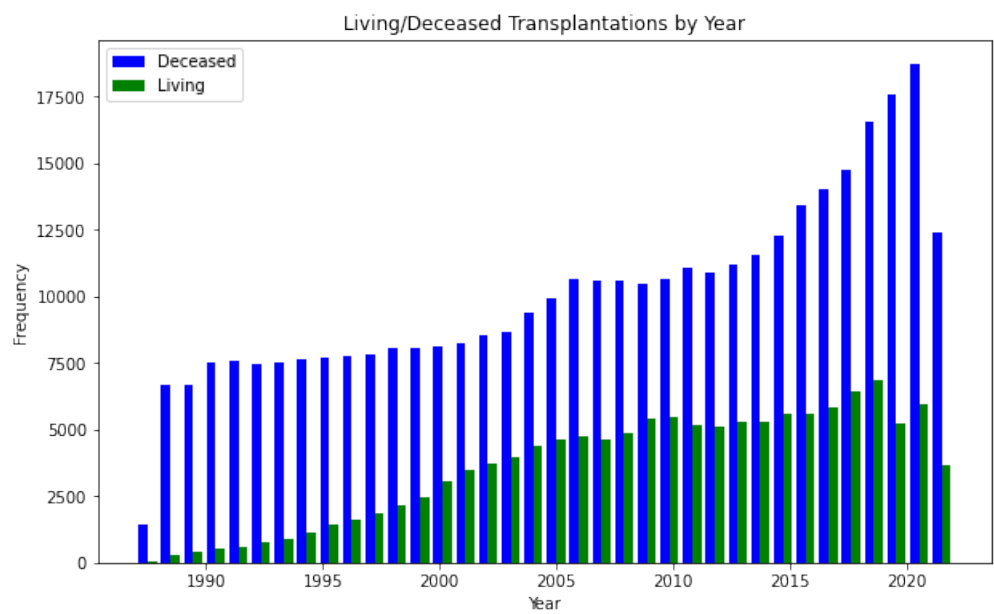
Figure 4.4: Histogram of Deceased/Living Transplantations by Year

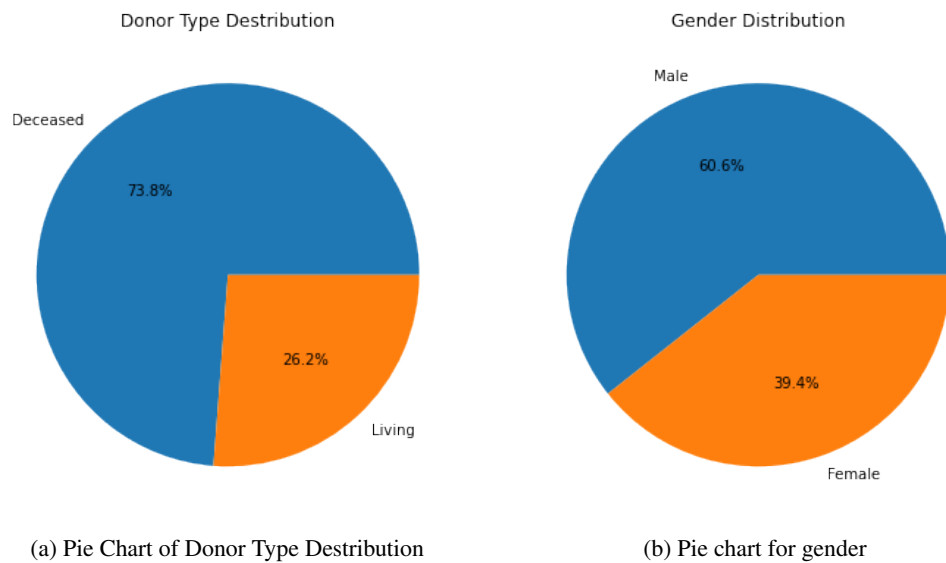(a) Pie Chart of Donor Type Destribution (b) Pie chart for gender

Figure 4.5: Pie charts for donor type and gender

Let us take a look at gender's influence on survival. In Figure 4.3b, we can see the Kaplan-Meier survival curves for men and women on the whole dataset. As can be seen from the graph, females generally have a lower risk than their male counterparts. Women usually live longer [14]. Quite a significant factor is the difference between male and female immune responses - males have a greater risk of getting an infection than females, and the intensity of the infection is higher [15]. Furthermore, the influence of immunosuppressants makes the problem of infection even worse.

The Use of Dialysis

In this subsection, we will explore the influence of dialysis on survival. In the Figure 4.6a we can see two survival curves for the patients who were on dialysis and who were not. As can be seen, the patients who were on dialysis before the transplantation have a greater risk than those who were not. It agrees with [33].

Ethnicity

This subsection explores the survival curves of different ethnic groups. In Figure [31] are plotted seven survival curves for various ethnicities. As can be seen in the image, five ethnicities share about the same survival probability, while two groups differ substantially from each other. White Americans had a higher survival probability than other ethnical groups, later converging to the others. It corresponds to ([31]) examining graft survival.
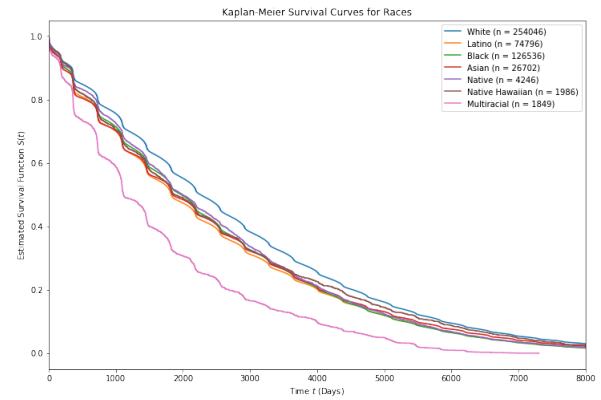
The lowest survival probability was in the multiracial group. Their number, however, was also the lowest - only 1849 instances, so it is not enough to make any conclusions. This group might be removed as there are very little instances.

4.4.2   Deceased Group

- a table for categorical frequencies at deceased YNU
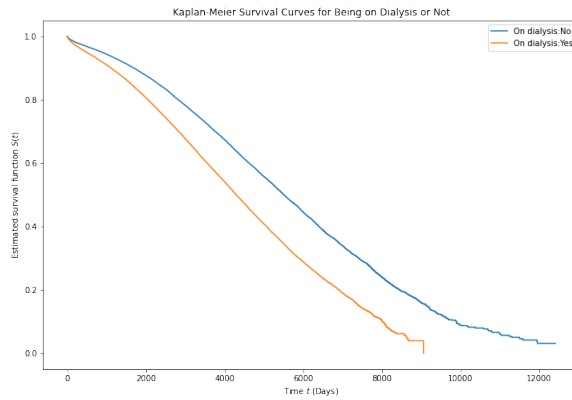
(a) Kaplan-Meir survival curve for using dialysis or not

(b) Kaplan-Meir survival curve for ethnicities

Figure 4.6: Survival Curves for dialysis and ethnicities

Table 4.2: Descriptive Statistics for Numerical Features for Deceased Donors

	count	mean	std	min	25%	50%	75%	max
AGE	117568.0	51.883259	13.112097	18.0	43.00	53.00	62.0	90.0
BMI_CALC	117568.0	27.770096	5.419293	15.0	23.80	27.30	31.3	72.2
AGE_DON	117568.0	38.101618	16.595043	0.0	24.00	40.00	51.0	88.0
CREAT_TRR	117568.0	8.251785	3.508183	0.1	5.70	7.82	10.3	28.2
NPKID	117568.0	0.122083	0.361227	0.0	0.00	0.00	0.0	5.0
COLD_ISCH_KI	117568.0	18.016163	8.981343	0.0	11.87	17.00	23.0	99.0

- ETHNCAT deceased table

- pie deceased: gender, hcv, diab - done

4.4.3 Living Group

- ETHNCAT living table

- a table for categorical frequencies at living YNU

- N/P/ND table

- Pie living: Diab, gender, liv_don_type, abo_mat

Table 4.3: Frequency of Yes/No Variables in Deceased Donor Group

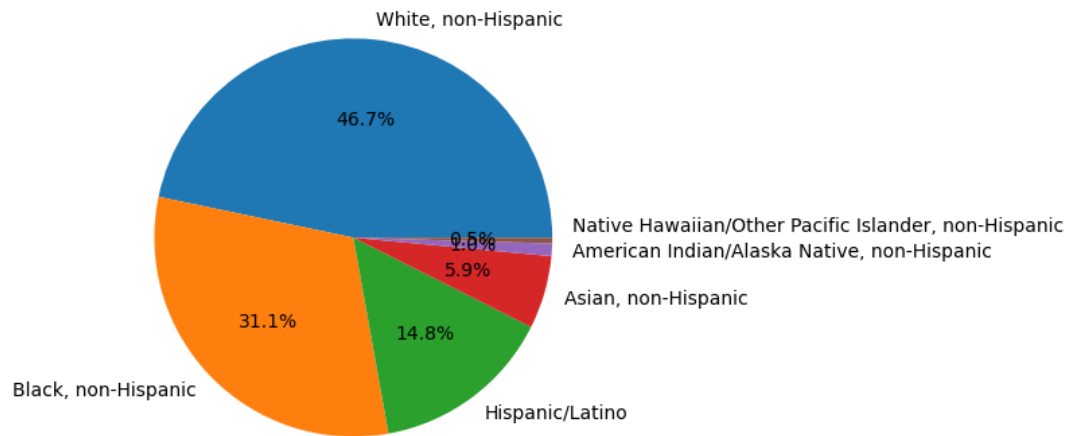|	Value	ON_DIALYSIS	PRE_TX_TXFUS	DIABETES_DON
N	Yes	25881	88215	109920
Y	No	91687	29353	7648

Figure 4.7: ETHCAT deceased pie chart
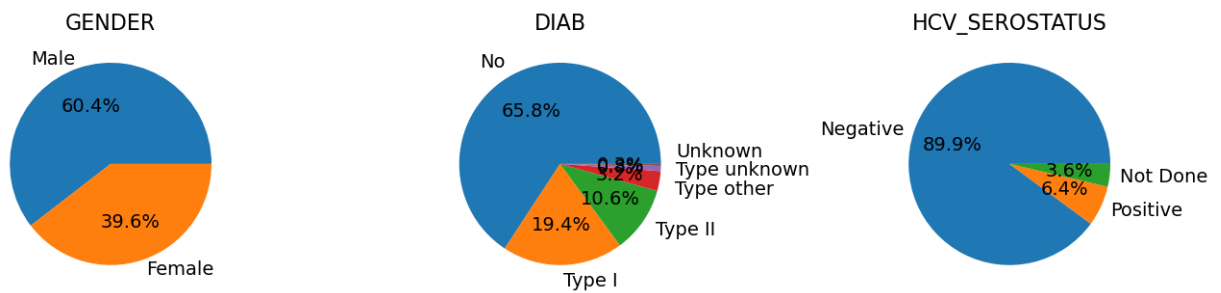

Figure 4.8: Feature Frequencies In ... in Deceased Sub group

Table 4.4: Descriptive Statistics for Numerical Variables in Living Donor Group

	count	mean	std	min	25%	50%	75%	max
KI_CREAT_PREOP	52420.0	0.87	0.38	0.10	0.70	0.8	1.00	25.0
SERUM_CREAT	52420.0	1.78	1.42	0.10	1.10	1.4	1.90	21.0
NPKID	52420.0	0.10	0.33	0.00	0.00	0.0	0.00	4.0
AGE	52420.0	47.46	13.78	18.00	37.00	48.0	58.00	85.0
HGT_CM_CALC	52420.0	171.11	10.67	109.00	162.60	170.2	178.00	213.4
BMI_DON_CALC	52420.0	26.90	4.37	15.01	23.71	26.6	29.71	71.9
AGE_DON	52420.0	41.34	11.55	15.00	32.00	41.0	50.00	77.0

Table 4.5: Frequency table for ETHCAT in Deceased Donor Group

Category	ETHCAT	ETHCAT_DON
White, non-Hispanic	34127	35778
Black, non-Hispanic	7664	7386
Hispanic/Latino	7565	6639
Asian, non-Hispanic	2340	1958
American Indian/Alaska Native, non-Hispanic	365	291
Native Hawaiian/Other Pacific Islander, non-Hi...	185	216
Multiracial, non-Hispanic	174	152

Table 4.6: Frequency of Yes/No/Unknown Variables in Living Donor Group

	Value	ON_DIALYSIS	PRE_TX_TXFUS
N	Yes	22488	42388
Y	No	29932	10032

Table 4.7: Frequency of N/P/ND Variables in Living Donor Group

	Value	HCV_SEROSTATUS	HBV_CORE
N	N	50237	43276
ND	P	801	6429
P	ND	1382	2715
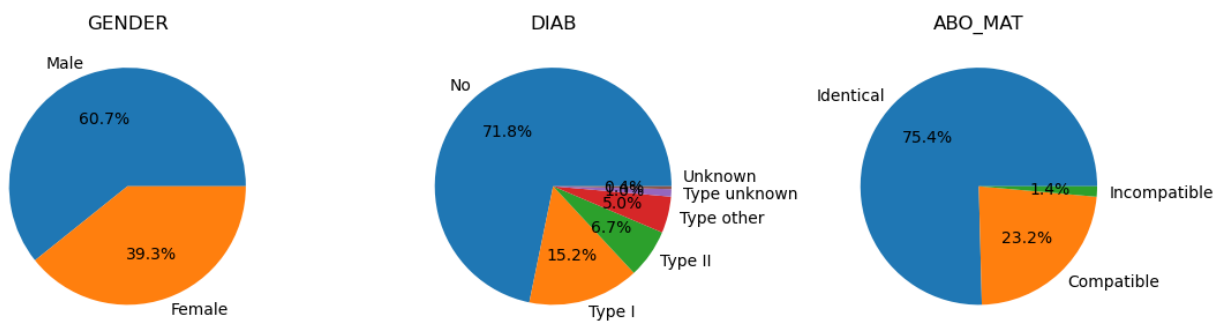
Figure 4.9: Pie Chart for Gender, Diab and ABO match in living subgroup
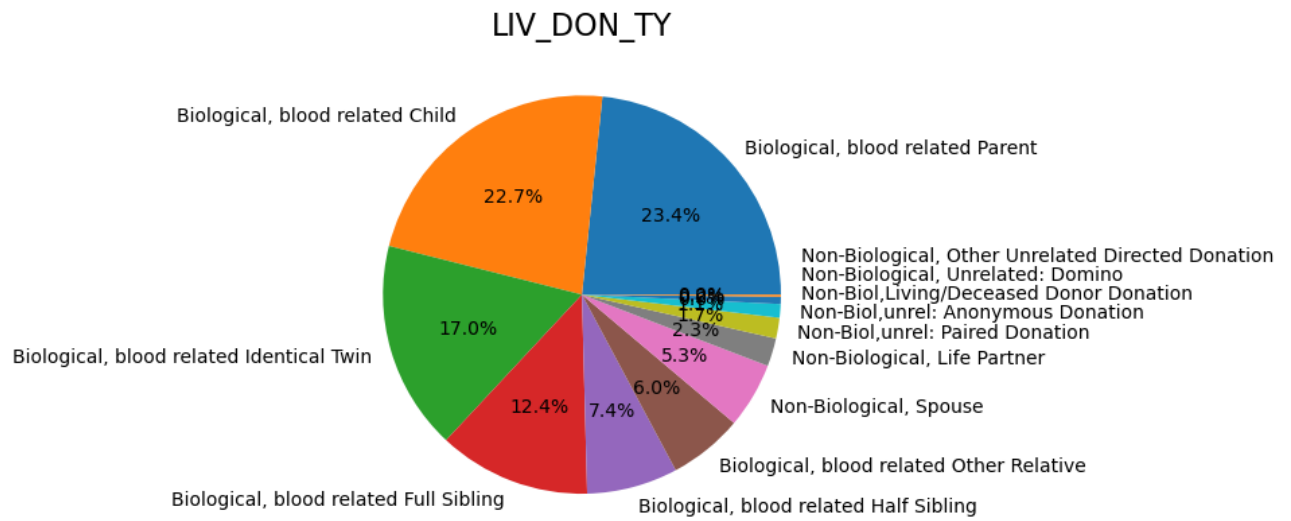
Figure 4.10: Pie Chart for living donor type

4.5   Dataset building, Exclusion criteria and noice reduction

4.6   Feature Engineering

- dialysis time - good

- differences in height, weight, bmi, and age provided only negative importance.

.

sd

Chapter 5

Machine Learning Model

5.1 Problem Formulation

Predicting the survival time after a successful kidney transplant can be approached in three ways: as a regression problem, classification problem, or through survival analysis.

A *regression* model may seem an intuitive choice, as we want to predict a numerical value – the survival time. But it is not the best option for the following reasons:

1. **The censored dataset**. The dataset has a high level of censoring – 76%. The dataset contains the number of days survived, along with the survival status. Including both living and deceased patients would introduce too much noise to the model, making it highly inaccurate. It is impossible to predict the number of days survived with regression methods based on a dataset comprised of both living and deceased patients.

2. **Censoring removal would produce bias and signifficantly reduce the dataset**. We could remove all censored instances, but that would reduce the dataset from 500 000 to roughly 120 000 examples. It would also introduce significant bias, as the dataset would contain only deceased patients, and most of them passed away before the introduction of modern techniques for treating the rejection. As a result, the model created from such a dataset would be highly inaccurate.

3. **Regression predicts only one single number.** It poses a problem, especially over extended time frames, as there are too many factors that we can't account for, leading to incorrect predictions.

Another way of formulating the problem is *classification*. We can theoretically divide the dataset into groups: "less than one year", "one to five years", "five and more", or even more groups and train a classifier based on them, as it was done by .... et al.. And again, we would face problems of censoring and bias mentioned above. So the classification is not the best option as well.

A more appropriate way of problem formulation is in terms of *survival analysis*. Survival analysis methods handle censoring and provide a better form of prediction: survival function or hazard function, which represents survival probability or the failure rate at each moment in time, respectively.

5.2 Model Selection and Training Approach

In our study, we utilized the algorithms from the scikit-survival library to train our models. When choosing the models that we would fully train and fine-tune in the end, we tried the majority of the models that scikit survival offers on a small subset of the dataset and chose the ones that perform best while taking adequate amounts of time. Despite SVM-based models demonstrating superior performance on the small subset of data, we decided against using them due to their inherent inability to handle large

datasets, which results in extended training times and frequent crashes when attempting to train on the slightly larger subset, not even mentioning the entire training set.

After rigorous experimentation, we ultimately selected Random Survival Forest (RSF) and Gradient Boosted Survival Analysis (GBSA) as our preferred models. These are ensemble models that have been widely recognized for their high accuracy and robustness despite prolonged training time. We also explored CoxnetSurvivalAnalysis (Cox Elastic Net, later referred to as coxnet), which demonstrated a notably low training time, which gave us the freedom to experiment with less time invested compared to the other models. Further research led us to discover an approach to training the Coxnet in the scikit survival documentation[25], which offered comparable performance to other models, requiring only a small fraction of the training time and computational resources. This showed that the Cox Elastic Net is more than a viable alternative to powerful and robust ensemble models, especially when time and computational resources are limited.

However, we encountered some challenges due to the relatively large dataset, even after all exclusion criteria and the removal of NaN values. RSF and GBSA demonstrated a long training time and high computational demands. The latter resulted in frequent kernel crashes in our Jupyter notebooks after long hours of training. To mitigate these issues, we decided to divide our dataset based on living and deceased donors and to reduce the amount of used features. The models were trained for each subset separately. To reduce the number of features, we trained models on smaller subsets of data and removed ones that showed negative or zero feature importance. This approach allowed us to account for features unique to a particular subset (KDPI for deceased and LIV_DON_TY for living) and to tailor our models to a specific type of transplantation. Sizes of the resulting datasets are as follows: 53,082 instances in living (42,465 in training), and 117,536 instances in deceased (94,028 training). By training separate models for each subset, we were able to develop potentially more accurate and effective models.

5.3 Results

In this study, six models were fine-tuned and trained, three for each subset. We utilized the Cox Elastic Net, Random Survival Forest, and GradientBoostedSurvivalAnalysis models from the scikit-survival library. Two models were trained locally (Coxnet and GBSA), and one was trained in a cluster (RSF). The preferred model identified in this study was Coxnet, due to its impressive performance and relatively short training time of 30 to 180 seconds, compared to other models that can take several hours to train.

It is important to note that the older version of scikit-survival was used on the cluster - 0.14.0, due to the limitation in the Python version on the cluster where these models were trained (3.6.8). Locally, the latest version at time 0.22.1 was used. This fact will later have some implications.

As numerical performance measures, the Uno concordance index (c-index), integrated Brier score (IBS), and the mean cumulative/dynamic area under curve (AUC) were chosen. The performance of trained models for living and deceased datasets can be observed in Tables 5.1 and 5.2 As can be seen, the deceased models perform worse than the living ones. Moreover, models have more or less the same performance for the same dataset, which might mean that models capture data well and the only bottleneck is the dataset. Consequently, the only potential way to improve the performance is to expand the dataset with further feature engineering. For more detailed information on the training of each model, please refer to the corresponding section from the following sections.

In living, the winner is coxnet, because despite the fact that it has the same discrimination ability (c-index and mean AUC) as RSF, the overall performance (IBS) is better. For deceased, the winner is GBSA: it slightly outperforms other models across all metrics.

As time-dependent metrics, ROC AUC and Brier Score were chosen. Please, look at Figure 5.1a, where we can see the plot of AUC over time for all living models. Even though the mean AUCs are

Model	Uno c-index	IBS	Mean AUC
Coxnet	0.723	0.136	0.743
Gradient Boosting	0.722	0.136	0.742
Random Survival Forest	0.723	0.139	0.744
Kaplan-Meier (for IBS reference)	-	0.247	-

Table 5.1: Model comparison for living dataset

Model	Uno c-index	IBS	Mean AUC
Coxnet	0.695	0.163	0.729
Gradient Boosting	0.699	0.162	0.734
Random Survival Forest	0.691	0.164	0.724
Kaplan-Meier (for IBS reference)	-	0.247	-

Table 5.2: Model comparison for deceased dataset

(a) AUC for all living models
(b) Time-dependent Brier Score for all living models
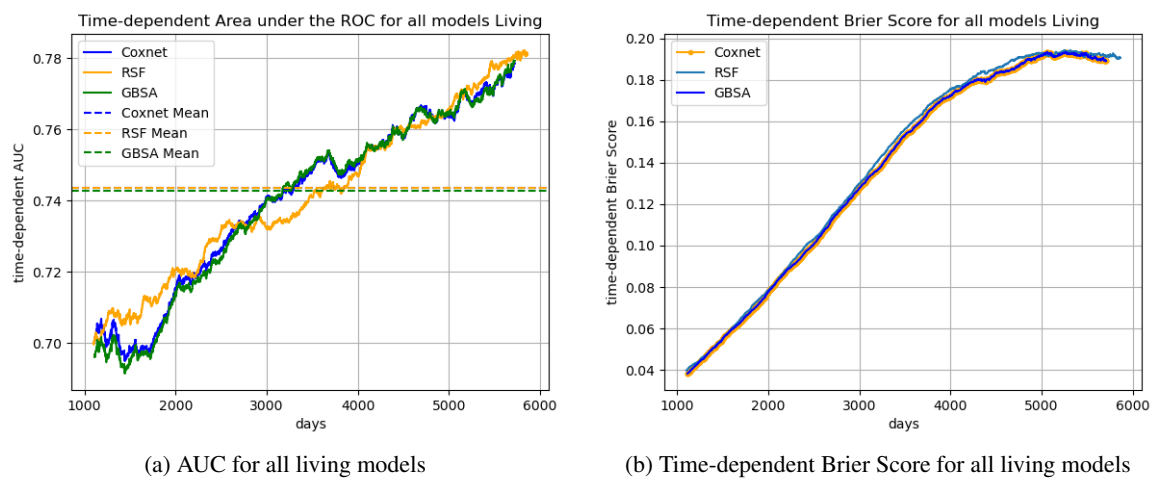
Figure 5.1: Time dependent Brier and AUC scores for all living models

(a) AUC deceased
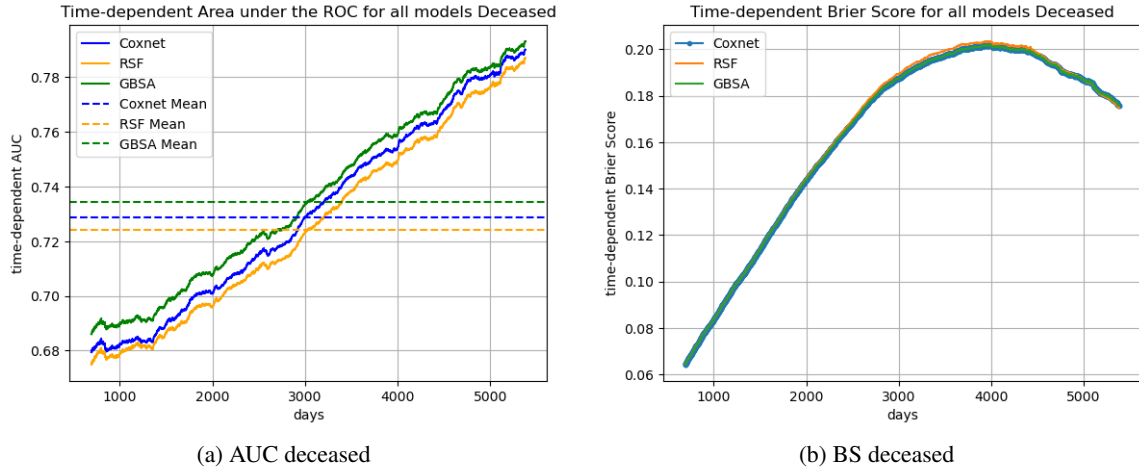
(b) BS deceased

Figure 5.2: Time dependent Brier and AUC scores for all deceased models

almost identical, values vary over time. From day 1000 to 2800 RSF performs better, then from day 2900 to day 4000 coxnet and GBSA overperform RSF. From day 4000 to 5000 all models perform similarly, except for a small rise of coxnet and GBSA, which later converges back with RSF. Day 5000 to 6000 RSF outperforms the two models again. RSF outperforms Coxnet and GBSA on 3/5 of the timeline, underperforms on 1/5, and is mostly similar on 1/5.

In Figure 5.1b we can see time-dependent Brier scores for all living models. Coxnet and GBSA here are identical, despite having different learners. We had to adjust the visualization settings for the other to be seen. The RSF curve is slightly higher than the two, meaning that overall performance is slightly worse, despite mostly superior discrimination in the previous graph.

In the Figure 5.2a we can see the AUC graph for the deceased models. GBSA shows superior discriminative ability, Coxnet is in the middle, and RSF shows the worst performance. Although all similarly rise over time.

Please, look at the Figure 5.2b, where the Brier Score is plotted for the deceased models. Similarly to living, the performance of Coxnet and GBSA is identical, but RSF is slightly worse, although by a small margin.

In conclusion, for the living, RSF shows slightly superior discrimination (AUC), but the overall prediction accuracy (BS) is inferior to Coxnet and GBSA. For the deceased, GBSA shows superior discrimination to Coxnet and RSF, but the prediction accuracy (BS) is the same for GBSA and Coxnet, and worse for RSF. Overall, Coxnet is preferred for both living and deceased due to its low training time. However, if training time is not a concern, GBSA may be preferred for the deceased cohort due to its slightly higher discriminative ability.

5.3.1 Coxnet

Coxnet, also known as a Cox elastic net, or Cox regularized regression, is a linear model known for its ability to handle large datasets efficiently. The performance is impressive as well.

As can be seen in the Table 5.3, the Coxnet performed the best for the living group, the worst for the deceased.

Dataset	Uno c-index	IBS	Mean AUC
Living	0.723	0.136	0.743
Deceased	0.695	0.163	0.729
Kaplan-Meier (for IBS reference)	-	0.247	-

Table 5.3: Coxnet performance on the test set

Hyperparameters   The coxnet has only two hyperparameters: L_1 ratio and α. The L_1 ratio defines the relative weight of the l_1 and l_2 penalty. We have found that the choice of L_1 ratio did not significantly impact the c-index, IBS, or mean AUC values. At both high (0.9) and low (<0.1) values of L1 ratio those values were about the same. However, a higher value of L_1 ratio (0.9) was selected for stability purposes, as suggested by the documentation of the scikit survival library[25].

To find the best α we used the approach from the documentation[25]. In short, it is calculated by fitting the model and choosing α that generalizes best from the resulting list *CoxnetSurvivalAnalysis._alphas*. As was covered in 3.4, the hyperparameter tuning is usually performed with either Grid-Search or RandomizedSearch. As there are not many alphas, we used GridSearchCV from scikit learn to find the best one.

Feature importance   The importance of more than 60 features was examined with the permutation importance method. Only a fraction of them had any importance. Those of negative and zero importance were removed, increasing model performance. Interestingly, HLA had little influence on long-term survival, so they were removed and the model performance improved. This is similar to what was noted by Terasaki in 2.3. The coefficients from the trained models, which represent feature importance, are visualized in Figures 5.3 and 5.4.

As we can see, the most predictive features are the patient's diabetes status, age, ethnicity, hepatitis C status, and dialysis.

Feature Engineering   We attempted to do feature engineering with donor-recipient height, weight, bmi and age differences inspired by [47]. The introduction of these features actually worsened the performance, so they were excluded. The engineered feature that actually improved the performance is the number of days patient was on dialysis (DIALYSIS_TIME), inspired by [48].

Data Pipelines   Data preprocessing is a critical step in machine learning, particularly for improving the accuracy and reliability of models. The two ways of data preprocessing are employed depending on the context: scikit-learn pipeline transformers, and the scikit-survival functions. The former is used for KidneyLife production, while the latter is used for training and analyzing the feature importances.

Scikit-learn transformers produce a numpy array, while scikit-survival functions output pandas data frame, which means that the column names are not preserved with the former. Moreover, the way of processing and the number of columns in outputs are different, so we could not just copy the column names from one to the other. As a result, the analysis of feature importances with transformers is virtually impossible.

The reason for employing scikit-learn pipeline transformers is because we needed a pipeline that produces consistent results in the production environment. Rather than developing a pipeline from scratch, the transformer pipeline used during model training is pickled and then used in production. However, this also means that if the model is to be used in production, it needs to be retrained using scikit-learn pipeline transformers. It is worth noting that the approach to preprocessing does not affect model performance.
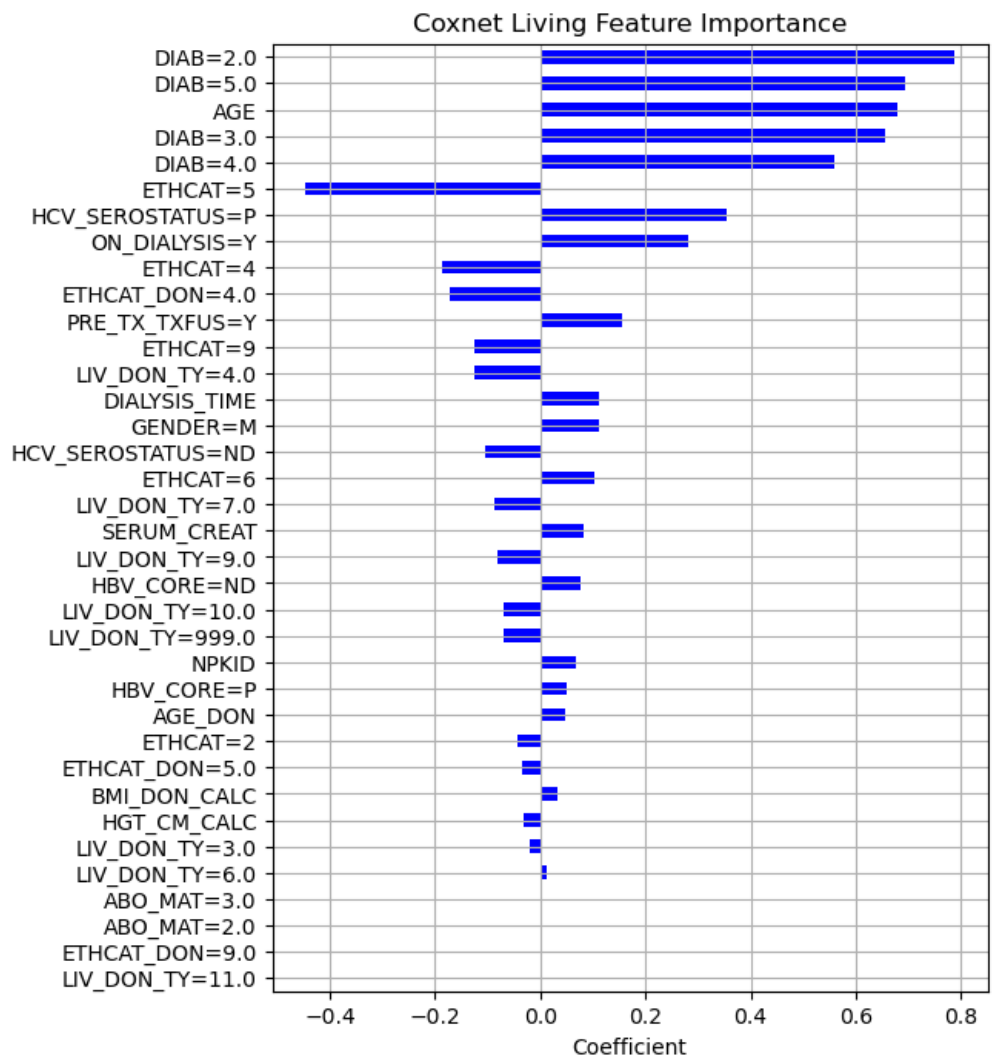
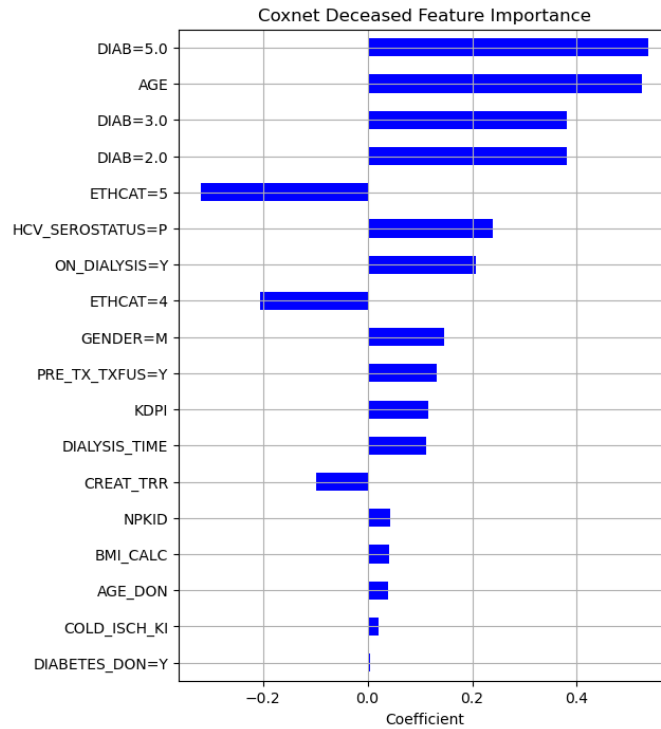Figure 5.3: Feature Coeficient Importance for Coxnet Living

Figure 5.4: Feature Coefficient Importance for Coxnet Deceased

Dataset	Uno c-index	IBS	Mean AUC
Living	0.723	0.139	0.744
Deceased	0.691	0.164	0.724
Kaplan-Meier (for IBS reference)	-	0.247	-

Table 5.4: RSF.

5.3.2 Random Survival Forest

Random survival forest is a powerful ensemble machine learning algorithm, that is comprised of multiple submodels, and therefore it takes a lot of time to train, making it a bit difficult to work with, especially during the hyperparameter tuning. It is extremely memory hungry during training and when making a prediction, making it unsuitable for deployment. Random Survival Forest is the main reason we divided train data into subsets of living and deceased donors, as the training on the whole dataset often lead to jupyter notebook kernel crashes due insuficient memory after many hours of training, despite the fact that the university cluster had 384 GB of RAM.

From the positive side, RSF is able to learn a lot from smaller amounts of data. While coxnet performs well on large datasets, it does worse than RSF on smaller ones. We noticed it during hyperparameter tuning: we used a subset of 1000 transplantation from the training dataset for that and RSF fitted them pretty well. We evaluated the predictions from the whole test set. So much so, that when we trained it on the whole training dataset there was very little difference in performance (C-index and IBS in both cases were almost identical, but tiny bit better (IBS the lower the better) with full dataset: 0.13906552(1000instances) vs. 0.139001 (full ds)). 1000 instances training time 5 minutes vs 66 minutes on full ds (61545)

Hyperparameter	Values Tried	Chosen Value
Number of Estimators	1, 5, 10, 30, 50, 70, 80	50
Max Depth	2,4, 6, 8, 10, 12, 14	12
Min Samples Split	2, 4, 6, 8 10, 12, 14, 16, 18	16

Table 5.5: RSF hyperparameters

Dataset	Uno c-index	IBS	Mean AUC
Living	0.722	0.135	0.743
Deceased	0.699	0.162	0.734
Kaplan-Meier (for IBS reference)	-	0.247	-

Table 5.6: SGBA results.

On the table 5.4 we can see the performance of RSF. Similarly to Coxnet it performs worse in deceased than for living. The performance itself is very similar to coxnet and GBSA.

The fine-tuning was performed with a custom script designed to imitate GridSearchCV but without cross-validation, similar to the one described in the previous subsection dedicated to the coxnet. The hyperparameters that were fine-tuned, values tried, and values chosen in the end can be seen on Table 5.5. Number of estimators was chosen as 50, as with higher values (80) we did not see any improvement, but the training time increased substantially (101 minutes). However, usually the higher the number of estimators, the better.

You can see the feature importances on Figures 5.5 and 5.6. The process of feature selection can be found at "FeatureSelection/rsf_living.ipynb" and "FeatureSelection/rsf_living.ipynb" (plot, but calculated in cluster file *deceased.ipynb*). For the living, the most important ones are, as before, recipient age, diabetes status, dialysis status and dialysis time, etc. For the deceased, situation is similar, but the recipient serum creatinine before transplant is a bit higher in the rating.

Living trained 65 minutes, deceased about 80 minutes . Both on the 50 estimators. Upping the estimators is a potential avenue to improvement, but one would need to be careful to not overfit the model. And that would take a lot of time: 200 estimators would require 4 hours of training and we might just overfit the model, or run out of RAM. Overfitting can be avoided with regularization, but we highly doubt that we would surpass the performance of coxnet survival.

5.3.3 Gradient Boosting Survival Analysis

CoxnetSurvivalAnalysis is a gradient-boosted Cox proportional hazard loss with regression trees as base learner. It is a relatively slow algorithm as it can only run on one CPU core and trains only one estimator at a time. So the training time might be quite large, as we cannot take advantage of cluster computing power. However that can also be an advantage, as we can train the model on PC locally. Another advantage is that the training time is predictable and the algorithm tells us during training how much time is left. Which is not perfectly precise, but it gives a good approximation.

Two other bits of information that the algorithm shares during training are loss and out-of-bag improvement (prediction accuracy improvement). The information is logged for the first ten estimators for each estimator, then for each decade, then for each hundred.

On Table 5.6, we can see the performance of two GBSA models for living and deceased subsets. As with two previous models, living model shows better performance than the deceased. For the deceased it also happens to be the best of them all with c_index 0.699 and IBS 0.162.
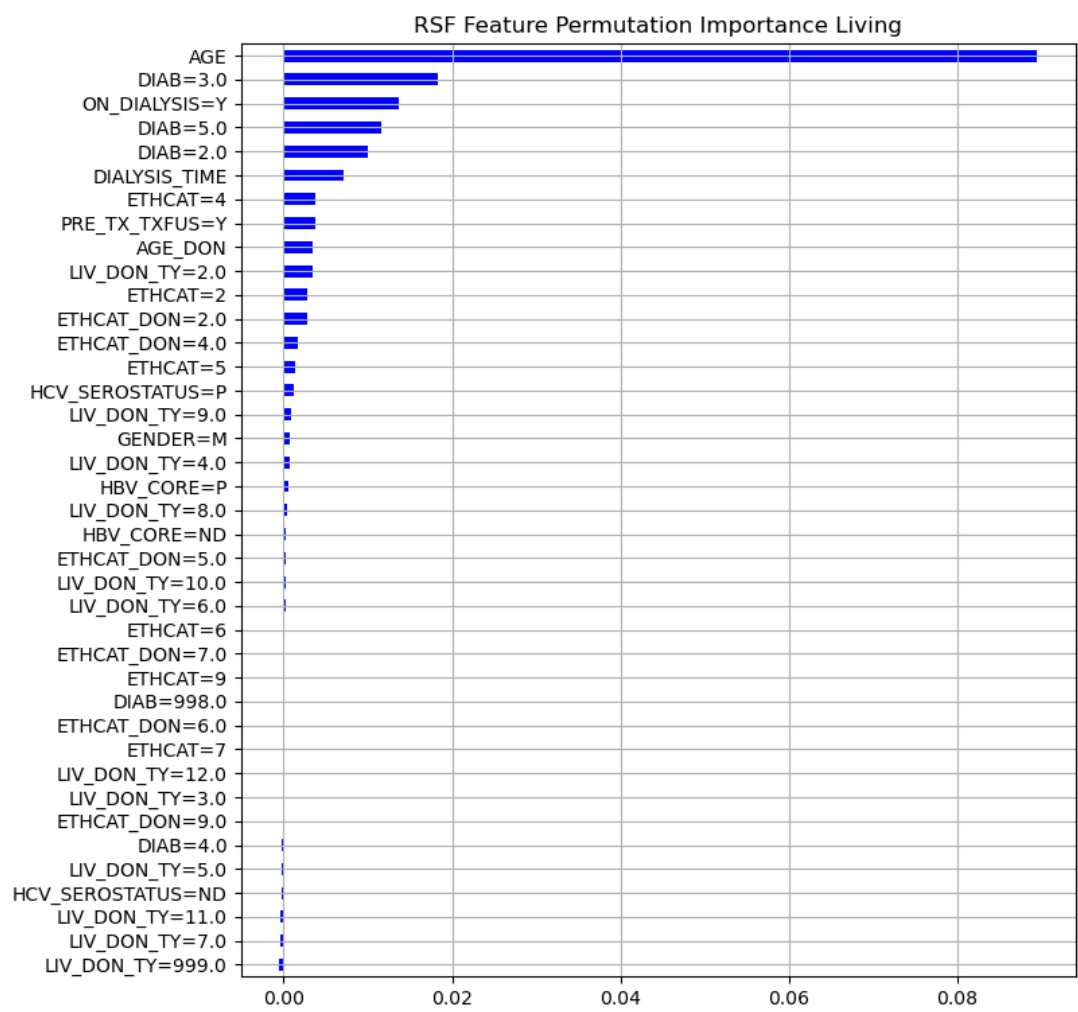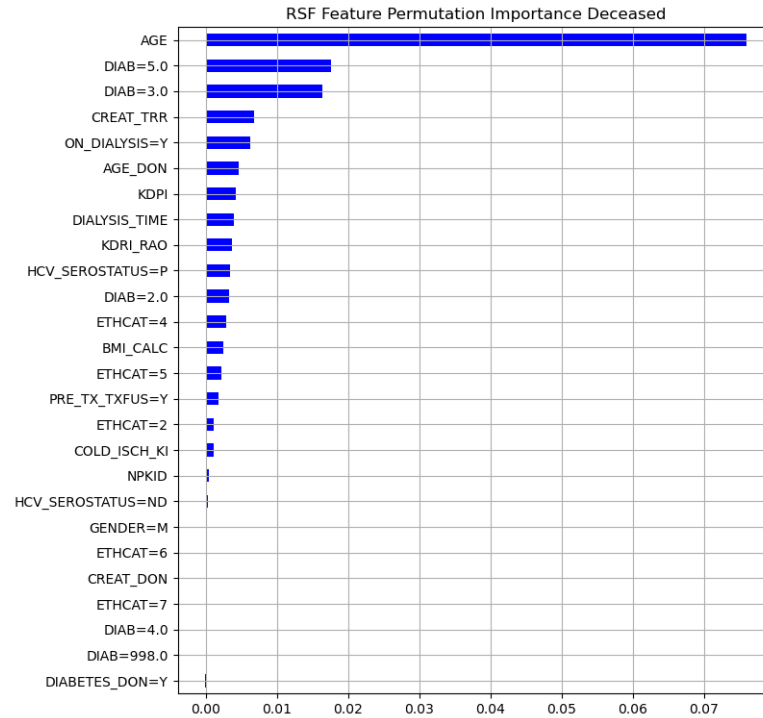
Figure 5.5: Feature Importance for RSF living

Figure 5.6: RSF feature importance deceased

We used three main hyperparameters: number of estimators, learning rate and max_depth. Grid search estimated the best max_depth to be 6, however experimentally we found that 4 gives better results. Learning rate we selected experimentally: and *0.2* produced the best results. With lower values (0.05, 0.1, 0.15) the model was underfit, with higher (0.5, 1) overfit. Number of estimators was estimated with early stopping monitor. It is a common approach for GB algorithms, and the code was borrowed from the scikit-survival documentation The monitor looks at the average improvement of the last 25 iterations and if it was negative for the last 50, it aborts training. For deceased the best number of estimators was 110 (trained 160) and 94 for living (trained 144). On Figures 5.7a and 5.7b you can see the plot of OOB improvement per learner. Red dashed line is a cutoff, after which the performance starts to decline.

Models were trained locally. The training took 75 minutes for living and 60 for deceased. With lower learning rate it was much longer (396 minutes for living on 0.05 learning rate).

Initially we performed feature selection on models with small amount of learners, and it produced an intresting result: only a small selection of features was deemed to have any importance. It did not seem right. As we already had several models trained on many more learners, we decided to assess feature importance on them. We discovered that many features that initially were deemed irrelevant, actually were relevant. Meaning that this approach to feature selection does not work for GBSA and feature importance must be assessed on fully trained model. For example, with RSF it does not matter, as feature importance is the same for underfit and overfit model.

On Figures 5.8 and 5.9 you can see feature importances estimated with permutation importance for living and deceased GBSA. For living, there are a couple of features with negative importance (PRE_TX_TXFUS, KI_KREAT_PREOP, HCV_SEROSTATUS and GENDER), but when removed, the performance drops from 0.722 to 0.720.
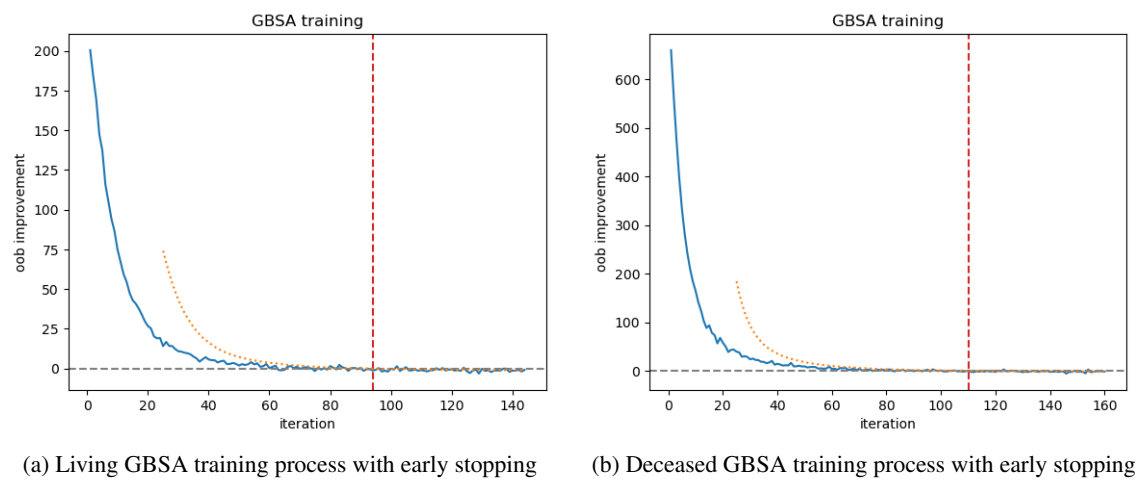
(a) Living GBSA training process with early stopping (b) Deceased GBSA training process with early stopping

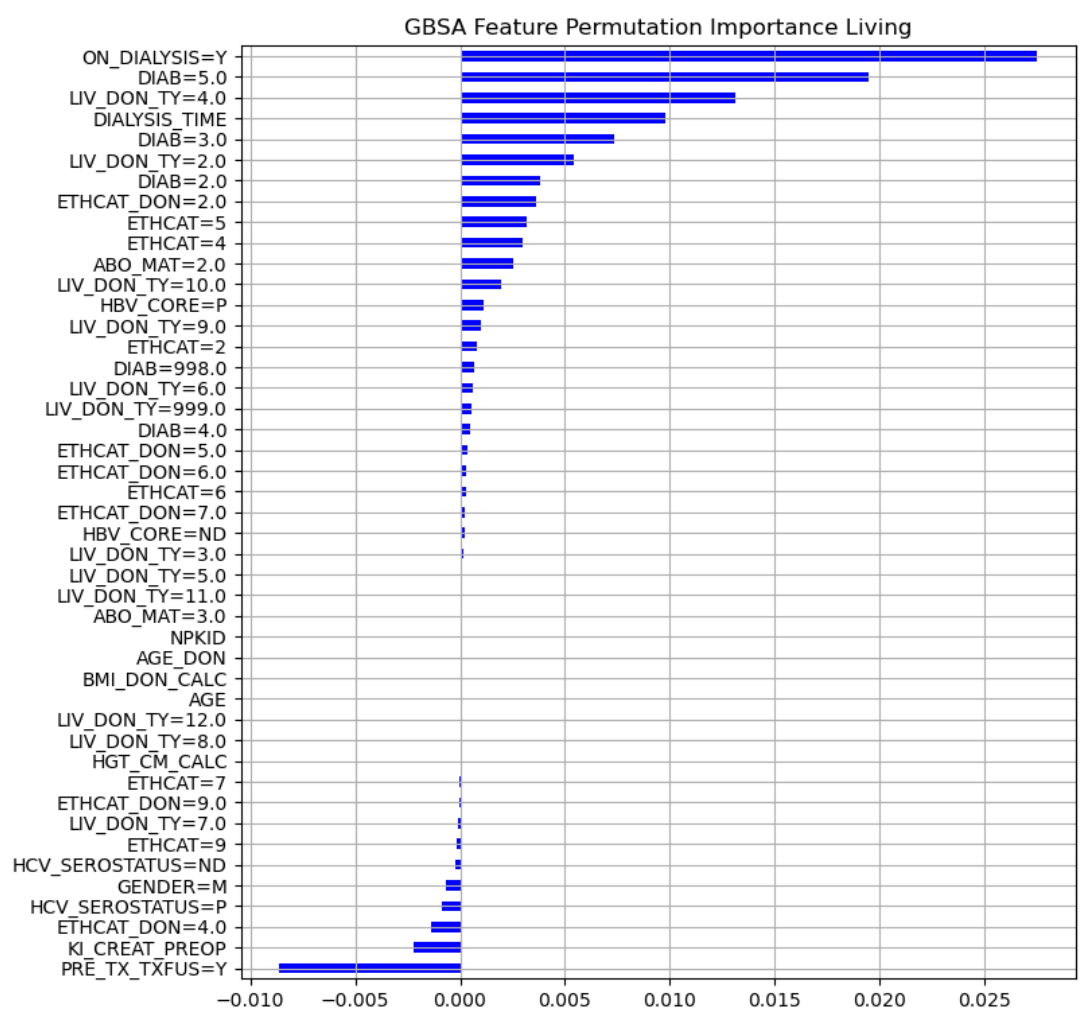Figure 5.7: GBSA training process with early stopping

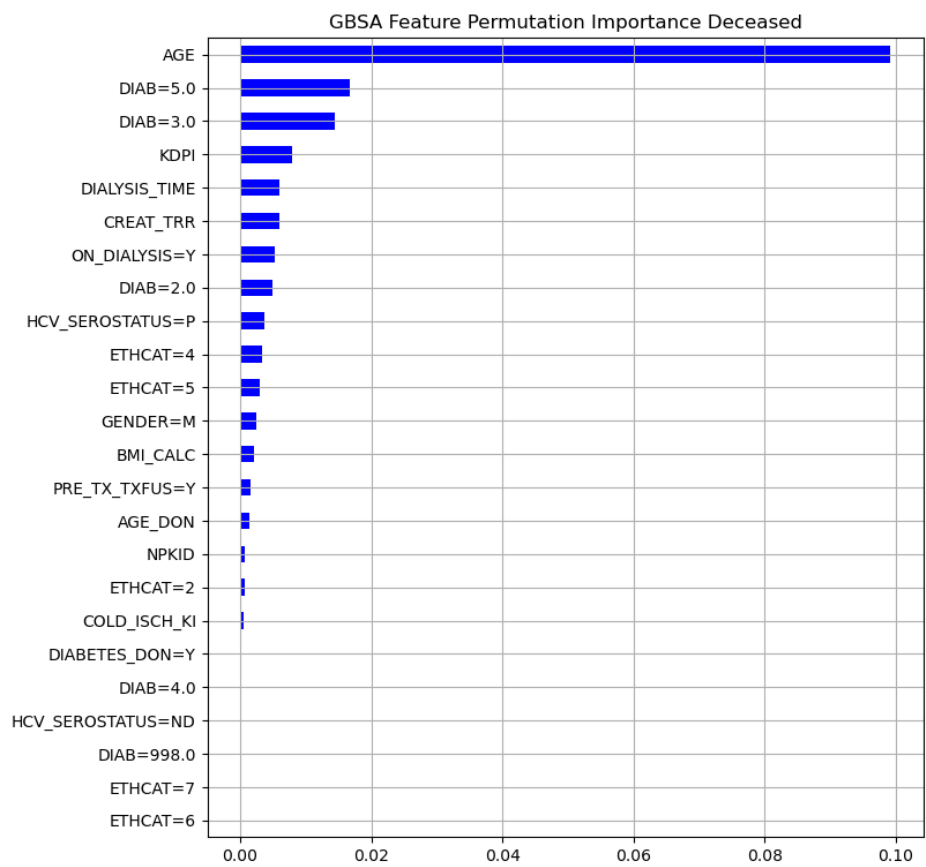Figure 5.8: GBSA living feature importance

Figure 5.9: GBSA deceased feature importance

5.4 Scoring algorithm

the cumulative hazard suits the place of transplantation score very well

5.5 Limitations

5.6 Further work

Chapter 6

Applications

The focus of this chapter is the software used in the realm of kidney transplantation. We will start by exploring k*idney pair donation* (KPD), with its intricacies and necessary terminology. Then, we will examine three solutions for KPD. Finally, we will introduce KidneyLife - a survival estimation application to complement KPD software to aid medical professionals in decision-making.

6.1   Existing Solutions

To the best of our knowledge, there is no survival estimation software for kidney transplantation as of yet. So let us review other adjacent solutions: kidney pair matching software. We will review three options: TX Matching, KidneyMatch, and KPDGUI. It is worth noting that there is limited information available online about alternatives. This shortage may stem from the fact that such software is developed for each transplantation program individually or from the preference of some developers to market their solutions directly rather than on the Internet. Nevertheless, some hospitals still rely on more traditional and manual matching techniques with no aid from specialized software.

When finding a compatible donor within the recipient's social circle is not feasible, the patient has an option to register in the *kidney paired donation* (KPD) program, also known as the *kidney exchange program* (KEP). KPD is an approach to living donor transplantation, where exchanges between pairs with incompatible donors are arranged in a way that allows each recipient to obtain a compatible kidney. While, theoretically, a large number of pairs can participate in the exchange, the higher the number of patients in the sequence of transplantations, the higher the risk of someone refusing to donate their kidney or being unable to due to a medical condition, which could compromise the entire sequence. So, the number of pairs in one sequence is usually limited to some number that makes sense for the center.

An example of how KPD works is as follows. Suppose A, B, and C are donor-recipient pairs with incompatible donors, and a recipient is compatible with the donor from the next pair. Donor A donates their kidney to recipient B; donor B donates theirs to recipient C; and donor C donates theirs to recipient A. Such a sequence of transplantations is called a loop. An illustration of that can be seen in Figure 6.1a.

In some cases, altruistic or non-directed donors (NDD) are available. These donors are willing to donate a kidney to anyone compatible without being bound to a specific recipient. In such instances, a sequence of transplantations called a chain is formed, progressing forward from one recipient to the next, culminating in the recipient from a regular waiting list, as can be seen in Figure 6.1b.

Another thing that such software does is it performs so-called virtual cross-match. It is a digital simulation of a test that predicts hyperacute rejection (briefly explained in 2.3). However, a virtual cross-match test is not a substitution for a regular one and must be complemented by a physical test, as occasional disparities may arise between the two.
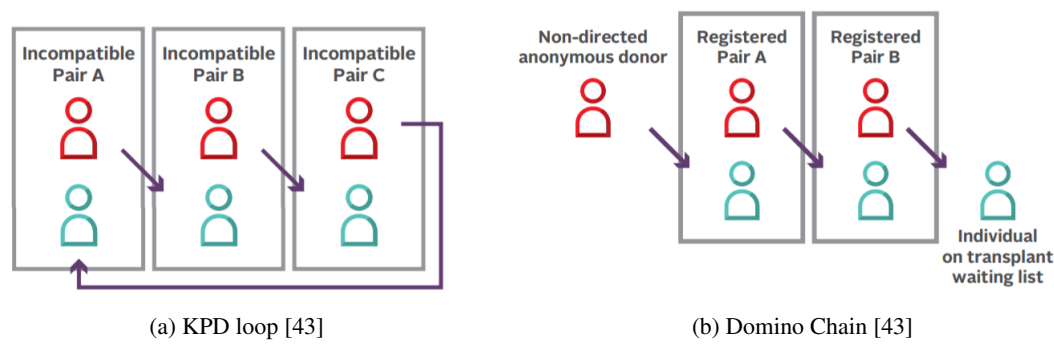
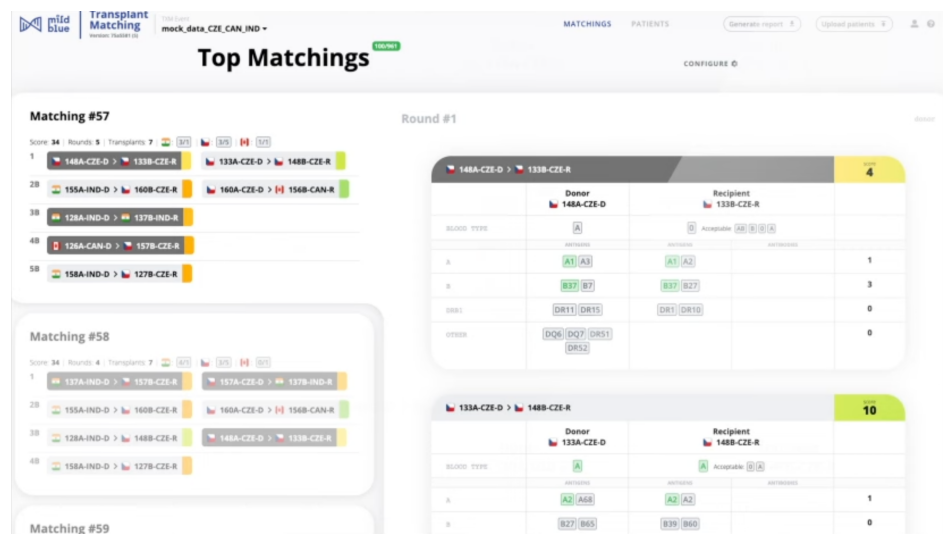(a) KPD loop [43] (b) Domino Chain [43]

Figure 6.1

Figure 6.2: Txmatching

6.1.1 TX Matching

TX Matching is open-source software that allows KPD centers to find optimal matches from a pool of incompatible donor-recipient pairs and altruistic donors. It utilizes one of two algorithms (solvers): integer linear programming (ILP) solver and All Solution Solver to find the best possible loops and chains.

One of its core features is multi-national support that allows for collaboration in the organ exchange between the Czech Republic, Austria, and Israel. Additionally, it offers customizable pools of patients, referred to as "TXM events", allowing for the segregation of patients based on many factors, including logistic feasibility. This feature ensures that matches are not only biologically compatible but also feasible according to other factors.

Furthermore, TX Matching has a wide range of configurable settings, allowing users to adjust parameters to their needs and requirements, enhancing the relevance of solved matchings. Matching scoring mechanism includes evaluation of blood group and HLA compatibility. Although, TX Matching is used primarily by the Czech Institute of Clinical and Experimental Medicine, efforts have been made to expand to other transplantation centers, but unfortunately, they were unsuccessful.
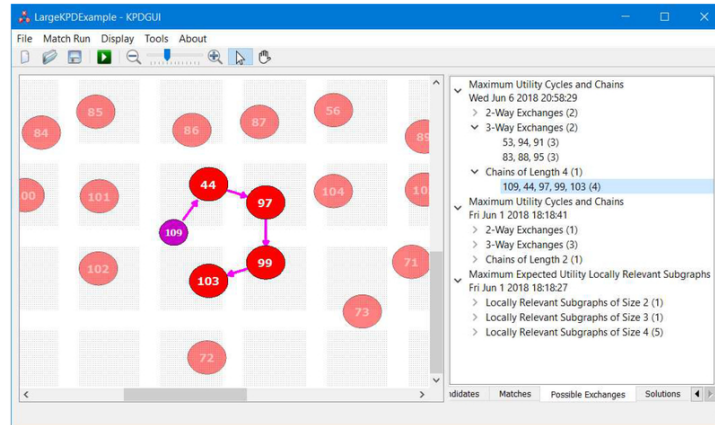
Figure 6.3: KPDGUI interface [44]. On the image you can see a chain.

6.1.2   KidneyMatch

KidneyMatch is the software developed and used by The Alliance for Pair Kidney Donation (APKD). They highly emphasize the security of their app: multifactor authentication, role-based authorization, and audit trails on every action in the software. KidneyMatch allows specifying the level of the match run: internal, regional, or national. The matches may be specified even further with various matching criteria, blood group, HLA, and discretionary exclusion criteria (DEC). They also allow specifying the expected solution: from a single match to chains and loops of different sizes.

To ensure the best possible matches, KidneyMatch collects more than 80 data points about each donor and more than 90 data points about each recipient. Such extensive data collection not only ensures good matching but also contributes to the refinement and development of matching methods. Furthermore, KidneyMatching supports additional medical data such as scans and charts, enhancing user experience and allowing for more well-informed decision-making. [45].

6.1.3   KPDGUI

KPDGUI (Kidney Paired Donation Graphical User Interface) is an interactive open-source software designed for the optimization and visualization of KPD programs. It addresses the aforementioned problem of donor withdrawal from KPD by arranging fallback options in case someone indeed left KPD and the exchange cannot proceed. What distinguishes it from the rest is that the application provides interactive visualization of the pool of incompatible pairs, non-directed donors, and the suggested loops and chains. [44].

6.2   KidneyLife

Our supervisors originally suggested creating a module for Txmatching using the developed models, as is reflected in the bachelor project task. However, due to differences in project philosophy and the kind of data used in Txmatching, it was decided to create a separate application. In the following section, we will present the application, its architecture, how it works, the process of its development, and what can be done further.
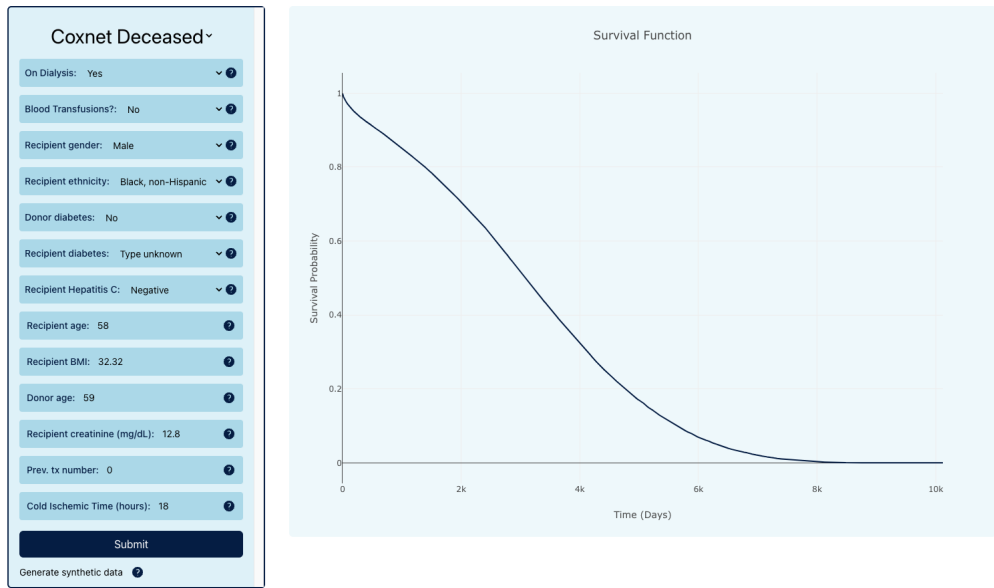
Figure 6.4: Kidney Life UI

6.2.1   Overview

We present KidneyLife, an inference application dedicated to the user-friendly use of machine learning models for survival analysis. The application aims to aid physicians in their decision-making process by providing a comprehensive estimate of a kidney transplant recipient's lifespan. It can be used in conjunction with the previously described KPD software.

Figure 6.4features KidneyLife's user interface. On the left, you can see the form for the input of the data. On its top, you can notice a dropdown menu for the model selection. Categorical features can be selected through dropdowns, while numerical data can be manually entered into text inputs. On the bottom of the form, you can see the Submit button, which sends the data to the server to make a prediction and performs checks for correct input and the absence of missing values. For more details on the validation and the front-end application, please refer to Section 6.2.2 under the "Front End" subsection. Further insights into the UI you can get at 6.2.4. Details regarding the prediction methods can be found in Section 6.2.2 under the "Back End" subsection.

Under the Submit button, you can find a button fetching synthetical data from the server. It allows for quick model illustration without the need for manual data entry. The details of the synthetic data generation can be found in Section 6.2.2 under the "Back End" subsection.

It is important to note that KidneyLife should be perceived as a minimum viable product (MVP), offering a minimum usable set of features for user feedback. This approach was adopted to address the rapidly growing development complexity and uncertainty of user needs and preferences. Section 6.2.5 outlines initial conceptualization of the application and the potential avenues for further development.

6.2.2   Architecture

The application was built with Docker Compose. The Docker Compose stack consists of backend, frontend, and reverse proxy services. Docker is a software platform that allows developers to package their applications in containers, a standard unit of software that bundles code and its dependencies to ensure fast and reliable execution across various computing environments [39]. Docker compose is

a utility for building and distributing multi-container applications. Each container within the Docker Compose application stack is called a service.

Front End   The front-end service contains an application developed with React, the popular JavaScript front-end framework developed by Facebook in 2013. The architecture of the front-end application is rather simple, as it consists only of a few key components. A component is an independent and reusable piece of code that takes the form of a JavaScript function that returns HTML. The main page component contains all other components and all logic related to the temporary data storage and sending requests.

The InputField component renders the input type text. Any characters apart from numbers and a dot are replaced with an empty string. It was done so, as the input type number provided inconsistent behavior: it worked as expected in browsers based on Chromium, such as Brave and Chrome, but allowed other characters in other browsers, such as Firefox and Safari. Moreover, the input type number allows the character "e", as it is a part of scientific notation. Furthermore, it contained unappealing arrows for increasing and decreasing an input number whose styles could not be altered.

The Select component renders the dropdown menu that allows one to select values from the provided list, which is particularly useful for categorical values. The ModalContainer serves as a wrapper and a background for the form. Finally, the PlotComponent renders the interactive plot from the Plotly.js package.

The front-end validation includes checks for the provided values. First and foremost, it checks if the value was indeed provided. It also includes min/max value validation, ensuring that the value falls into a specific range. Moreover, there is float/integer validation, depending on the data type the field requires. Finally, it verifies that the selected categorical value is indeed from the provided list of values, preventing users from adding their own category using the browser's built-in developer tools and sending the request to the server.

The styling was done with Tailwind, a CSS library that significantly simplifies the process of transforming the design into code compared to vanilla CSS, which can be cumbersome and time-consuming.

As a part of the front-end service build process, React code is compiled into production-ready optimized code, which is then served with the Nginx server.

Back End   The Django Rest Framework, a popular Python web framework for building RESTful APIs, was used for the back end. An application program interface (API) is a set of guidelines that dictate the procedures for connectivity and data exchange between applications or devices, while a REST API specifically adheres to the principles of the representational state transfer (REST) architectural style. Essentially, API serves as a mechanism for accessing a resource in one application or service from another. The application or service that is accessing the resource is called the client. The application or service providing the resource is referred to as a server. The resources accessed here are the machine learning models trained previously, which were serialized or "pickled" with Python package pickle and are deserialized when we need to make a prediction.

The back-end REST API consists of two endpoints: one for prediction and the other for synthetic data generation. The prediction endpoint expects the request body with two parameters: the model name and the data dictionary. It then validates the data, loads the model, passes data through the data pipeline to make the data usable by the model, loads the model, makes a prediction, and sends the results to the front end as a response.

The synthetic data generation endpoint has only one parameter: the model name. When a request is sent, it loads the proper model description JSON document and looks at its statistical data, particularly the 10th and 90th quantiles for numerical data and the frequencies for categorical data. Then, under the assumption of normal distribution, we generate data for each numerical feature and, under given value
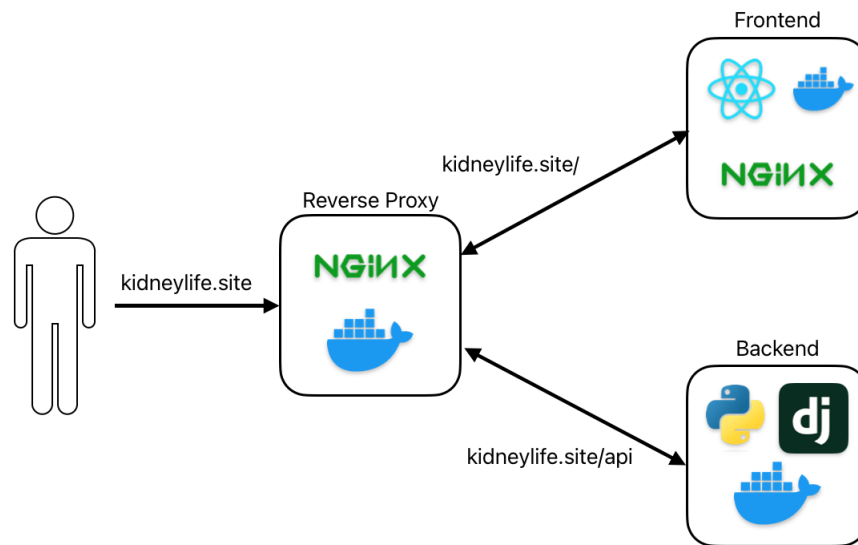
Figure 6.5: Application architecture.

frequencies, the values for categorical features. These quantiles are used to avoid peculiar situations when minimal values are generated along with the maximal (e.g. it could generate a pair of 6-year-old donor and 86-year-old recipient).

Reverse proxy   Even though the front-end code is hosted in the docker container on the server, it is executed in the user's browser. Cross-Origin Resource Sharing (CORS) is a mechanism that allows an application on one URL to request data from another URL. The browser implements the Same-origin policy as a part of its security measures, allowing requests from its own URL and blocking requests from other URLs unless certain conditions are met. When a browser sends a request to the server, it has an Origin property in its header. The server adds the header Access-Control-Allow-Origin to its response. If Origin and Access-Control-Allow-Origin are not the same, the browser will prevent the server from sharing the contents of the response with the client.

It is not possible, however, to identify or specify the URL of each user, nor is it secure to allow calling the request from all sources. To solve this problem, we use the reverse proxy. A *reverse proxy* is an intermediary server that directs the client request to the appropriate backend server [41]. When we access the applications's URL we get redirected to the front end. When we want to make a prediction, we send a request to the proxy, it redirects the request to the back end, and we get our prediction and render it on the front end. This schema is illustrated on the Figure 6.5.

In addition to the CORS handling, reverse proxy increases security, as it serves as a barrier between the internet and the backend. It also increases the application's scalability, as we can create multiple backends on different servers and balance the load between them.

6.2.3 Deployment

To deploy the app, we first need a place to deploy it to. We rent a virtual private server (VPS) or a virtual machine on the cloud. Initially, we used Google Cloud on the free trial, but it proved to be too

expensive, so we later migrated to another VPS hosting provider.

After securing the VPS, we bought the domain name "kidneylife.site" from the domain registrar Namecheap. Then, we registered the domain in the domain name system (DNS), associating the VPS IP address with the acquired domain.

The application was then deployed to the VPS using a Shell script. It goes through the following steps:

1. Check if the connection with the VPS can be established

2. Zip the application folder

3. Send the zip to the VPS

4. Unzip the folder

5. Build and run the docker compose stack with the production environment variables.

6.2.4 Functionality and UX

KidneyLife is a standard machine learning inference application. Inference application stands for an application whose sole purpose is user-friendly access to the machine learning model. As such, it is a one-page application that has select input with two models, the inputs are rendered based on the selected model. And the interactive graph made with Plotly.js. Finally, it features a button "Generate synthetic data" made for testing and illustration purposes.

The design of the application was crafted with consideration of several design principles. Shades of blue were chosen because of their calming effects and their association with reliability[42], which is particularly advantageous for doctors, as they will benefit from anything making their lives calmer. Moreover, blue is generally a safe color, as it appeals to a larger audience. Simplicity, being one of the main principles of modern web design, was naturally prioritized as inference applications inherently have little complexity. Furthermore, the larger margins and paddings were employed to provide breathing room within the interface, preventing it from feeling overcrowded. Lastly, the design exhibits responsiveness, accommodating a wide range of devices, from tablets to large 2k monitors, delivering a consistent and comfortable user experience across different screen sizes.

6.2.5 Future Work

There are several avenues for enhancing the application's functionality and user experience. First, integrating additional machine learning models is a promising avenue for providing more comprehensive and accurate predictions. While models mostly from scikit survival were used in this work, other packages, such as lifelines, can offer a wider range of survival analysis models.

Secondly, implementing a robust login system is essential to ensure secure access to the application's features and confidential data. It is especially important if we aim to implement the features outlined in the subsequent paragraphs, as they imply the storage of sensitive information. User authentication mechanisms, such as username-password authentication or OAuth, can restrict access to unauthorized individuals, safeguarding sensitive data associated with subsequent features.

Moreover, introducing patient (recipient and donor) and pair management systems can improve user experience by enabling the saving of the predictions for the provided pair of patients and models in the database. The feature will allow users to look at the predicted survival curve later only with a couple of clicks, eliminating the need to enter the data manually each time. The UI concept of this system can be seen in Figure 6.6.
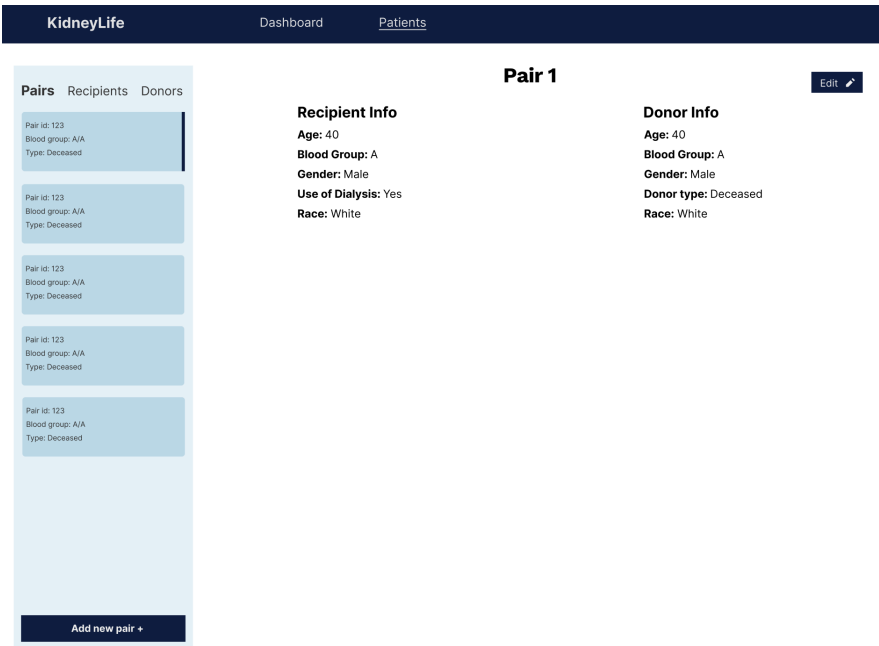
Figure 6.6: Patient and pair management system concept

Furthermore, integrating the FHIR HL7 standard would enhance interoperability and data exchange between the application and the hospital's ecosystem. FHIR, which stands for *Fast Healthcare Interoperability Resources* (FHIR) is a *Health Level Seven* (HL7) standard for electronic healthcare data exchange. Although only a few hospital systems currently support FHIR HL7, its adoption is viewed as the future of healthcare data exchange.

If any of these changes are to take place, we need to add continuous integration (CI) and continuous delivery (CD). CI/CD automates testing and deployment processes, improving efficiency and reducing the risk of errors. While the deployment has already been automated, the testing has not. Git Hub Actions can be utilized to test the application automatically on the server every time the changes are pushed to Git Hub, ensuring that pull requests cannot be merged into the main branch if the tests fail. While the tests for the backend already exist, automating them with GitHub actions would be the next step.

Cross-browser compatibility is another avenue for enhancing the application. It is crucial to ensure that the application looks the same across different operation systems and browsers. It can be achieved by designing custom scrollbars, improving the select fields by creating custom dropdown menus, and ensuring they look consistent across all browsers. However, it is possible that we may have missed something, so further testing is needed.

During the later stages of development, it was discovered that a nonprofit organization with the name KidneyLife already exists. If someone is interested in the application, we would need to rename and rebrand it to prevent any confusion and potential legal issues.
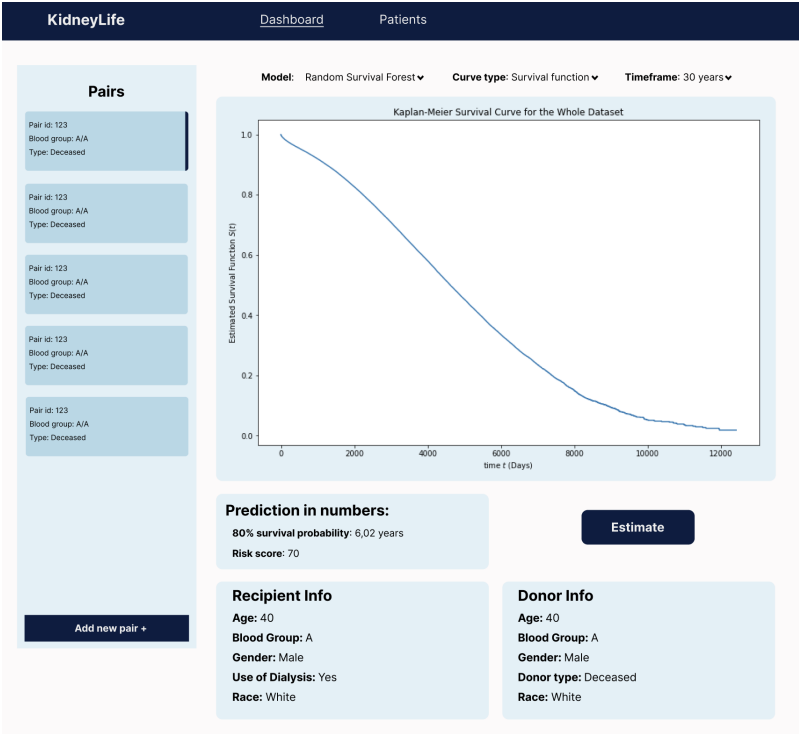
Figure 6.7: Kidney Life Dashboard page concept

Conclusion

Text of the conclusion...

Bibliography

[1] Knechtle, S. J., Marson, L. P., & Morris, P. (2019). Kidney transplantation - principles and practice: Expert consult - online and print (8th ed.). Elsevier - Health Sciences Division

[2] Nobel prize in physiology or medicine (2022) Our Scientists. Available at: https://www.rockefeller.edu/our-scientists/alexis-carrel/2565-nobel-prize/ (Accessed: February 6, 2023).

[3] Barker, C. F., & Markmann, J. F. (2013). Historical Overview of Transplantation. Cold Spring Harbor Perspectives in Medicine, 3(4). https://doi.org/10.1101/cshperspect.a014977

[4] Matevossian, Edouard, et al. "Surgeon Yurii Voronoy (1895-1961)-a pioneer in the history of clinical transplantation: in memoriam at the 75th anniversary of the first human kidney transplantation." Transplant International 22.12 (2009): 1132.

[5] PUNT, Jenni et al. Kuby immunology. Eight. vyd. New York: Macmillan Education, 2019. ISBN 9781319114701;1319114709;

[6] ABBAS, Abul K., Andrew H. LICHTMAN a Shiv PILLAI. Basic immunology: functions and disorders of the immune system. Sixth. vyd. Philadelphia: Elsevier, 2020. ISBN 9780323549431;0323549438;

[7] NCI Dictionary of Cancer terms (no date) National Cancer Institute. Available at: https://www.cancer.gov/publications/dictionaries/cancer-terms/def/abo-blood-group-system (Accessed: March 6, 2023).

[8] Dean L. Blood Groups and Red Cell Antigens [Internet]. Bethesda (MD): National Center for Biotechnology Information (US); 2005. Chapter 2, Blood group antigens are surface markers on the red blood cell membrane. Available from: https://www.ncbi.nlm.nih.gov/books/NBK2264/

[9] Aurélien Géron. Hands-on Machine Learning with Scikit-Learn and TensorFlow Concepts, Tools, and Techniques to Build Intelligent Systems. O'Reilly Media, Inc., Sept. 2019.

[10] Andriy Burkov. THE HUNDRED-PAGE MACHINE LEARNING BOOK. Andriy Burkov, 2019.

[11] Makary M A, Daniel M. Medical error—the third leading cause of death in the US BMJ 2016; 353 :i2139 doi:10.1136/bmj.i2139

[12] Bruce, P., Bruce, A., & Gedeck, P. (2020). Practical statistics for data scientists: 50+ Essential concepts using R and python (2nd ed.). O'Reilly Media. p. 141

[13] Kleinbaum, D. G., & Klein, M. (2011). Survival analysis: A self-learning text, third edition (3rd ed.). Springer.

[14] Ostan R, Monti D, Gueresi P, Bussolotto M, Franceschi C, Baggio G. Gender, aging and longevity in humans: an update of an intriguing/neglected scenario paving the way to a gender-specific medicine. Clin Sci (Lond). 2016 Oct 1;130(19):1711-25. doi: 10.1042/CS20160004. PMID: 27555614; PMCID: PMC4994139.

[15] vom Steeg LG, Klein SL. SeXX Matters in Infectious Disease Pathogenesis. PLoS Pathog. 2016 Feb 18;12(2):e1005374. doi: 10.1371/journal.ppat.1005374. PMID: 26891052; PMCID: PMC4759457.

[16] Rodrigues S, Escoli R, Eusébio C, Dias L, Almeida M, Martins LS, Pedroso S, Henriques AC, Cabrita A. A Survival Analysis of Living Donor Kidney Transplant. Transplant Proc. 2019 Jun;51(5):1575-1578. doi: 10.1016/j.transproceed.2019.01.047. Epub 2019 Jan 21. PMID: 31155195.

[17] Nemati E, Einollahi B, Lesan Pezeshki M, Porfarziani V, Fattahi MR. Does kidney transplantation with deceased or living donor affect graft survival? Nephrourol Mon. 2014 Jul 5;6(4):e12182. doi: 10.5812/numonthly.12182. PMID: 25695017; PMCID: PMC4317718.

[18] Pisavadia B, Arshad A, Chappelow I, Nightingale P, Anderson B, Nath J, Sharif A. Ethnicity matching and outcomes after kidney transplantation in the United Kingdom. PLoS One. 2018 Apr 13;13(4):e0195038. doi: 10.1371/journal.pone.0195038. PMID: 29652887; PMCID: PMC5898720.

[19] Guillermo García García, Arpana Iyengar, François Kaze, Ciara Kierans, Cesar Padilla-Altamira, Valerie A. Luyckx, Sex and gender differences in chronic kidney disease and access to care around the globe, Seminars in Nephrology, Volume 42, Issue 2, 2022, Pages 101-113, ISSN 0270-9295, https://doi.org/10.1016/j.semnephrol.2022.04.001. (https://www.sciencedirect.com/science/article/pii/S0270929522000092)

[20] Mange KC, Joffe MM, Feldman HI. Effect of the use or nonuse of long-term dialysis on the subsequent survival of renal transplants from living donors. N Engl J Med. 2001 Mar 8;344(10):726-31. doi: 10.1056/NEJM200103083441004. PMID: 11236776.

[21] Rahman MS, Ambler G, Choodari-Oskooei B, Omar RZ. Review and evaluation of performance measures for survival prediction models in external validation settings. BMC Med Res Methodol. 2017 Apr 18;17(1):60. doi: 10.1186/s12874-017-0336-2. PMID: 28420338; PMCID: PMC5395888.

[22] Evaluating survival models — scikit-survival 0.21.0. (n.d.). Readthedocs.Io. Retrieved July 18, 2023, from https://scikit-survival.readthedocs.io/en/stable/user_guide/evaluating-survival-models.html

[23] Ping Wang, Yan Li, and Chandan k. Reddy. 2019. Machine Learning for Survival Analysis: A Survey. ACM Comput. Surv. 51, 6, Article 110 (February 2019), 36 pages.https://doi.org/10.1145/3214306

[24] Definitions:, 1. 1. (n.d.). Survival distributions, hazard functions, cumulative hazards. Stanford.edu. Retrieved October 23, 2023, from https://web.stanford.edu/~lutian/coursepdf/unit1.pdf

[25] Penalized Cox Models — scikit-survival 0.22.1. (2015). Readthedocs.io. https://scikit-survival.readthedocs.io/en/stable/user_guide/coxnet.html

[26] Ishwaran, H., Kogalur, U. B., Blackstone, E. H., & Lauer, M. S. (2008). Random survival forests.

[27] Wang, H., & Li, G. (2017). A selective review on random survival forests for high dimensional data. Quantitative bio-science, 36(2), 85.

[28] Breiman, L. (2001). Random forests. Machine learning, 45, 5-32.

[29] Gönen M, Heller G. Concordance probability and discriminatory power in proportional hazards regression. Biometrika 2005;92(4):1799–09.

[30] Enrico Longato, Vettoretti, M., & Barbara Di Camillo. (2020). A practical perspective on the concordance index for the evaluation and selection of prognostic time-to-event models. Journal of Biomedical Informatics, 108, 103496–103496. https://doi.org/10.1016/j.jbi.2020.103496

[31] Uno H, Cai T, Pencina MJ, D'Agostino RB, Wei LJ. On the C-statistics for evaluating overall adequacy of risk prediction procedures with censored survival data. Stat Med. 2011 May 10;30(10):1105-17. doi: 10.1002/sim.4154. Epub 2011 Jan 13. PMID: 21484848; PMCID: PMC3079915.

[32] Kindt TJ Goldsby RA Osborne BA Kuby J. Kuby Immunology. 6th ed. New York: W.H. Freeman; 2007.

[33] Health. (2022). Kidneys. Vic.gov.au. https://www.betterhealth.vic.gov.au/health/conditionsandtreatments/kidney

[34] Facts About Chronic Kidney Disease. (2020, May 15). National Kidney Foundation; https://www.kidney.org/atoz/content/about-chronic-kidney-disease

[35] Kamyar Kalantar-Zadeh, Jafar, T. H., Nitsch, D., Neuen, B. L., & Perkovic, V. (2021). Chronic kidney disease. The Lancet, 398(10302), 786–802. https://doi.org/10.1016/s0140-6736(21)00519-5

[36] Holland, K. (2019, May 23). Everything You Need to Know About Kidney Failure. Healthline; Healthline Media. https://www.healthline.com/health/kidney-failure#outlook

[37] Causes of chronic kidney disease. (2022, August 29). National Institute of Diabetes and Digestive and Kidney Diseases; NIDDK - National Institute of Diabetes and Digestive and Kidney Diseases. https://www.niddk.nih.gov/health-information/kidney-disease/chronic-kidney-disease-ckd/causes

[38] Estimated Glomerular Filtration Rate (eGFR). (2015, December 24). National Kidney Foundation; https://www.kidney.org/atoz/content/gfr

[39] What is a Container? | Docker. (2023, October 26). Docker. https://www.docker.com/resources/what-container/

[40] Use Docker Compose. (2024). Docker Documentation. https://docs.docker.com/get-started/08_using_compose/

[41] What is a Reverse Proxy Server? | NGINX. (2023, June). NGINX. https://www.nginx.com/resources/glossary/reverse-proxy-server/

[42] Kendra Cherry, Mse. (2022, November 22). How the color blue impacts moods, feelings, and behaviors. Verywell Mind. https://www.verywellmind.com/the-color-psychology-of-blue-2795815

[43] Kidney Paired Donation (KPD) Program. (2024, January 3). Professional Education. https://professionaleducation.blood.ca/en/organs-and-tissues/programs/kidney-paired-donation-kpd-program

[44] Bray, M., Wang, W., Rees, M. A., Peter X-K. Song, Leichtman, A. B., Ashby, V. B., & Kalbfleisch, J. D. (2019). KPDGUI: An interactive application for optimization and management of a virtual kidney paired donation program. Computers in Biology and Medicine, 108, 345–353. https://doi.org/10.1016/j.compbiomed.2019.03.013

[45] 4 Key Features of the APKD Software Program | Alliance for Paired Kidney Donation. (2022, December 9). Alliance for Paired Kidney Donation |. https://paireddonation.org/4-key-features-of-the-apkd-software-program/

[46] Aufhauser, D. D., Peng, A. W., Murken, D. R., Concors, S. J., Abt, P. L., Sawinski, D., Bloom, R. D., Reese, P. P., & Levine, M. H. (2018). Impact of prolonged dialysis prior to renal transplantation. Clinical Transplantation, 32(6). https://doi.org/10.1111/ctr.13260

[47] Naqvi SAA, Tennankore K, Vinson A, Roy PC, Abidi SSR. Predicting Kidney Graft Survival Using Machine Learning Methods: Prediction Model Development and Feature Significance Analysis Study. J Med Internet Res. 2021 Aug 27;23(8):e26843. doi: 10.2196/26843. PMID: 34448704; PMCID: PMC8433864.

[48] Aufhauser DD Jr, Peng AW, Murken DR, Concors SJ, Abt PL, Sawinski D, Bloom RD, Reese PP, Levine MH. Impact of prolonged dialysis prior to renal transplantation. Clin Transplant. 2018 Jun;32(6):e13260. doi: 10.1111/ctr.13260. Epub 2018 Jun 25. PMID: 29656398; PMCID: PMC6023748.