BHT Berliner Hochschule für Technik

Kidney Graft Loss Prediction leveraging Multimodal Health Records

Proposing a machine learning model that uses late fusion to combine multiple supervised learning approaches to predict kidney graft loss.

submitted by

Jan Frick

Matr.-Nr.:828413

of the Faculty VI – Informatics and Media
of the Berliner Hochschule für Technik Berlin
presented Master Thesis
to aquiere the academic degree

Master of Science (M.Sc.)

in the field of

Media Informatics

Date if delivery September 12, 2022

Studiere Zukunft

Supervisors and Examiners

Prof. Dr. Alexander Löser	Berliner Hochschule für Technik
Prof. Dr. Agathe Merceron	Berliner Hochschule für Technik
Dipl.-Ing. Jens-Michalis Papaioannou	Berliner Hochschule für Technik
Dr. Med. Marcel Naik	Charité – Universitätsmedizin Berlin

Abstract

People with a kidney transplant live with a constant risk of kidney graft loss. Identifying patients that have a high chance of graft loss enables physicists to increase the medical care for them and delay or ideally prevent this event. We propose a machine learning model that finds high risk patients by predicting kidney graft loss. The model is trained with dataset consisting of 2893 patients that received a kidney transplant at the Charité – Universitätsmedizin Berlin. The data consists of demographic, sequential and text data (i.e., examination reports, physician letters). The text data is converted into embeddings by the state of the NLP model BERT.
The final model is a stacked ensemble that combines the predictions that have been created using multiple supervised learning techniques including XGBoost, Multilayer Perceptron and T-LSTM. We train the models for two different tasks, first a long term task to predict kidney graft loss between one and six years after transplantation, and second a short term task to predict kidney graft loss between one and two years after transplantation.
Our results show that we can successfully train a model for the long term prediction task, achieving a recall of 0.83 and F1-Score of 0.66. A lack of positive samples for the short term task leads to worse results. The ensemble model achieves a recall of 0.47 and F1-Score 0.52.
Overall, we can show that a set of data balancing techniques, incorporating text data and finding the right fusion method are key elements to train a model with multimodal data and an imbalanced dataset.

Declaration of Authorship

I hereby certify that the thesis I am submitting is entirely my own original work, except where otherwise indicated. I am aware of the University's regulations concerning plagiarism, including those regulations concerning disciplinary actions that may result from plagiarism. Any use of the works of any other author, in any form, is properly acknowledged at their point of use.

Date Signature

Contents

List of Figures

List of Tables

List of Abbreviations

Abbreviation	Definition
ML	Machine Learning
AI	Artificial Intelligence
ESRD	End Stage Renal Disease
EHR	Electronic Health Records
ANN	Artificial Neural Network
LSTM	Long-Short Term Memory
T-LSTM	Time Long-Short Term Memory
LR	Logistic Regression
SVM	Support Vector Machine
RF	Random Forrest
MLP	Multilayer Perceptron
BERT	Bidirectional Encoder Representations from Transformers
MLM	Masked Language Model
BCE	Binary Cross Entropy
FL	Focal Loss
HL	Hinge Loss
SGD	Stochastic Gradient Descent
GD	Gradient Descent
Adam	Adaptive Moment Estimation
WRS	Weighted Random Sampling
TP	True Positive
FN	False Negative
FP	False Positive
TN	True Negative
TPR	True Positive Rate (also referred as Recall)
TNR	True Negative Rate
Acc	Accuracy
BAcc	Balanced Accuracy
MCC	Matthews Correlation Coefficient
AUC-ROC	Area Under the Receiver Operating Characteristics
AUC	Area Under the ROC Curve
PR-AUC	Precision Recall Area under the ROC Curve
Dataset MO	Dataset Mitte Old
Dataset MV	Dataset Mitte Virchow
AKI	Acute Kidney Failure
HPO	Hyperparameter Optimization
ST	Short Term
LT	Long Term

Chapter 1

Introduction

Machine learning (ML) is the scientific discipline that focuses on how computers can learn automatically thought experience. It is a subfield of artificial intelligence (AI) that combines computer science and statistics. Within AI, it has emerged as the method of choice for developing practical software for computer vision, speech recognition, natural language processing, robot control, and other applications. Many engineers prefer developing ML models for many tasks by showing it examples of desired input-output behavior instead of classic programming over developing if-else schemes [Jordan und Mitchell 2015].

[Xing et al. 2018] debates the impact of AI in medical physics research and practice. The debaters are Ph.D. Dr. Xing and Ph.D. Dr. Krupinski. Xing is the Jacob Haimson professor of medical physics and director of Medical Physics Division of Radiation Oncology Department at Stanford University. Krupinski is an experimental psychologist and the vice chair for research in the Department of Radiology at Emory University.
Xing argues that AI will fundamentally change the field of medicine and physicists need to adapt for the new world:

> We are on the verge of AI revolution that will fundamentally alter the field of medical physics and the way medicine is practised. … Will AI eventually be able to perform all the activities of medical physicists? No! But, those physicists who know little informatics/AI are more likely to be replaced by those who do.

Krupinski points out that AI just mimics the data it's been trained on and misses the creativity that is required for multiple medical physicists tasks.

> Deep learning and AI are still a long way from being creative and this has been the case from the very beginning of AI implementations … whether it is solving a complicated clinical problem, developing a new line of research investigation, or communicating and collaborating with colleagues and patients, involves creativity and ingenuity.

While there is a debate about how the AI will change the field of medicine, both debaters agree that change is going to come. This thesis does not aim for a radical change, but rather support medical physicists in research and decision-making. The general goal is to develop machine learning models that predict kidney graft loss.

1.1 Motivation

Maintaining healthy kidneys is fundamental to staying overall healthy and keeping a general well-being. This is due to its vital functions that include removing urea and liquid waste from the blood in the form of urine, to regulate the body's blood pressure and to balance salts, electrolytes and other substances in the blood. Furthermore, they aid the formation of red blood cells and keeps the fluid and acid-base balance neutral [hopkinsmedicine.org 2021].
Conditions like repeated urinary infections, diabetes, or high blood pressure harm the kidneys, which can ultimately lead to an end stage renal decease (ESRD). This is a permanent stage of chronic kidney disease, where kidney function has declined to the point that the kidneys can no longer function on their own.
A patient having ESRD has two options to keep their body healthy: Dialyses or recieving a kidney transplant. Dialysis is a sub-optimal solution, because it worsens the patient's life quality significantly and is very expensive. A kidney transplant is the preferable solution, but waiting lists for transplants are long. At the end of 2021 11,156 people were waiting for a kidney in Germany. Over the year, 2,653 people were newly registered and 3,344 ones left due to transplantation, death or other reasons [DSO 2021]. Patients that revived a kidney transplant have the risk of a kidney graft loss. Figure 1.1 shows the risk of a graft loss after transplantation. While one-year kidney graft survival has improved over the years, the graft survival afterwards stayed constant.

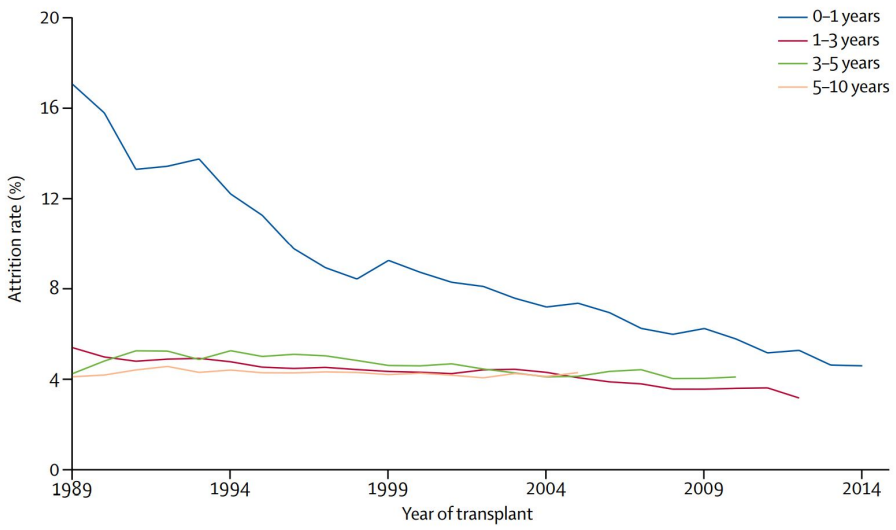

Figure 1.1: All-Cause Graft Loss Yearly Attrition Rates in the USA, by Year of Transplant. Rates of one-year graft loss have dropped substantially over the past 25 years, but rates of longer-term graft loss have remained relatively constant [Wekerle et al. 2017].

In order to improve the kidney graft survival after a transplantation, machine learning can help by identifying patients with a high risk of kidney graft loss. This identification assists in the way that the follow-up-care can be more personalized, increased for high-risk patients and decreased for low-risk patients. Thus, it ideally results in a general increased kidney health and reduced costs for healthcare.
Finally, a well-trained machine learning model could provide information about which data is the most significant for a high risk of kidney graft loss. This information would support further researches to reduce the long term risk of a kidney graft loss.

1.2 Task

The main task of this project is to predict, if a patient that received a kidney transplant will undergo kidney graft loss. This is accomplished with a machine learning approach. Setting up a machine learning approach can be divided into three steps. The first is to get a suitable dataset, the second is to preprocess the data and the last one is to train a machine learning model. The Charité – Universitätsmedizin Berlin supplies the data in form of electronic health records (EHR). The data preprocessing includes filtering, cleaning and transforming the data into a format that machine learning models understand. This step also creates labels that divide the EHR into two classes:

- Positive Class: Patients with a kidney graft loss prediction
- Negative Class: Patients without a kidney graft loss prediction

In order to train a model, it will map the EHR as an input vector to a class as an output vector. This task is called classification and the learning process is called supervised learning. First solutions for this task have been developed by [Haldar 2020] and [Hienen 2021]. They tested different ML learning models and achieved the best results using a random forest model. It uses the demographic information of a patient and data that has been collected over a year after the transplantation that includes laboratory values and medication prescriptions. The model of [Haldar 2020] predicts if a kidney graft loss will appear in short term (between one and one and a half years after transplantation) and the model of [Hienen 2021] long term (between one and six years after transplantation). Figure 1.2 visualizes the time frames.

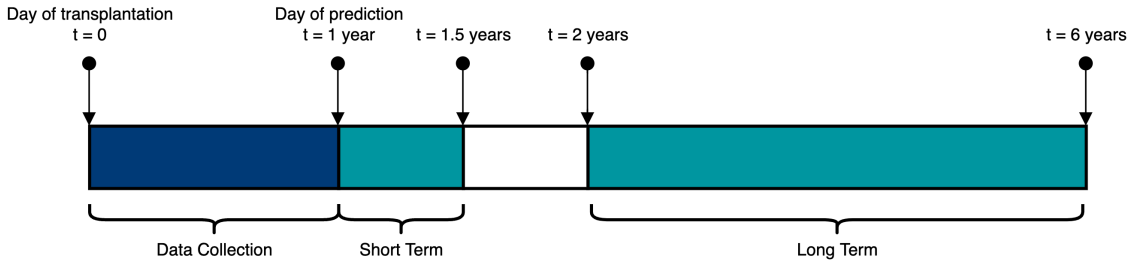

Figure 1.2: Data collection and prediction time frames. The graphic shows three time frames after the day of transplantation. The first is the year of data collection, the second is the short term prediction and the last is the long term prediction.

Splitting the positive class into short tern and long term results in a model that is able to predict a period in which the event occurs. This split comes with the downside that the positive class is smaller, since only a small amount of kidney graft loss events happen in this specific time-frames. [Haldar 2020] and [Hienen 2021] could create a random forest model that resulted in a F1-score of 0.53 for short-term and 0.61 for long-term.
[Reuter 2021] continued the work on the short term task and [Islam 2021] continued the work on the long term task. They presented more sophisticated models using artificial neural networks (ANN). With ANN, they were able to build an ensemble model that is able to include text data from the data collection process. The main finding was that text data can improve the prediction score. Short term prediction could be improved to a F1-score of 0.59 and long term to 0.64.

1.3 Problem Definition

In general, the main problems to develop a successful kidney graft loss prediction model using the TBase dataset are:

- Small dataset of around 3000 samples
- Imbalanced data that hast only 1-3% positive samples
- Multimodal data structure including sequential data and text

A more detailed problem definition can be done by looking at the work of [Reuter 2021] and [Islam 2021]. While they could improve the work of [Haldar 2020] and [Hienen 2021], they also pointed out problems with their models. They found that the main problems already occur during the data preprocessing. First, values used to create means haven't been unified. In a metric system, this easily results in values that are wrong by the factor of 1,000. Another problem is that values with a negative value have been excluded totally in the creation of mean values. Moreover, some features like gender appeared multiple times, due to different spellings, e.g., 'female' and 'Female'. In this case, all spelling versions of this feature need to be clustered to one term. Last, if a EHR had missing categorical features, it was sometimes replaced with a continuous value. All these problems in data preprocessing result in a lot of noise in the data and needs to be handled.

Furthermore, they addressed the problem of class imbalance with several methods, but suggested putting further effort in the topic, since the positive class performs worse than the negative class. This especially applies for short term prediction where the model of [Reuter 2021] could predict 95% of the negative class correctly but only 43% of the positive class.

[Islam 2021] criticizes that the data of the ensemble model is trained separately and a multi-modal fusion approach has the potential to outperform the ensemble model. The problem by training the models separately is, that the models don't have the full context. E.g., if the text model would have context of the patient's demographic data, it may interpret the text differently depending on criteria like age, weight, or reason for receiving a kidney transplant.

The data representation of a patient can be divided into time-variant and time-invariant data. The time-variant data is sequentially gathered during the data collection process and includes values like creatinine, protein, medication and all texts. This data can't be processed properly with the current models, instead they are using arithmetic mean values. Creating mean values looses the information of rises and peaks, but especially a rise in creatinine can be an important indicator for a kidney graft loss [Palmisano et al. 2021]. In order to use this time-dependent information, [Reuter 2021] proposes to use the ML models Long-Short Term Memory (LSTM) or Time-LSTM (T-LSTM). These are ANN models designed to utilize information that is sequentially structured.

A discussion with the medical expert also showed that the application's target need to be redefined. [Reuter 2021] and [Islam 2021] considered the F1-Score to be the most important metric to maximize, as it represents the results of an imbalanced data set in a harmonic mean. But in a medical domain, it is much more important to predict all patients that will undergo a negative outcome correctly (positive class) than patients that will stay healthy (negative class). Lastly, the discussion also showed that [Reuter 2021] and [Islam 2021] didn't use all the available data. It was excluded, because the patients visited a different hospital that stored less data. This data set should still include relevant information, thus having the potential to improve the outcome.

1.4 Hypotheses

As a result of the problem definition, it can be concluded that the application has a lot of unused potential. The problems and suggestions can be summarized in four hypotheses:

Hypothesis 1: Data centric approaches improve all baselines

The first task of the thesis is to investigate the data preprocessing and the data itself. Fixing the problems in preprocessing or setting up a new one should lead to an improvement of all baselines. A deep-dive into misclassified samples of the data could indicate more problems in the data processing. Furthermore, including more data should also improve results. In order to quantify the outcomes, fixed machine learning models will be used, while the data changes. This process is called a data centric approach.

Hypothesis 2: Model centric approaches improve all baselines

The second hypothesis is that adapting the machine learning model setup will improve all baselines. This can be done by adding techniques to handle the class imbalance, trying different loss functions, or try new machine learning models. Machine learning models have many parameters, in order to find the best combination of parameters, loss functions and class imbalance techniques a hyperparameter optimization will be applied. The approach with fixed data and changing machine learning models is called model centric approach.

Hypothesis 3: Including the time-dimension creates a better baseline

Using a LSTM or T-LSTM creates a machine learning model that is able to use sequential information that isn't included in current models. Thus, a setup including one of these machine learning models should create a baseline that outperforms the current models.

Hypothesis 4: Redefining the target metric improves the results from a medical perspective

The target metric evaluates the models' outcome. Since we use an imbalanced dataset, it is important to choose a metric that also represents the minority class. From a medical perspective, it is even more important to predict the negative outcome correctly. More about class priorities can be found in Section 7.1.1. Current models only predict 43% of short term and 65% of long term kidney graft loss predictions correctly. Besides a weak general performance of the current models, this can also be due to chosen evaluation metric F1-score. Maximizing a different score could help to improve the positive class.

1.5 Thesis structure

The thesis is split into 8 chapters. The chapters cover the following topics:

Theoretical Background: This chapter shows the theoretical background that is needed to understand the concepts that are used throughout the thesis.

Dataset: The chapter dataset describes the data source TBase and the data itself.

Data Centric Approach: The chapter data centric approach describes the methodologies used to create a new data baseline. This includes adding new data, creating a new data preprocessing, and redefining the labels.

Model Centric Approach: This chapter is about the methodologies used for the model centric approach. It presents each ML model and the fusion method and hyperparameters it uses.

Implementation: The chapter shows the computing environment, used software packages and the implementation of early stopping, k-fold cross-validation and Hyper-Parameter Optimization.

Evaluation: The chapter evaluation presents the results of all conducted experiments, gives an error analysis and discusses notable topics.

Conclusion and Future Outlook: The final chapter concludes the thesis and give a future outlook about further improvements and practical use.

Chapter 2

Theoretical Background

This chapter shows the theoretical background that is needed to understand the concepts that are used throughout the thesis. The first section explains different machine learning approaches. It is followed by a section about relevant evaluation metrics for classification. The next section describes what kind of techniques are used to encounter the data imbalance and the last section is about different loss functions, which are used by the ML models.

2.1 Supervised Learning Approaches

[García et al. 2015] divides ML between supervised learning and unsupervised learning. Supervised learning is an approach to discover relationships between input attributes and a target attribute, while unsupervised learning has no target attribute and needs to find patterns like regularities, relationships, or similarities itself. With the aim to predict kidney graft loss, there is a clear target attribute for this task, thus we use supervised learning approaches to create machine learning models.

2.1.1 Logistic Regression

Logistic regression (LR) is a supervised learning approach of modeling the probability of a discrete outcome given an input variable. The LR Model is an extension to the linear regression model. In comparison to linear regression, it performs better in classification, because it returns a probability between 0 and 1 instead of an extrapolated value that can be below zero and above one. With probabilities, it is easy to set a threshold to distinguish between classes [Molnar 2022]. In order to calculate a binary output, the LR model applies a nonlinear sigmoidal function to the input, which can be a simple true/false. The model can be extended to a multinomial logistic regression in order to achieve a multiple class outcome [Ashenden 2021].

2.1.2 Support Vector Machine

The support vector machine (SVM) is an algorithm that finds a hyperplane in an N-dimensional space, where N is the number of features. The hyperplane separates the samples into two classes. The goal is to find the hyperplane with the biggest distance to both classes, so that the samples can be classified with more confidence. The distance from the hyperplane to a sample is called margin. Support vectors are samples that are close to the hyperplane. They influence the position and orientation of the hyperplane in order to maximize the margin [Gandhi 2018].

2.1.3 Random Forest

Random forest (RF) is a learning algorithm consisting of many decisions trees. A decision tree splits the data multiple times, starting at the root node. Each data split is a decision that ends

at the leaf node, the average outcome of all training data in a leaf node is used to predict the outcome. The advantage of a decision tree in comparison to linear or logistic regression is that it can also learn relationships between the features [Molnar 2022].
[Breiman 2001] proposed RF as an improvement to decisions trees, because they tend to overfit. When Decision Trees find features that are strong predictors, the trees select them many times, which creates a correlation. Instead, RF uses the ensemble learning method boostrap aggregation or short bagging. This creates many trees consisting of random features from the dataset. RF takes their majority vote for classification and average in case of regression.

2.1.4 XGBoost

XGBoost stands for extreme gradient boosting and is, like random forest, a decision-tree-based ensemble machine learning algorithm. It uses a gradient boost framework, which sequentially creates weak models and employs a gradient descent algorithm to minimize errors. XGBoost enhances this base framework through system optimization and algorithmic enhancements. System optimizations are parallelization of tree buildings, tree pruning using a depth-first approach, cache awareness and out-of-core computing. These system optimizations improve the computational performance significantly compared to a model with just gradient boosting. Algorithmic enhancements prevent overfitting with regularization, efficiently handle missing data and have a build-in cross-validation [Morde 2019].

2.1.5 Multilayer Perceptron

A multilayer perceptron (MLP) is a feed forward artificial neural network that generates a set of outputs from a set of inputs. Artificial neural networks are inspired by human brains and the way neurons of the human brain function together. Feed forward means that the network has only one direction and does not form a cycle. In its simplest form, a feed forward artificial neural network is made of a single layer perceptron. An MLP has an input and an output layer with one or more hidden layers, shown in Figure 2.1. The perceptron has a series of inputs that are multiplied with weights. The sum of all weighted inputs forms the output. This output can be used for classification, where a threshold determines the class of the output. The accuracy of the output can be increased by through learning. Learning occurs in the perceptron by adjusting connection weights after the data has been processed. The adjustment is based on the amount of error in the output compared to the expected result. The error is calculated by a loss function [Gupta 2019].
Forward / Backward Propagation

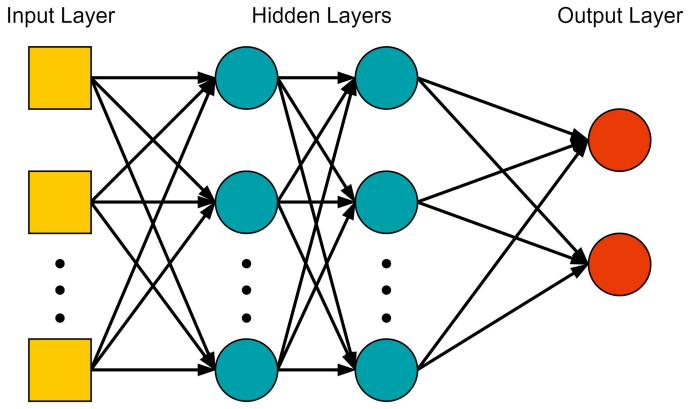

Figure 2.1: Multilayer Perceptron Network. The figure a multiplayer perceptron network consisting of input layer, two hidden layers and the output layer.

2.1.6 BERT

BERT is the abbreviation for bidirectional encoder representations from transformers.
[Devlin et al. 2018] proposed this language model with the key innovation of bidirectional training. Previous language models were only able to either look at a text sequence from left or right, or combined left-to-right and right-to-left training. BERT's bidirectional training is possible due to the pre-training objective called masked language model (MLM). MLM randomly masks some tokens of the input text and aims to predict the original vocabulary on it's left and right context. MLM joined with a next sentence prediction allows pretraining a deep bidirectional transformer. A pretrained BERT-Model can be adjusted to a variety of language tasks by adding another layer on top of the core-model. This process is called fine-tuning and can be used for classification tasks, question answering tasks and named entity recognition tasks [Horev 2018].
BERT utilizes the feedforward network transformer [Vaswani et al. 2017]. Transformers have a high level of parallelization and an attention mechanism that that allows learning contextual relations between words. Transformers itself include an encoder to read text input and a decoder to make predictions. Creating a language model with BERT only requires the encoder mechanism.

2.1.7 T-LSTM

[Baytas et al. 2017] proposed the T-LSTM as a time-aware version of the Long Short-Term Memory (LSTM) with the aim to handle irregular time intervals in longitudinal patient records. T-LSTM and LSTM are Recurrent Neural Networks that are able to process sequential data like speech, video, or time-series data. [Hochreiter und Schmidhuber 1997] proposed LSTM as an improvement to other RNNs, because their gradients signal tend to either blow up or vanish. Gradients are values used to update a neural network's weights. If the RNN gradient blows up it may lead to oscillating weights, if it vanishes it either takes a huge amount of time or does not contribute much to the learning process.
LSTM makes use of a cell state and various gates. The cell state acts as a memory that carries information through the sequence chain, thus creating a kind of long term memory. Gates are neural networks that learn what information is relevant to remember and what can be forgotten. The standard LSTM architecture with forget gates, input gates, output gates and a memory cell is designed to work sequential data with regular time intervals. [Baytas et al. 2017] proposed the T-LSTM to create a LSTM architecture that works with irregular time intervals by taking the time-lapse between sequential data into account. The architecture has a new input Δt in order to process the time-gap between the data. If the data has a high Δt it means that there is no new information recorded for a long time and the short-term memory should play a less significant role for the current prediction.

2.1.8 Stacked Ensemble

Stacking is used to create an ensemble ML model, that is able to combine the strengths of different ML models. It does so by aggregating probabilities of different ML models as an input and creating a new classification as an output. It is common to use a linear model such as LR or SVM with linear kernel to create a stacked ensemble model for classification.

2.2 Loss Functions

Loss functions calculate the error between the result of a machine learning model and the actual truth. An optimization algorithm uses the loss to update ML model parameters, e.g. the weights of a neural network. There are multiple options for loss functions, this section presents three loss functions for binary classification problems.

2.2.1 Binary Cross Entropy

Binary Cross Entropy (BCE) compares each predicted probability to the true value (0 or 1) and calculates the score. The score is calculated based on the distance from the expected value. It is logarithmic scaled between 0 and 1 with a low score for small differences and a high score for large differences. The final loss is the average of all scores. BCE can use class weighting in order to address class imbalance. A weighting factor multiplies the loss of positive samples. Thus, the loss will be stronger in misclassified positive samples. This weighting factor will be referred to as positive weight (pos_weight) [Lin et al. 2017].

2.2.2 Focal Loss

[Lin et al. 2017] proposed Focal Loss (FL) as an improved version of BCE that is designed to address extreme imbalance. A focussing parameter gamma reduces the loss of well-classified examples and increases the loss of bad-classified examples.

2.2.3 Hinge Loss

Hinge Loss (HL) is usually used for training SVM models. It calculates the loss from margins to the hyperplane. For classification with MLP it calculates the loss from labels and probabilities.
The loss calculation depends on labels for the class. In our case 0 is negative true and 1 is positive true. For the negative class all predictions with the value of 0 or below results in a loss of 0, and higher values are the difference to 0. For the positive class all predictions with a value of 1 or higher result in a loss of 0 and predictions with a value of below 1 result in the difference to 1 [Gentile und Warmuth 1998].

2.3 Regularization Techniques

Regularization refers to a set of different techniques that apply a penalty to increase the magnitude of parameter values in order to reduce overfitting. This section explains the three most common regularization techniques called L1, L2, and dropout.

2.3.1 L1 and L2 Regularization

L1 is also called Lasso algorithm [Tibshirani 1996]. L1 adds the absolute value of magnitude of the coefficient as a penalty term to the loss function. Thus, coefficient of less important features are stronger reduced or even excluded.
L2 is also called Ridge Regression [Hoerl und Kennard 2000]. L2 adds the squared magnitude of the coefficient as a penalty term to the loss function, which helps to avoid over-fitting.

2.3.2 Dropout

Dropout randomly drop units temporarily from a neural network during training. This prevents units from co-adapting too much. During training, dropout samples from an exponential number of different "thinned" networks [Srivastava et al. 2014].

2.4 Optimizers

Optimizers are algorithms or methods used to change the attributes of your neural network such as weights and learning rate in order to reduce the losses.

2.4.1 Stochastic Gradient Descent

The Stochastic Gradient Descent (SGD) is an extension of the most basic optimization algorithm Gradient Descent (GD). GD uses the entire dataset to calculate the derivative and update the weights. This requires a lot of memory, SGD overcomes this by only taking one sample at a time. SGD performs frequent updates with a high variance, which causes the objective function to fluctuate. This fluctuation helps to overcome the function getting stuck at a minimum [Ruder 2016].

2.4.2 Adam and AdamW

[Kingma und Ba 2014] proposed Adaptive Moment Estimation (Adam) as a method for efficient stochastic optimization. It is an algorithm designed for large datasets and/or high-dimensional parameter spaces. The method combines the advantages of two other extensions of stochastic gradient descent: First, the ability of Adaptive Gradient Algorithm that maintains a per-parameter learning rate to deal with sparse gradients. And second, the ability of Root Mean Square Propagation that also keeps per-parameter learning rates that are adapted based on the average of recent magnitudes of the gradients for the weight.
[Loshchilov und Hutter 2017] proposed AdamW as an improved version of Adam. They showed that the weight decay of Adam is coupled to the gradient update, thus it does not perform like the intended L2 regularization. AdamW decouples the weight decay from the gradient update, which leads to a better generalization performance than Adam.

2.5 Data Balancing Techniques

Real world Datasets are often imbalanced, thus the distribution across the known classes is biased. The imbalance can range from a slight bias to an extreme imbalance, where for one example in the minority class there are hundreds, thousands, or millions of example in the majority class. Class imbalance challenges classification tasks, because most ML models assume an equal number of samples for each class. Their optimization is designed to reduce a loss function. By default, predicting the majority class more accurate results in a bigger loss reduction, thus they develop poor accuracy for the negative class.
Section 2.2 shows loss functions that use a positive class weight to encounter the class imbalance problem. This section presents the methods Downsampling, Oversampling and Weighted Random Sampling to encounter the class imbalance

2.5.1 Stratified Test Split

Not all the samples can be used to fit a machine learning model. They need to be split in three datasets:

- Training dataset: Data that is used to fit the parameters to the machine learning model
- Validation dataset: Data that used to evaluate different combinations of hyperparameters
- Test dataset: Data that is used to evaluate the generalization ability of the final model

A stratified test split is applied to ensure that each split represents the dataset. The main criteria should be the label, so that each split has an equal ratio of labels. The stratifying criteria can be extended to other features, e.g. to ensure that each split has a similar distribution of age.
Ensuring that each split represents the dataset is also helpful for balanced datasets, but imbalanced datasets require an even distribution of labels to ensure that the model can learn and evaluate every class.

2.5.2 Down- and Oversampling

Downsampling and oversampling are addressing the class imbalance by resampling the dataset. Downsampling reduces the size of the majority class, and oversampling increases the size of the minority class. These methods are only applied to the training dataset, in order to avoid manipulating the evaluation of the validation and test dataset. Oversampling is achieved by implementing SMOTE [Chawla et al. 2002]. SMOTE creates synthetic samples of the positive class to create a balanced dataset.

2.5.3 Weighted Random Sampling

Weighted Random Sampling (WRS) can be used to reduce the bias in a unbalanced dataset. WRS assigns weights to each class relative to the class distribution. When the ML model is loading the training dataset, WRS randomly picks samples, where the probability of each sample to be selected is determined by its relative weight [Efraimidis 2010].

2.6 Evaluation Metrics for Classification

Evaluation metrics are needed to evaluate the effectiveness and efficiency of a classification ML model. There are multiple evaluation metrics that are established and used by the scientific community. This section gives an overview about all evaluation metrics that are used in the thesis.

Confusion Matrix

Data that is used to train a machine learning model needs to be labeled. The label is the true value for each sample. In binary classification, a label can either be positive or negative. A prediction ML model predicts the class of the sample. After prediction, each sample has a true value and a predicted value, resulting in four possible outcomes:

- True Positive (TP): True value positive and predicted value positive
- False Negative (FN): True value positive and predicted value negative
- False Positive (FP): True value negative and predicted value positive
- True Negative (TN): True value negative and predicted value negative

The sum of all samples for each outcome is displayed in a confusion matrix shown in Figure 2.2. The confusion matrix is used to evaluate the performance on each class.

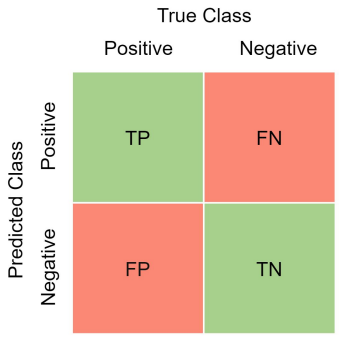

Figure 2.2: The figure shows an example of a Confusion Matrix.

Recall

The recall or true positive rate (TPR) is the percentage of positive samples that have been predicted correctly.

$$Recall = \frac{TP}{TP + FN}$$

True Negative Rate

The true negative rate (TNR) is the percentage of negative samples that have been predicted correctly.

$$True\ Negative\ Rate = \frac{TN}{TN + FP}$$

Accuracy

The accuracy (Acc) is the percentage of all samples that have been predicted correctly.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Balanced Accuracy

The balanced accuracy (BAcc) or Macro Recall is the arithmetic mean of recall and TNR.

$$Balanced\ Accuracy = \frac{Recall + TNR}{2}$$

Precision

Precision is an evaluation metric that shows the percentage of positive predicted samples that are correctly predicted.

$$Precision = \frac{TP}{TP + FP}$$

The macro precision is shown throughout the thesis. It calculates metrics for each label, and finds their unweighted mean. This does not take label imbalance into account.

$$Macro\ Precision = \frac{Precision_{pos} + Precision_{neg}}{2}$$

F1-Score

The F1-Score represents the harmonic mean of precision and recall.

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall}$$

Matthews correlation coefficient

The Matthews Correlation Coefficient (MCC) statistical rate, which only produces a high score, the confusion confusion matrix achieved good results in all four outcomes [Chicco und Jurman 2020].

$$MCC = \frac{TP * TN - FP * FN}{\sqrt{(TP + FP) * (TP + FN) * (TN + FP)}}$$

AUC-ROC

AUC-ROC also known as Area Under the Receiver Operating Characteristics is a measurement for distinction amongst classification problems at various thresholds [Bradley 1997]. The Area under the ROC Curve (AUC) score can be used to determine the effectiveness of classification models and is calculated using the ROC-curve. The ROC curve is a graphical evaluation method that is not dependent of a specific threshold. ROC graph is a plot of FPR on the x-axis, and TPR on the y-axis. There is a point on the plot for each possible threshold based on the values of FPR and TPR. The curve is drawn by linear interpolation among these points [Fernández et al. 2018].

PR-AUC

PR-AUC stands for Precision Recall AUC and is an alternative to AUC-ROC. The graph is plotted using precision and recall instead of TPR and TNR. Recall is on the x-axis and precision on the y-axis. PR-AUC is more reliable for an evaluation using an imbalanced dataset [Fernández et al. 2018].

2.7 Summary

In this chapter, we cover all the necessary fundamentals that are needed to understand this thesis. First, we briefly explained all supervised Ml models that we use to predict kidney graft loss. The next section covers loss functions that ML models use to calculate the error between the result of a machine learning model and the actual truth. The Section 2.3 shows regularization techniques that ML models use to reduce overfitting. The following section covers optimizers. Optimizers change attributes like weights and learning rates during the training of an ML model in order to minimize the loss. In the last section, we show evaluation metrics that we use to evaluate the effectiveness and efficiency of a classification ML model.

Chapter 3

Dataset

This chapter describes the data source and the data itself. All about preprocessing is included in Chapter 4. The first section covers the EHR TBase that is used as the database. The Section Multimodal Patient representation gives a detailed overview about the leveraged TBase Data.

3.1 Electronic Health Record TBase

TBase is an EHR for kidney transplant recipients. Kidney transplant recipients are referred as patients throughout the thesis. It combines data entries of clinical data (e.g., clinical notes, medication lists, radiology, and pathology data) with manual data entries (e.g., clinical notes, medication list, and transplantation data). Charité – Universitätsmedizin Berlin provides data as a database for applications in routine clinical care and research [Osmanodja et al. 2021]. The database includes demographic data about organ receiver and donor, including information like age, height, or blood group. Furthermore, it keeps track of medications and laboratory values like protein and creatinine. TBase consists of about 10GB of data including 35 tables with 350 features. It recorded 6000 patients, 4000 transplantations, 60.000 diagnoses, 9 million laboratory values in 100.000 visits from 1999 to 2014 [Christoph et al. 2015].
The database includes demographic data about organ receiver and donor, including information like age, height, or blood group. Furthermore, it keeps track about medications and laboratory values like protein and creatinine.
Each patient that is included in TBase represents a patient that had at least one kidney transplant.

3.2 Multimodal Patient Representation

Not all patients and Features that are included in TBase are used, Section 4.1 shows how different criteria for a cohort selection create two datasets:

- Dataset Mitte Old (MO): Includes 1263 patients that underwent the kidney transplant in Charité Mitte
- Dateset Mitte Virchow (MV): Includes 2893 patients that underwent the kidney transplant in Charité Mitte or Virchow

3.2.1 Time Invariant Demographic Data

The time invariant demographic data is all information about a patient that is highly unlikely to change during the patient observation. Thus, it will be seen as fixed data that remains the same during the data collection process (e.g., the donor or recipient age does not change at the birthday). This information will be referred to as demographic data. [Fernández et al. 2018] categorizes data in quantitative, which can be discrete or continuous, and qualitative, which can be nominal or ordinal. Table 3.1 shows the demographic features divided by the data categories:

Quantitative		**Quantitative**	
Discrete	**Continuous**	**Nominal**	**Ordinal**
Ischaemie_kalt	Recipient Height	Blood group Recipient	Birth Date
Ischaemie_warm	MMA_broad	Blood group Donor	Graft Loss Date
Ischaemie_gemischt	MMB_broad	Basic Kidney Disease	Death Date
Dialysis Count	Donor Height	Cause of Death	First Dialysis Date
Donor Age	Donor Weight	Cause of Kidney Loss	Last Seen Date
		Sex of Recipient	
		Sex of Donor	
		Dialysis Kind	
		Location	
		Kind of Donation	
		Primary Function	
		Organ Quality	

Table 3.1: Time Invariant Demographic Data. The table shows the demographic data categorized into discrete, continuous, nominal and ordinal values.

While most of the demographic data is self-explanatory, some need a brief explanation. Ischaemie describes the ischemia times. 'Ischaemie_kalt' is the time where the kidney was outside of a body e.g. a transport box. 'Ischaemie_warm' is the time, when the kidney is placed into the body and connected to the vessels. 'Ischaemie_gemischt' when the kidney is connected with the vessels and warmer. 'MM_broad' and 'MM_split' are the mismatches in immunological characteristics between donor and recipient in gene loci A, B and DR.

3.2.2 Text Data

All visits of patients in the hospital or in an outpatient clinic are documented with text. One type of text is clinical notes. They include information about the well-being of the patient, but can also include information about medication changes, evaluation of laboratory values or vital parameters, treatments, the history of the current illness or complaints.

The other type of text is results of specific exams during their visits like pathology, microbiology, x-ray, or ultrasound. Some of them are highly structured, others contain continuous text from a physician's survey.

3.2.3 Time variant Data

Sequential data can exist multiple times during the data collection process and is called time variant data. It can be split into three categories: Laboratory values, medications and hospitalization. Laboratory values and medications are continuous data, and hospitalization is nominal. Table 3.2 shows the time variant features divided by type of feature.

Laboratory Values	Medications	Hospitalization
KreatininHP	L04AD01	Hospitalization
ProteinTUR	H02AB04	
ProteinKSU	L04AA06	
ProteinDSU	L04AD02	
ProteinCSU	H02AB06	
LeukoTUR	L04AA10	
LeukoEB	L04AA04	
CRPHP	L04AC01	
AlbuminKSU	L04AC02	
AlbuminKUR	L04AX01	
	L04AA18	
	L04AA28	
	L04AB02	

Table 3.2: Time Variant Data. The table lists all time variant data split in the categories laboratory values, medications, and hospitalization. Protein, Albumin and CRP are protein, Kreatinin is creatinine and Leuko are leukocytes. The capital letters at the end of each laboratory value represent the measurement technique. Medications show medical codes that describe several immunosuppressants that are relevant from a medical perspective. Hospitalization shows if the patient was in the hospital or in an outpatient clinic at a date.

3.3 Summary

The chapter dataset gives an overview about TBase and the leveraged data. TBase is an EHR for kidney transplant recipients. There are two datasets, Dataset MO (1263 patients) and Dataset MV (2893 patients). The data is multimodal, consisting of time invariant data (demographic data), time variant data (laboratory values, medications, and hospitalization) and text (clinical notes and exams from pathology, microbiology x-ray and ultrasound).

Chapter 4

Data Centric Approach

This chapter shows, how a data centric approach improves the performance of all models. [Ng 2021] defines the goal of data centric AI with following question:

> How can you systematically change your data (inputs x or labels y) to improve performance?

Therefore, we use fixed ML models and change the data preprocessing for this approach, resulting in new data baselines. The models being used to evaluate the results of this approach are RF and an MLP model with text data. We chose RF, because we can easily compare results of dataset MV with the RF model that [Haldar 2020] and [Hienen 2021] trained with dataset MO. The RF model is not trained with text, thus we need the MLP text model to be able to compare the differences in the text preprocessing. A big difference between the preprocessing of the data baselines thesis is the included data, therefore they are referred to as dataset Mitte Old (MO) and dataset Mitte Virchow (MV):

- **Dataset MO**: Includes Patients that underwent the kidney transplant in Charité Mitte. Used in [Haldar 2020], [Hienen 2021], [Reuter 2021] and [Islam 2021].
- **Dateset MV**: Includes Patients that underwent the kidney transplant in Charité Mitte or Virchow. The model centric approach in Chapter 5 uses it.

The preprocessing for dataset MV is newly set up, since the dataset MO has too many problems to investigate the code and fix it. Furthermore, the investigation of the misclassified samples from [Reuter 2021] and [Islam 2021] reveals additional improvements that can be applied at each step of data preprocessing. [Fernández et al. 2018, 5–6] divides the preprocessing into four steps:

1. Selection of the data
2. Preparation of the data
3. Transformation of data
4. Reduction of data

The following sections can be assigned to the steps of data preprocessing. Sections 4.1 and 4.2 describe the selection process. Data preparation is shown in section 4.3. The transformation of data is described in Sections 4.4, 4.5, 4.6, 4.7 and 4.8. Reduction of data is also shown in Section 4.8, but not to create a quick estimation of the performance, as intended in [Fernández et al. 2018]. It is used to handle data imbalance.

4.1 Cohort Selection

Overall TBase contains 19,642 kidney transplant recipients, but in order to train a machine learning model they need to have a minimum quality. To achieve this quality, potential outlier and sparse EHR need to be filtered. The dataset MO has the following criteria for the cohort:

Cohort criteria dataset MO

- Have no other organ transplant than kidney transplant
- Are at least 18 years old
- Received their kidney transplant after year 1999
- Underwent the kidney transplantation in Charité Mitte
- Didn't have a kidney graft loss during the data collection

Applying these rules to dataset MO results in a cohort of 1,263 Patients. Patients under 18 are excluded, because their aftercare is provided by the pediatrics department. Patients that had their transplantation before year 2000 are excluded due to incompatible text data. Patients that have a kidney graft loss within the first year after transplantation are excluded, since the label definition wouldn't classify them correctly. It would assign them a negative label, which should include patients with a low risk of kidney graft loss. But if a patient has a kidney graft loss within the first year, they should have had a high risk.
The investigation of dataset MO shows that these criteria needs to extended. Patients that die during the data collection timeframe need to be excluded, since they act as outliers. They are outliers, because there is no information about if the death was related to the kidney transplant. Furthermore, the investigation shows that dataset MO still includes patients that have a kidney graft loss event during the data collection. They are excluded for dataset MV. Applying the finding results in the following criteria:

Cohort criteria dataset MV

- Have no other organ transplant than kidney transplant
- Are at least 18 years old
- Received their kidney transplant after year 1999
- Underwent the kidney transplantation in Charité Mitte or Virchow
- Didn't have a kidney graft loss before prediction time frame
- Didn't die during the data collection or prediction time frame

Applying these rules to dataset MV results in a cohort of 2893 KTR.

4.2 Feature Selection

Not all the available data should be used in the process of training a machine learning model. Features that are sparse can introduce noise, which increases the memory needs of the model without improving it. This can be avoided by dropping these features [Prakash 2021].
Another reason to exclude features is, that they include information that is not known at the time of prediction. This is the case for the features 'Cause of Death', 'Cause of Kidney Loss',

'Graft Loss Date' and 'Death Date'. These data is only used to create the labels, but not during the training.
The time variant data includes some very sparse features. 'AlbuminKSU' and 'AlbuminKUR' only appear at 3% and 5% respectively of all days that have laboratory values, thus they are dropped. Most of the medications are also very sparse. A discussion with the medical expert showed that only the 13 medications shown in Table 3.2 are relevant. All other medications are dropped.

4.3 Data Cleaning

The objective of data cleaning is to ensure the quality of the selected data. After this step, the data should be free of any inconsistencies [García et al. 2015, p.10]. The described cleaning of demographic and time variant data only applies to dataset MV. Dataset MO uses the cleaning of [Haldar 2020] and [Hienen 2021]. Besides one improvement of the text cleaning routine it is the same as in [Reuter 2021] and [Islam 2021].

Demographic Data

Exploring the demographic data shows that the discrete, continuous and ordinal data is consistent, while the nominal data has a high variance in categories. A high variance makes it difficult for a machine learning algorithm to recognize patterns. In order to decrease it, similar categories are clustered. The clustering is done in cooperation with medicals experts, that manually investigated all nominal data. E.g., an investigation of feature 'Basic Kidney Disease' may show that there are the diseases A1, A2, A3, B1 and B2, the expert decides that all A diseases are similar, while B diseases are not. Then, the diseases are clustered to A, B1 and B2.

Time Variant Data

An exploration of the time variant data shows a lot of noise due to inconsistent value types and ranges. Figure 4.1 shows cleaning steps that are applied to the laboratory values to reduce the noise.

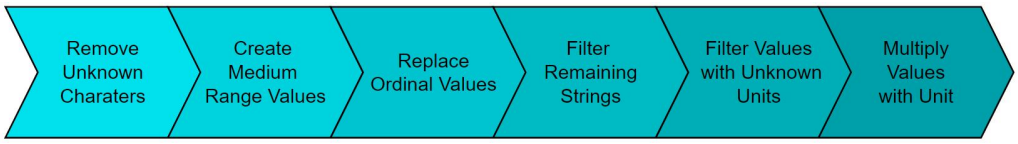

Figure 4.1: Laboratory Data Cleaning Steps. The figure shows the sequence of cleaning steps applied to laboratory data in order of their application.

Remove Unknown Characters. Many values have the type string instead of a numerical data type. Most of the time it is to imply the inaccuracy of the measurement by using characters like '>', '<' or 'ca.'. To convert this values to numerical values, the string characters are filtered. A value of '<3' becomes to '3'.

Create Arithmetic Mean Range Values Other strings represent ranges like '5-10', they are replaced by their medium value. In this case, the value '5-10' becomes '7.5'.

Replace Ordinal Values. Some measurement techniques create ordinal values that are documented with pluses and minuses, e.g. '+', '+++', or '-'. This step replaces them with comparable values of the continuous measurement techniques. E.g. a 'ProteinTUR' value of '+++' is converted to a value of 500.

Filter Remaining Strings. The remaining strings in the laboratory data are uninterpretable values like 'Falschabnahme', 'MATERIAL' or 'nicht bestimmbar'. All remaining values of type string are filtered in this step.

Filter Values with Unknown Units. Each laboratory value has a unit, e.g. 'mg/l' or 'g/l'. Some values with very rare or unknown units like 'ml/min', 'Gpt./l' or 'mg/g Krea.' are filtered.

Multiply Values with Unit The last step is to unify the units for each kind of laboratory value. Values that don't have the most common unit are factorized, so that all values have the same unit. For example the most common unit for kreatininHP is 'mg/dl', a value with a different unit like '50 mg/l' is converted to '5 mg/dl'.

Text Data

The text data is the most inconsistent part of the available data. While exams still have a bit of consistency in the way they are structured, clinical notes can be written down in various ways depending on the medical personnel. The unstructured documentation combined with personal preferences of documenting text leads to a high variance. Furthermore, the texts include a vast amount of medical abbreviations, where different abbreviations can have the same meaning. This use of abbreviations with a text model that has not been fine-tuned to understand them might lead to a loss of information [Shinohara et al. 2013]. Additional noise like HTML tags and meaningless symbols is created between different interfaces from the user interface to the databank request. In order to handle the inconsistency and noise of the text data, multiple cleaning steps have to be applied. Figure 4.2 shows the order of these steps.

Figure 4.2: Text Data Cleaning Steps. The figure shows the sequence of cleaning steps applied to text data in order of their application.

Replacing Unknown Characters. The text contains a lot of special characters that BERT is unable to encode. These characters can be Greek or Latin letters, like μ or $\varnothing$. During tokenizing, BERT replaces out-of-vocabulary tokens with UNK tokens [Wolf et al. 2020]. The information of the character gets lost in this process. To prevent this from happening, the special characters are replaced before tokenizing. There are three types of replacements. First, replacing characters with a term that includes the meaning of the character, e.g. '°' to 'Grad, 'μ' to 'micro' or ½ to '1/2'. Second, replace letters having accents with the basic letter, like 'é' to 'e'. Lastly, remaining special characters that aren't classifiable are completely removed.

Remove Unwanted Text with Regex. The next step is to remove unwanted text chunks. One reason to remove text chunks is that they don't include information such as HTML-tags, URLs and short phrases like 'gesehen durch' or 'Therapie durch'. Other text chunks include information about the patient like telephone numbers, zip codes, city names, admission numbers, laboratory numbers and need to be removed for privacy reasons.

Convert TNM-Codes. The texts include TNM classifications. TNM is a system for the classification of cancerous tumors that is internationally used by doctors and researchers. The abbreviation stands for tumor, nodes and metastases [IQWiG 2016]. During this step, they are translated to readable text describing the state of tumor.

Replace Abbreviations. In order to handle the abbreviations, they are unified and written in complete words using a dictionary. 2822 medical abbreviations for the dictionary are available using Wikipedia[1]. 300 of them were ambiguous and thus translated by the medical expert.

Add Fullstop add End of Text. In Section 4.4 texts that are written down on the same day are concatenated. This step creates a paragraph between the concatenated texts by adding a fullstop at the end of each text.

Replace Small Texts. Some texts contain no or very little information. In correspondence with the medical expert, we decided that text with fewer than five words don't feature any problematic occurrences. Thus, this step replaces the content with 'Patient lebt.', which means the patient is alive.
[Reuter 2021] and [Islam 2021] proposed a more sophisticated text cleaning approach, which resulted in a worse performance. A further investigation of the cleaning routine showed that too many texts of type 'Microbiology' are transformed to 'Patient lebt.' Thus, we exclude this transformation for text of type "Microbiology" in the cleaning of dataset MV.

Add Category into Text. This step adds the category of each text at the beginning.

Remove Outliers. Some text data contain the same text multiple times. This creates very long and noisy texts that are outliers. In order to remove these outliers, texts with a text length above 99% quantile are removed.

4.4 Feature Engineering

The machine learning models used for this task are not able to process raw text data, the text needs to be transformed into a numerical representation. [Reuter 2021] and [Islam 2021] trained multiple BERT models for this transformation and achieved the best result using German Base-Bert [Chan et al. 2020].
We use this model to generate the BERT-embeddings for dataset MO and dataset MV. The performance of the model is improved by pretraining it on predicting risk-factors of graft loss. The risk factor labels are created using the RIFLE scheme [Cruz et al. 2007]. The scheme defines stages of acute kidney failure (AKI), thus the BERT model connects risk factors with the text.
If a patient has multiple texts for one day, they are concatenated before transforming them to BERT-embeddings. After the transformation, the BERT-embeddings are in a sequential numerical representation. Most machine models used for this task are not able to process sequential data. This data needs to be transformed into a single feature vector format. For the text data, this is the mean of all BERT-embeddings for each patient. Time-variant data appears more frequently, especially in the first months after the transplantation. Thus, a mean is created for each of the first six months for every time-variant feature. Additionally, a mean value over the whole data collection time frame is added to the feature vector for each time variant feature. Figure 4.3 visualizes this process.

T-LSTM is an ML model that is able to process sequential data. Each patient requires multiple feature vectors that can be processed sequentially. In order to create this data structure, multiple feature vectors have to be created for each patient that contain text and time variant data. Since text data appears less frequently than the time variant data, we create one vector for each text and concatenate the time variant data. Because the time variant data is not necessarily available on the same date as the text, a mix of last observation carried forward, and next observation carried

[1] https://de.wikipedia.org/wiki/Medizinische_Abk%C3%BCrzungen

forward is used to replace missing data. Additionally, the demographic data is concatenated to every feature. The pseudo code in Algorithm 1 shows this process.

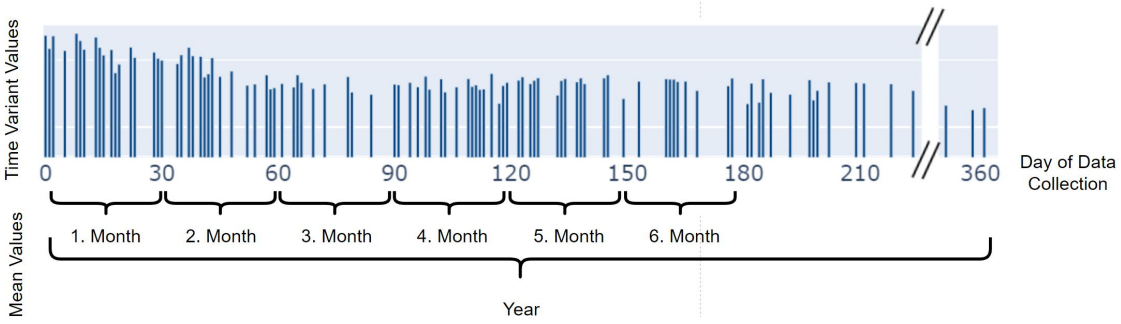

Figure 4.3: Time Variant Feature Engineering. The figure shows how sequential time variant data is transformed into time variant features. Seven features are created for each time variant data. One mean value for each of the first six months after transplantation, and one mean value for the whole year.

Algorithm 1: Create T-LSTM features. Concatenating text data and time variant data. Next, fill missing time variant data with last and next observation carries forward.

(T-LSTM) features = **Merge** texts with time variant data **On** date
features = **Merge** features with demographic data
for $Features$ **do**
    $date_i = featureDate$ ▷ 7 days last observation carried forward
    while $date_i <= featureDate - 7days$ **do**
        $date_i = date_i - 1\ day$
        Feature: Replace missing time variant data
    end while
    $date_i = featureDate$ ▷ 2 days next observation carried forward
    while $date_i <= featureDate + 2days$ **do**
        $date_i = date_i + 1\ day$
        Feature: Replace missing time variant data
    end while
    $date_i = featureDate - 7days$ ▷ Last observation carried forward
    while $date_i <= transplantationDate$ **do**
        $date_i = date_i - 1\ day$
        Feature: Replace missing time variant data
    end while
end for

4.5 Label Definition

Labels are used to annotate if samples belong to the positive class or negative class. Patients that undergo a kidney graft loss during the prediction time frame are labeled positive, and the remaining patients are labeled negative. For this task, we defined two time frames:

- **Short Term:** The model predicts that the patient will undergo a kidney graft loss within the next year. Further medical investigation should be done very soon.
- **Long Term:** The model predicts that the patient will undergo a kidney graft loss within the next five years. Further medical investigation should be done soon.

In dataset MO the positive class also includes patients that died during the prediction time frame. That reduces the data imbalance, and a finding of [Haldar 2020] is that it improves the outcome. The downside is that the prediction has less information, we don't know if the model predicts death or a kidney graft loss. Dataset MV doesn't include patients that die during the prediction time frame, thus having a higher data imbalance. Especially the class imbalance in short term would be very high; only 0.78% of the samples would belong to the positive class. In order to increase the amount of positive samples, we extend the short term time frame to one year. The extended short term time frame results in 1.3% positive class samples. Figure 4.4 shows the prediction time frames for dataset MV.

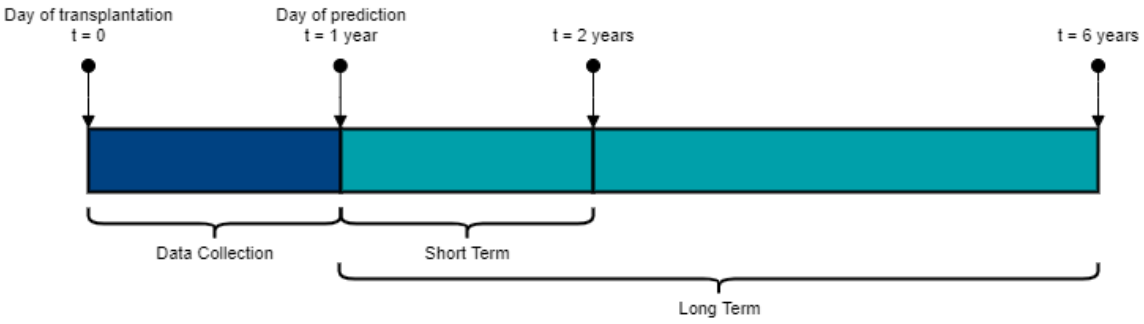

Figure 4.4: Data Collection and Prediction Time Frames for Dataset MV. The figure shows three time frames after the day of transplantation. The first is the year of data collection, the second is the short term prediction and the last is the long term prediction.

The resulting class distribution is shown in figure 4.5. The amount of negative samples in dataset MV is more than doubled compared to dataset MO. The positive class for short term prediction is increased by 61% due to the extended time frame and more data. The positive class for long term even decreased by 23%, because patients that die during the prediction time frame are not labeled as positive in dataset MV.

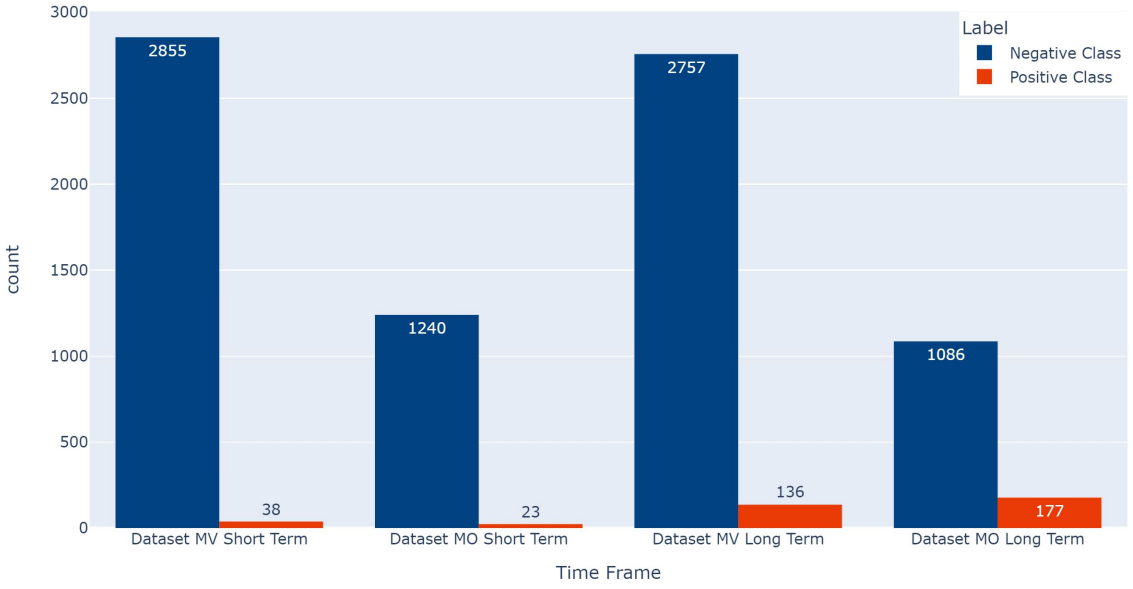

Figure 4.5: Class distribution for each dataset and time frame. The diagram shows multiple bars that display the distribution between the positive and negative class for each dataset and time frame.

4.6 Stratified Test Split and K-Fold Cross Validation

Dataset MO is separated into 70% training dataset and 30% test dataset. Samples are chosen randomly with an even distribution of labels and no cross validation.
For Dataset MV, we implement a stratified test split that splits the samples priority based. After ensuring that the labels are stratified, the samples are stratified for specific features with the following criteria.

Stratified Test Split Criteria: Label > Text count > Age > Dialysis Count

This non-random split has two benefits. First, each split is a good representation of the data and second, different models trained with this data always have the same samples in each split, thus the results are well comparable. Dataset MV is separated into 66.7% training data, 16.7% validation data and 16.7% test data.
This split is created in multiple steps. First, the samples are sorted after the stratified test split criteria. Next, a loop over the sorted list takes samples and puts four to the training set, one to the validation set and one to the test set. The loop continues until the list is empty. Afterwards, the validation and test set have 6 positive short term samples and 22 positive long term samples. This amount is not enough to create an evaluation that represents the general performance of the model. In order to improve the generalization, we add a 3-fold cross-validation, which generates three train/validation/test folds. The Validation dataset and the test dataset have different samples in each fold, thus the evaluation can be done with 50% of all samples. Figure 4.6 visualizes the 3-fold cross validation.

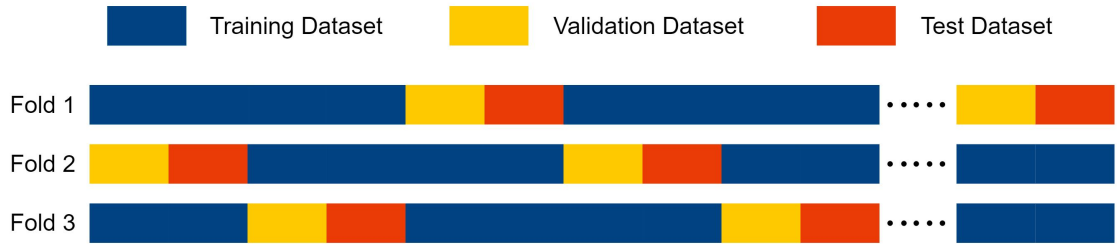

Figure 4.6: 3-Fold Cross-Validation. The figure shows how samples of dataset MV are divided between training dataset, validation dataset and test dataset. Each bar represents one fold of the 3-fold cross validation and has different samples in validation and test dataset. They are sorted by the stratified test split criteria.

4.7 Encoding, Scaling and Imputation

Encoding, Scaling and Imputation are three steps to transform the data in a format that is suitable for machine learning algorithms. For encoding, we use One-Hot-Encoding to transform the nominal demographic data into a binary data format.
Min-Max scaler is used to standardize the data. The scaler computes the minimum and maximum on the training dataset and transforms it. The minimum becomes 0, the maximum becomes 1 and all other values are placed in between. The same computation is then applied to the validation dataset and test dataset. The computation from the trainings set is used in order to prevent leaking information from these sets to the trainings set [Khanna 2020].
During imputation, missing values are replaced with substituted values. Missing values in training datasets are replaced with the mean value of the feature, and missing values in validation and test datasets are replaced with the value '-1'.

4.8 Down- and Oversampling

In order to handle data imbalance on a data centric level, we implement downsampling and oversampling. For downsampling, we add a function that divides the samples of the negative class. The function takes an integer parameter that acts as divisor for the training dataset (e.g., if the parameter has a value of '4', every fourth sample remains in the training dataset). The parameter for downsampling-rate will be called 'DS-rate' from now on. DS has two exceptions: The value of '0' deactivates downsampling, and the value of '1' divides the negative class by 1.5 (taking 2 out of 3 samples). Figure 4.7 displays included and excluded training data for the DS-rates '0', '1', '2' and '4'. The downsampling is automatically stratified, since the samples are sorted by the stratified test split shown in section 4.6.
Oversampling is achieved by implementing SMOTE [Chawla et al. 2002]. SMOTE creates synthetic samples of the positive class to create a balanced dataset.

Figure 4.7: Downsampling for multiple DS-rates. The figures shows how DS removes samples from the negative class of the training dataset. DS 0 includes all samples, DS 1 removes every third sample, DS 2 includes every second sample and DS 4 includes every fourth sample.

4.9 Summary

The chapter explains how a data centric approach creates a new data baseline. The preprocessing for this baseline is newly set up including more data (1263 to 2893 KTR), new cohort selection, data cleaning, feature engineering, stratified test split, 3-fold cross-validation, encoding, scaling, imputation, down- and oversampling. The preprocessing for the text data in dataset MV is based on the preprocessing of dataset MO with the improvement, that it loses less information during data cleaning (see replace small texts in Section 4.3).
Furthermore, we redefine the labels. Dataset MV only predicts kidney graft loss, while dataset MO predicts kidney graft loss or death. This change makes the prediction more accurate, but increases the data imbalance. The time frame for the short term prediction in dataset MV is extended to one year, in order to increase the positive samples from 0.78% to 1.3%.

Chapter 5

Model Centric Approach

This chapter is about methodologies used to improve and create the ML models. The first section compares different ML evaluation metrics in order to find the best one for the task's objective from a medical perspective. The section Machine Learning Models explains the hyperparameters of the different ML models, the data that they are using and how different models are combined in order to achieve the best outcome.

5.1 Define Model Objective

To evaluate a binary classification, the objective of the model has to be defined. There are several evaluation metrics that are used to calculate a score using the predictions or probabilities from the models' outcome. [Reuter 2021] and [Islam 2021] choose F1 as the main metric to maximize, since it is widely known and represents a harmonic mean between precision and recall. However, [Chicco und Jurman 2020] show that F1 can be a misleading metric, when working with unbalanced data. The positive class is underrepresented, but from a medical perspective, predicting the positive class correctly is more important than correctly predicting the negative class.
An incorrect predicted sample of the negative class would lead to a patient classified with a higher risk, followed by a higher medical care and thus higher costs. A misclassified positive sample on the other hand, could lead to a reduced medical care for a patient in need. This could lead to a kidney graft loss, the patients' life quality would decrease significantly, since they would rely on dialysis, which is also very costly.
In order to focus on predicting the positive class correctly, the model needs to maximize an evaluation metric that is sensitive to this class, even with unbalanced data. To choose this metric, the MLP tabular model is optimized for different metrics. Afterwards, a discussion about the resulting confusion matrix leads to the chosen evaluation metric for the model objective. These tests are done with following metrics:

- **F1-Score**: The model objective of [Reuter 2021] and [Islam 2021]
- **MCC**: The best metric for imbalanced learning according to [Chicco und Jurman 2020]
- **Balanced Accuracy**: A metric independent of class imbalance
- **PR-AUC**: The AUC value of a precision-recall curve for different probability thresholds.
- **PR-Curve with fix threshold**: Evaluates all thresholds of the precision-recall curve in order to maximize one of the following metrics:
  - **PR-Curve with threshold for best F1-Score**
  - **PR-Curve with threshold for best MCC**
  - **PR-Curve with threshold for best Balanced Accuracy**

5.2 Machine Learning Models

Every ML model has parameters that are learned during the training and hyperparameters that can be defined before training. Hyperparameter optimization (HPO) tries to find the optimal value for each hyperparameter in order to maximize the target metric. To do so, we define the possible values beforehand. Another step of preparation is to choose the data being used for the ML model. For MLP, we create multiple baselines including different parts of data. LR, SVM, RF and XGBoost use demographic data and optional time variant data. A stacked ensemble combines all Ml models to create the final baseline. Figure 5.1 shows which models use what kind of data and how the outputs are used to create multimodal baselines. The results for the baselines are shown in Section 7.1.
The main technique to handle data imbalance with a model centric approach with LR, SVM, RF and XGBoost is to optimize the threshold during HPO. For MLP, different loss functions return a higher loss for the positive class.

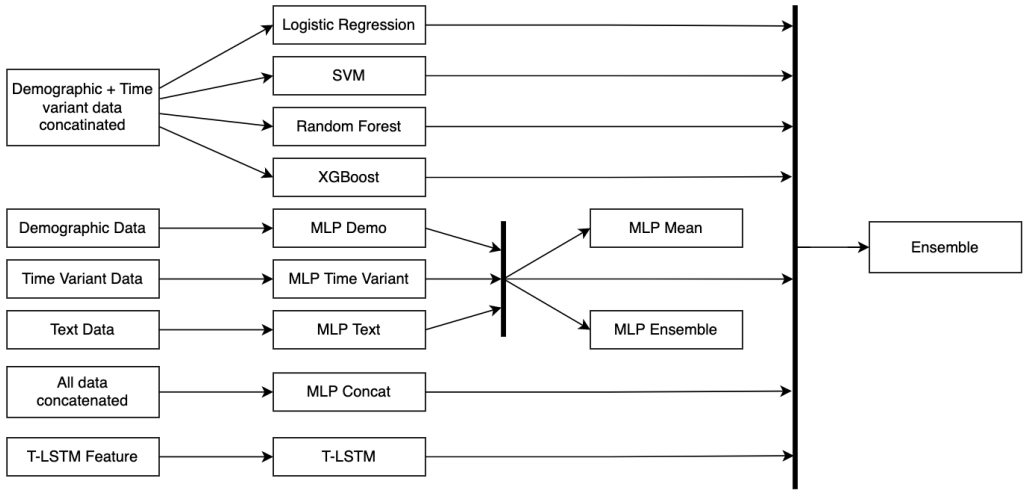

Figure 5.1: Model Centric Approach. The figure shows all baselines which are created using the model centric approach. The boxes on the left display the data that is used by the ML models. The outcome of MLP demo, time variant and text is concatenated to create an MLP Mean and an MLP Ensemble baseline. The outcome of all baselines, but MLP Mean and MLP Ensemble is concatenated and creates the input for the final ensemble model.

5.2.1 Logistic Regression

LR creates the first baseline. It has a rather simple setup, using three hyperparameters in addition to threshold and date centric parameters. Penalty and solver are fixed and L1-ration is optimized by HPO. Table 5.1 describes the hyperparameters.

Hyperparameter	Description
penalty	'elasticnet' is picked as it is able to support both, L1 and L2 penalty terms.
solver	'sage' is picked as it is the only solver that supports elasticnet.
L1-ratio	HPO optimzes a ratio between L1 and L2.
c	The inverse of regularization strength

Table 5.1: LR Hyperparameters. The table shows LR hyperparameters and their descriptions. Penalty and solver are fix and L1-ration and c are optimized during HPO.

5.2.2 Support Vector Machine

The next baseline is SVM. It creates quick results as it requires only few computational power [Gandhi 2018]. Depending on the kernel, it uses different algorithms for classification. Threshold, data centric parameters and the hyperparameters shown in 5.2 are optimized by HPO. Remaining SVM hyperparameters use the default value.

Hyperparameter	Description
kernel	HPO chooses between the kernels linear kernel (linear), polynomial kernel (poly), radial basis function (rbf) and sigmoid kernel (sigmoid).
c	The inverse of regularization strength; The penalty is a squared l2 penalty.
shrinking	HPO can activate 'shrinking' to improve computation time by temporary leaving out weak features.

Table 5.2: SVM Hyperparameters. The table shows all SVM hyperparameters that optimized during HPO and their descriptions.

5.2.3 Random Forest

RF is a model that was also used with dataset MO, thus it creates a good baseline to compare the data preprocessing of dataset M and dataset MV. A big advantage of RF is that it has a high explainability, which we will use to create an overview about the feature importance. All RF hyperparameters are either optimized by HPO or use the default value. Table 5.3 shows the hyperparameters that are optimized by HPO:

Hyperparameter	Description
criterion	Categorical hyperparameter that chooses between the loss functions 'gini', 'entropy' and 'log_loss'.
bootstrap	Boolean hyperparameter that determines, if a tree uses a random part of the dataset or the whole one.
max_depth	Maximum depth of the tree
min_samples_split	Minimum number of samples required to split an internal node
min_samples_leaf	Minimum number of samples required to be at a leaf node
max_features	Categorical hyperparameter that defines if the amount of features considered when looking for the best split. Can be the square root 'sqrt' or binary logarithm 'log2' of all features.
n_estimators	Defines the number of trees created before calculating the outcome.

Table 5.3: RF Hyperparameters. The table shows all RF Hyperparameters that are optimized during HPO and their descriptions.

5.2.4 XGBoost

XGBoost is a decision tree based algorithm that can be considered as best-in-class for small- to medium-sized structured data [Morde 2019]. The ability to learn the best missing values depending on training loss should come in handy for the demographic data.
The general good performance of XGBoost does not automatically make it the most suited one. Due to the complexity of XGBoost it has a lot of hyperparameters shown in table 5.4.

Hyperparameter	Description
objective	Learning task for the corresponding learning objective The HPO chooses between logistic regression (binary:logistic) or hinge loss (binary:hinge), since our task is binary classification.
learning_rate	Shrinks the boosting effect on feature weights after each step This shrinkage prevents overfitting and results in a more conservative boosting process.
max_depth	Maximum depth of the tree
subsample	Subsample ratio of features being used for each iteration
colsample_bylevel	Subsample ratio of features being used for each tree
colsample_bytree	Subsample ratio of features being used for every new depth level reached in a tree
n_estimators	Number of trees created before calculating the outcome
alpha	L1 regularization
lambda	L2 regularization
rate_drop	Probability of dropping a fraction of previous trees during dropout
skip_drop	Probability of skipping the dropout procedure

Table 5.4: XGBoost Hyperparameters. The table shows all XGBoost Hyperparameters that are optimized during HPO and their descriptions.

5.2.5 Multilayer Perceptron

The ANN MLP usually requires huge amount of data that neither dataset MA nor dataset MV have. Still, [Reuter 2021] and [Islam 2021] achieved the best result for dataset MO using MLP models. The reason is, that they included unstructured text and MLP models can perform better with unstructured data compared to other ML techniques [Mathew et al. 2020].
Table 5.5 shows all hyperparameters that are optimized with HPO. Optimizer is excluded from the table, because the optimizer 'AdamW' always achieved the best performance. Other optimizer being tested are 'SGD' and 'Adam'.
For the loss function we try Binary Cross Entropy, Focal Loss and Hinge Loss. Since the loss functions have different hyperparameters and the HPO does not support dynamic tuning (tuning of hyperparameters depending on other hyperparameters), the loss functions are tested separately. Hinge Loss is not able to predict the positive class very well, because it misses a hyperparameter to multiply the loss of the positive class. Focal loss performs very differently on the folds of the 3-fold cross-validation, which results in a bad performance. We finally choose Binary Cross Entropy, because the hyperparameter 'pos_weight' improved the prediction of positive class a lot.

Hyperparameter	Description
learning_rate	Float hyperparameter that defines the step size at each iteration to minimize the loss.
layers	Number of hidden layers
dropout	Number of layer outputs are randomly ignored during training
batch_size	Number of training samples to work through before the model updates parameters
nb_units	Number of perceptrons per Layer
wrs	Boolean, true activates WRS
pos_weigt	Integer that multiplies the loss on the positive class

Table 5.5: MLP Hyperparameters. The table shows all MLP hyperparameters that optimized during HPO and their descriptions.

With the MLP model, we first create multiple baselines with each kind of data. The goal is to create an ablation study and compare the performance of demographic, time variant and text data.

single-modal baselines

- MLP Demographic
- MLP Time Variant
- MLP Text

Next, we utilize all data and fuse it to multi-modal MLP models. The first multi-modal baseline is MLP Mean. It calculates the mean value out of all single-modal baselines. This technique is a late fusion approach, since the fusion is the last step before creating the predictions.
For the concatenation baseline demographic, time-variant and the text feature vectors are concatenated which results in a vector of 943 features for each patient. This is an early fusion approach, because the fusion happens before the training.
The baseline MLP Ensemble is a stacked ensemble that uses an SVM model. Stacked ensemble is a late fusion technique that uses the probabilities of
the single-modal baselines as an input vector and has the same hyperparameters as the SVM model in table 5.2.

Multi-Modal baselines

- MLP Mean
- MLP Concatenation
- MLP Ensemble

5.2.6 T-LSTM

T-LSTM is a model that is able to work with sequential data. The data is combined in an early fusion shown in Section 4.4. Processing sequential data has the advantage, that the model can recognize peaks and rises in time variant data. Especially fluctuations in creatine are very important, since they are the indicator for an AKI event and AKIs are an important indicator for kidney health [Cruz et al. 2007]. We use T-LSTM instead of LSTM, because the time variant data and text data appear in irregular time intervals.
Table 5.6 shows the hyperparameter that are optimized during HPO. The loss functions from the original model has been changed from softmax cross entropy to weighted cross entropy, because weighted cross entropy can handle imbalanced data with the hyperparameter 'pos_weight'.

Hyperparameter	Description
learning_rate	Float hyperparameter that defines the step size at each iteration to minimize the loss.
dropout	Number of layer outputs are randomly ignored during training
hidden_dim	Number of perceptrons per hidden layer
fc_dim	Number of perceptrons in the last hidden layer
pos_weight	Integer that multiplies the loss on the positive class

Table 5.6: T-LSTM Hyperparameters. The table shows all T-LSTM hyperparameters that optimized during HPO and their descriptions.

5.2.7 Stacked Ensemble

The baseline Stacked Ensemble combines all ML models shown in this chapter in order to create the final baseline. It uses a SVM model with the hyperparameters from Table 5.2.

5.3 Summary

The chapter is about the methodologies used for the model centric approach. It presents each ML model, its fusion method and its hyperparameters. We conduct multiple experiments with the MLP model: MLP Demographic, MLP Time Variant and MLP Text are single data baselines and MLP Concatenate, MLP Mean and MLP Ensemble are multi-modal baselines. Other conducted ML models are LR, SVM, XGBoost and T-LSTM. Finally, a stacked ensemble combines the strengths of the all conducted ML models to our final model.

Chapter 6

Implementation

This chapter describes the environment, including hardware and dependencies, that have been used for the task. Furthermore, it shows how the techniques early stopping, k-fold cross validation and HPO are implemented.

6.1 Environment

Computations were performed on "HPC @Charité", the central computing environment of the Charité – Universitätsmedizin Berlin. The HPC @Charité is designed and operated by the central division IT of the Charité – Universitätsmedizin Berlin. It includes >3000 AMD CPU cores, >20TB of RAM, 45 NVIDIA A100 GPUs and 500TB of shared storage. The compute nodes run Rocky Linux 8.5, for NVIDIA GPUs driver version 470 and CUDA version 11.4 are used. Computational jobs are managed with the SLURM batch system.
The HPC @Charité is available for use to all research groups of the Charité – Universitätsmedizin Berlin, the Berlin Institute of Health, and their collaboration partners.

6.2 Software Packages

We used Python 3.9 as the main programming language and Conda as a packaging tool. Numpy 1.22.1 and Pandas 1.4.1 are used for data processing. We use PyTorch 1.10.2 to create the MLP models. LR, SVM and RF are imported from scikit-learn 1.0.2. XGBoost is imported from XGBoost 1.5.2. The T-LSTM is created with Tensorflow 1.14. SMOTE is imported from imbalanced-learn 0.9.0. HPO is created with Optuna 2.10.1. Lastly, the figures are created with plotly 5.7.0 or draw.io 20.0.3.

6.3 Early Stopping

Early stopping is used during training of MLP models in order to optimize the amount of trained epochs. The model evaluates the validation set after every epoch. Early stopping halts the training and picks the best model if the objective did not improve on the validation dataset for 15 epochs. If the model would continue to train it would overfit on the training dataset.

6.4 K-fold Cross-Validation

A 3-fold cross validation is implemented to produce a better estimation of the skill. A loop repeats the training three times, using different samples in training, validation, and test dataset each time. The samples are not shuffled randomly, but split stratified as shown in Section 4.4. Results that are shown throughout the thesis represent the mean values of all splits. Results shown

in Section 7.1.2 and Section 7.1.3 additionally show the variance of the three folds. Confusion matrices that are shown throughout the thesis represent the sum of all splits.

6.5 Hyper-Parameter Optimization

The HPO is implemented using the open source framework Optuna [Akiba et al. 2019]. Optuna creates a study to find the optimal set of parameters for each model. The study requires an objective to maximize or minimize and a set of hyperparameters to chose from. The objective is returned from each training that is wrapped by a function that returns the defined model objective. The available hyperparameter options are described in Section 5.2 and the picked values are shown in then Appendix A. Each study trains for 200 trials, which results in 600 trainings overall, including the 3-fold cross-validation. All other parameters for the study are set to default. The hyperparameter options have been adapted during the experiments. The ranges are extended, if the HPO picks a value that is the limit of the range. When the HPO always picks the same value, we exclude the hyperparameter and set it to a fixed value.
HPO uses the boolean 'SMOTE' to either activate (True) or deactivate SMOTE (False) with default parameters. It uses the boolean 'drop_lab' to exclude the time variant data (True) or include it (False).

6.6 Summary

The chapter shows the computing environment HPC @Charite, used software packages and the implementation of early stopping, 3-fold cross-validation and HPO.

Chapter 7

Evaluation

This chapter presents the results of all experiments that have been conducted. Section 7.1 shows the results for all models and from a quantitative perspective. Section 7.2 evaluates the results from a qualitative perspective. Section 7.3 discusses the results and mistakes that have been made. Finally, Section 7.4 summarizes this section.

7.1 Results and Quantitative Error Analysis

The section first evaluates the results from the model objective experiments. Next, it presents the results for the different MLP model baselines followed by the baselines for all ML models. Last, the section evaluates the results of the data centric approach.

7.1.1 Model Objective

In order to find the most suitable model objective, the model MLP Demographic has been optimized for different evaluation metrics. The HPO uses the model objective to select the combination of hyperparameters that achieved the highest value of the chosen metric. Early stopping uses this metric to stop the training once the chosen metric stops improving. Table 7.1.1 shows the outcome for each evaluation metric. The goal is to find a metric that achieves a good recall and no or a small drop in TNR, in order to improve prediction on the positive class.

Maximized Metric	BAcc	Recall	TNR	Acc	AUC	PR AUC
BAcc	0.527	**0.397**	0.657	0.645	0.562	0.059
F1	0.484	0.030	0.938	0.896	0,543	0.049
MCC	0.520	0.119	0.921	0.883	0.576	0.061
PR_AUC	0.471	0.148	0.794	0.765	0.500	0.041
PR BAcc	**0.547**	0.384	0.710	0.695	0.569	**0.075**
PR F1	0.509	0.061	0.957	0.915	0.575	0.064
PR MCC	0,501	0,015	**0.987**	**0.941**	**0,595**	0,059

Table 7.1: The table presents the results for the MLP Demographic optimized for different evaluation metrics. The column 'Maximized Metric' shows which evaluation metric has been maximized. The other columns show the results for the test dataset after each tuning and training.

F1, PR F1, MCC and PR MCC have strong TNR predicting more than 92% of all negative samples, but less than 12% of positive samples. PR_AUC has a better balance between recall and TNR, but achieves the worst BAcc. Maximizing BAcc has the highest recall of 0.397 followed by PR BAcc with 0.384. Their good recall results come with a lower but still acceptable TNR.
We decide to optimize the models for BAcc, since it achieves the highest recall.

7.1.2 Multilayer Perceptron Models

This section compares all conducted MLP experiments. This includes the three single data source baselines MLP Demographic, MLP Time Variant and MLP Text and the multimodal baselines MLP Concatenation, MLP Mean and MLP Ensemble. Results for training and validation data can be found in Appendix A.
The long term prediction results for MLP are listed in Table 7.2. MLP Text is the best single data baseline, but it can not outperform the multi-modal baselines. It has a higher recall than BAcc, while MLP Demographic and MLP Time Variant achieved a higher BAcc than recall. Therefore, MLP Text predicts the positive class better, while the other single data baselines predict the negative class better. The baseline MLP Ensemble has the best performance, achieving a BAcc of 0.78. Having a recall of 0.7, MLP Ensemble predicts 70% of positive samples and 86% of negative samples correctly. MLP Demographic and MLP Time Variant do also overfit on the training and validation data, but not as much as in short term. Thus, the ensemble baseline outperforms the text data baseline in long term prediction.

	BAcc	Recall	Precision	F1	MCC	AUC	PR AUC	Var(BAcc)
Demo.	0.56	0.53	0.51	0.41	0.05	0.57	0.07	**0.009**
Time Var.	0.54	0.40	0.51	0.45	0.04	0.56	0.08	0.042
Text Data	0.64	0.70	0.53	0.43	0.13	0.67	0.08	0.080
Concat.	0.57	0.50	0.52	0.44	0.06	0.56	0.06	0.051
Mean	0.74	**0.89**	0.54	0.45	0.20	0.84	**0.30**	0.029
Ensemble	**0.78**	0.70	**0.59**	**0.61**	**0.31**	**0.84**	0.26	0.049

Table 7.2: The table shows the results for the MLP models that are trained for the long term task. MLP Demographic has been shortened to Demo., MLP Time Variant to Time Var., and MLP Concatenated data to Concat.

Table 7.3 shows the short term prediction results of the MLP models. MLP Text achieves a BAcc of 0.68, which is not only the best single data source baseline, but also outperforms the multi-modal baselines. The ensemble is the best multi-modal baseline with a BAcc of 0.63. The confusion matrix for the ensemble baselines is shown in Figure 7.1. MLP Text has a recall of 0.69 which just above BAcc. All other baselines have a much lower recall than BAcc. Thus, MLP Text predicts positive and negative class close to equally and the other baselines predict the negative class better. MLP Concatenation achieves a BAcc of 0.56, therefore it is the baseline with the worst outcome.

	BAcc	Recall	Precision	F1	MCC	AUC	PR AUC	Var(BAcc)
Demo.	0.59	0.41	0.51	0.46	0.05	0.69	0.03	0.002
Time Var.	0.58	0.36	0.51	0.46	0.05	0.60	0.05	0.006
Text Data	**0.68**	**0.69**	0.51	0.43	**0.09**	**0.70**	0.05	0.001
Concat.	0.56	0.36	0.51	0.45	0.04	0.60	0.02	**0.000**
Mean	0.61	0.36	0.51	**0.49**	0.07	0.68	**0.10**	0.007
Ensemble	0.63	0.42	0.51	0.48	0.08	0.68	0.09	0.001

Table 7.3: The table shows the results for the MLP models that are trained for the short term task. MLP Demographic has been shortened to Demo., MLP Time Variant to Time Var., and MLP Concatenated data to Concat.

The confusion matrix for the baseline MLP Ensemble is shown in Figure 7.1. Both predict the majority of negative samples correctly, but only the model trained for long term prediction can predict most positive samples correctly, as well.

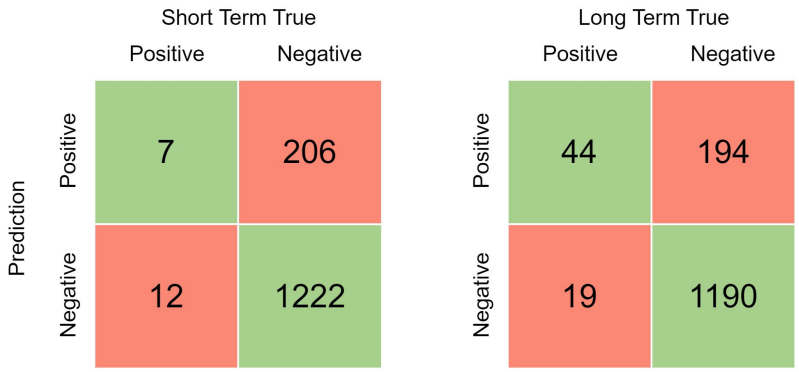

Figure 7.1: Confusion Matrix MLP Ensemble. The figure shows the confusion matrix for the baseline MLP Ensemble. The left matrix shows predictions for the short term task and the right one shows the predictions for the long term task. The data represents the sum of each test dataset from the 3-fold cross validation.

7.1.3 Model Centric Approach

This section compares all conducted ML models. **Table 7.4 presents the results for the long term prediction task.** The baseline Stacked Ensemble is by far the best one. A BAcc of 0.86 and a recall of 0.83 show that the model is able to predict 83% of positive samples and 89% of negative samples correctly. The confusion matrix for MLP Ensemble is shown in Figure 7.2. LR and SVM have a underwhelming BAcc just above 0.5, thus they perform just slightly better compared to random predictions. Both models score a BAcc on the training data of at least 0.7.RF has better results with a BAcc of 0.64, but only has Recall of 0.5, hence it only predicted half of the positive samples correctly. XGBoost scores a BAcc of 0.7, therefore it is the best ML model that is not using text data. The HPO for T-LSTM completed only 12 trials for long term prediction, thus it was not able to find an optimal set of hyperparameters. T-LSTM takes by far the longest time for each HPO trial, with approximately three to five hours per run.

	BAcc	Recall	Precision	F1	MCC	AUC	PR AUC	Var(BAcc)
LR	0.51	0.49	0.50	0.38	0.01	0.53	0.05	0.005
SVM	0.52	0.53	0.50	0.37	0.01	0.52	0.06	0.040
RF	0.64	0.5	0.53	0.51	0.14	0.74	0.14	0.039
XGBoost	0.70	0.63	0.55	0.53	0.2	0.70	0.39	0.012
MLP Ensem.	0.78	0.70	0.59	0.61	0.31	0.84	0.26	0.049
T-LSTM	0.59	0.54	0.52	0.45	0.08	0.63	0.09	0.042
Ensemble	**0.86**	**0.83**	**0.62**	**0.66**	**0.42**	**0.93**	**0.66**	0.042

Table 7.4: The table shows the results for the conducted ML models in long term prediction. MLP is represented by the best performing MLP baseline for the long term task which is MLP Ensemble. MLP Ensemble is shortened to MLP Ensem.

The final results for the short term prediction task are listed in Table 7.4. RF achieved a BAcc of 0.75 which is the best result for short therm prediction. This is unexpected, since RF does not use the text data and the MLP text data baseline has the best result comparing the

MLP models. The feature importance for the ensemble is shown in Figure 7.8. RF has by far the highest importance, which lines up with a BAcc of 0.95 for the training dataset. Even if The baseline Stacked Ensemble combines the predctions of all models it only achieves the third best BAcc score. A possible reason is shown in Subsection 7.2.4. LR and SVM have a bad performance. LR only predicts 21% and SVM just 16% of kidney graft loss correctly. Due to time constrains, the T-LSTM HPO only completed 18 trials instead of 200. A BAcc of 0.69 is a good result considering the uncompleted HPO.

	BAcc	Recall	Precision	F1	MCC	AUC	PR AUC	Var(BAcc)
LR	0.55	0.21	0.51	0.49	0.04	0.66	0.02	0.005
SVM	0.55	0.16	0.51	0.51	0.05	0.62	0.02	0.000
RF	**0.75**	0.57	0.54	**0.56**	**0.20**	**0.92**	**0.10**	0.007
XGBoost	0.66	0.37	0.54	0.55	0.15	0.70	0.06	0.003
MLP Text	0.68	**0.69**	0.51	0.43	0.09	0.70	0.05	0.001
T-LSTM	0.69	0.52	0.52	0.5	0.12	0.76	0.07	0.001
Ensemble	0.68	0.47	0.52	0.52	0.14	0.85	0.06	0.002

Table 7.5: The table shows the results for the conducted ML models in short term prediction. MLP is represented by the best performing MLP baseline for the short term task which is MLP Text.

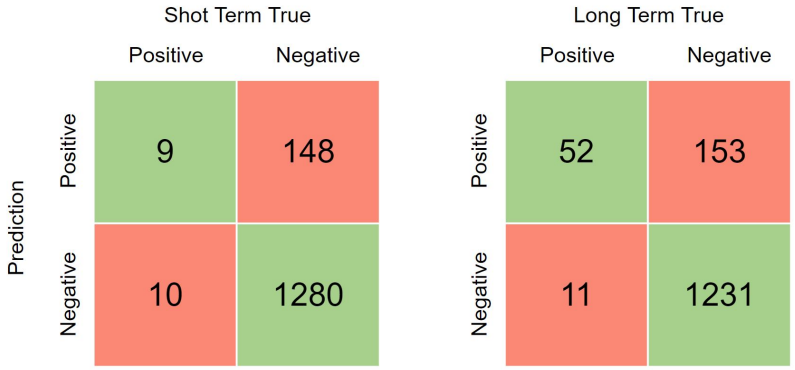

Figure 7.2: Confusion Matrix Ensemble Model. The figure shows the confusion matrix for the stacked ensemble model. The left matrix shows the predictions for the short term task and the right one shows the predictions for the long term task. The data represents the sum of each test dataset of the 3-fold cross validation.

7.1.4 Data Centric Approach

This section compares the data centric approach of dataset MO and dataset MV. It is important to note that this is not a comparison between preprocessings. The labels have been redefined in Section 4.5, dataset MA also includes patients that die during the prediction time frame. Furthermore, we added 3-fold cross-validation in Section 4.6, which greatly increases the generalization estimate. Thus, an improved preprocessing does not necessary lead to better results. A deeper analysis can be found in the qualitative error analysis.

Table 7.6 compares the datasets on demographic and time variant data using the random forest model. MV is better in all metrics for the short term task, especially recall and AUC. Dataset MV has a better recall for the long term task, but the main metric BAcc (see Section 7.1.1 for more

details to the metric pick) is worse. The data centric approach improved BAcc by 36% for the short term task and decreased it by 10% for the long term task.
Table 7.7 compares the datasets on text data using the MLP text model. BAcc decreased by 9% for the short term task and 4% for the long term task. Dataset MV improved the recall by 35% for the long term task.

Dataset		BAcc	Recall	Precision	F1	MCC	AUC
ST	MO	0.55	0.16	0.52	0.53	0.06	0.55
	MV	0.75	0.57	0.54	0.56	0.20	0.92
LT	MO	0.71	0.35	0.61	0.61	0.30	0.71
	MV	0.64	0.50	0.53	0.51	0.14	0.74

Table 7.6: The table compares the random forest results from dataset MA and dataset MV. The ST rows show the results for the short term task and the LT rows for the long term task.

Dataset		BAcc	Recall	Precision	F1	MCC	AUC
ST	MO	0.75	0.85	0.52	0.43	0.14	0.75
	MV	0.68	0.69	0.51	0.43	0.09	0.70
LT	MO	0.67	0.52	0.61	0.63	0.28	0.67
	MV	0.64	0.70	0.53	0.43	0.13	0.67

Table 7.7: The table compares the MLP text results from dataset MA and dataset MV. The ST rows show the results for the short term task and the LT rows for the long term task.

7.2 Qualitative Error Analysis

This section analyses the results from a qualitative perspective. First, we compare the performance of the stacked ensemble between the short term and the long term prediction task. Next, the section compares the quality of the datasets MO and MV. We analyze the performance of the MLP Concatenation model in Subsection 7.2.3. The next subsection shows the connection between ensemble importances and the generalization problem of the ensemble in short term prediction. Last, we analyze importances from the random forest model.

7.2.1 Task Comparison

The quantitative results in Section 7.1 show that models trained for the long term task generally achieves better results than ones trained for the short term task. Figure 7.3 visualizes this by comparing the results for the stacked ensemble model. The figure shows that both models are equally good at predicting the negative class, but the model trained for the short term task is worse in predicting the positive class. The main reason is the class imbalance, short term has only 1.3% positive samples, while long term has 4.7%.

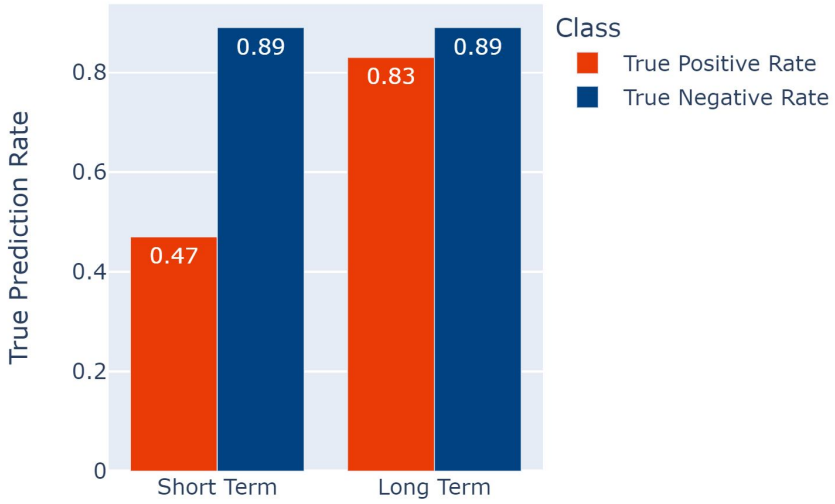

Figure 7.3: Class Comparison. The figure shows the results for the stacked ensemble model divided by task and class. The main finding is that both models predict the negative class well, but the model trained for short term prediction has a weak performance in predicting the positive class.

What neither, Figure 7.3 nor the quantitative results show, is a direct comparison of the samples for short term prediction. A reason for the worse performance of short term prediction task could be that positive samples in this time frame are more difficult to predict. In order to evaluate this, we analyze the positive samples of the stacked ensemble model based on the time between the transplantation and graft loss with Figure 7.4. There are 20 positive samples in the short term time frame (bins 1.0 and 1.5). The model trained for the long term task is able to predict 16 correctly, while the model trained for the short term task only predicts 9 correctly. The first bin contains 11 patients undergoing graft loss. The model optimized for long term prediction predicts 9 correctly, while the model for short term only predicts 5 correctly. The next bin contains patients that undergo a graft loss from 1.5 years after their transplantation until 2 years after their transplantation. The model for the long term task predicted 7 out of 8 correctly, and the model for the short term task only 4 out of 8. Aggregated, long term predicted 16 out of 20 and short term 9 out of 20. Thus, the model optimized for long term prediction is clearly better in predicting samples in the short term time frame.

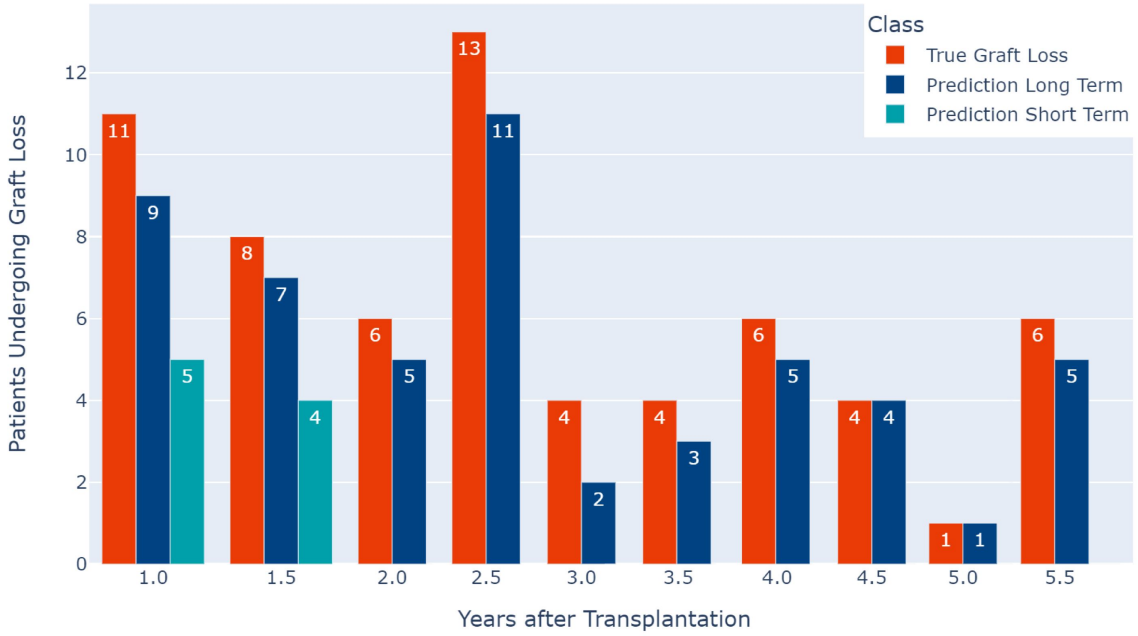

Figure 7.4: The figure shows the positive predictions of the stacked ensemble compared to the true graft loss label. The samples are clustered by years between transplantation and graft loss. Each bin represents a time span of half a year, resulting in 10 bins. The year on the x-axis represents the start of the time frame. The first two time frames include short term prediction and long term prediction. The remaining eight only include long term prediction. The main finding is that the model trained for long term prediction also predicts the samples in the first year better than the model trained for short term prediction.

7.2.2 Data Quality

With the data centric approach, we increased the amount of samples from 1263 in dataset MO to 2893 in dataset MV. This is due to including patients that underwent their kidney transplantation at the hospital in Virchow. Their EHR are excluded in dataset MO, because they are expected to have a worse data quality.
Figure 7.5 compares the BAcc of EHR from location Mitte with those of location Virchow. In both, short and long term prediction tasks, the EHR of Mitte achieves better results than Virchow. For long term only slightly, but in short term there is a big difference of about 0.208 less BAcc. The difference mainly results from samples of the positive class. The model predicted 7 out of 11 positive samples from Mitte and only 2 out of 8 from Virchow correctly. Overall, the figure indicates that there is a difference in data quality between EHR of Virchow and Mitte. But the difference in long term prediction is very small, thus we can assume that the quality can be easily balanced with enough samples in the positive class.

The comparison of MLP models in Section 7.1.2 shows that the text data creates the best performing single data baseline. The number of texts per patient varies greatly a and usually more data increases the performance of a ML model. Figure 7.6 explores the relation between days during the data collection time frame that contain texts and BAcc of the stacked ensemble model. The figure does not show a correlation between the conducted data, thus the amount of texts does not increase the model performance.

Figure 7.5: Stacked Ensemble BAcc by Location. The figure shows the BAcc for the stacked ensemble model. The results are divided by task and location. The main finding is that patients that underwent their kidney transplantation in Virchow are more difficult to predict in short term.

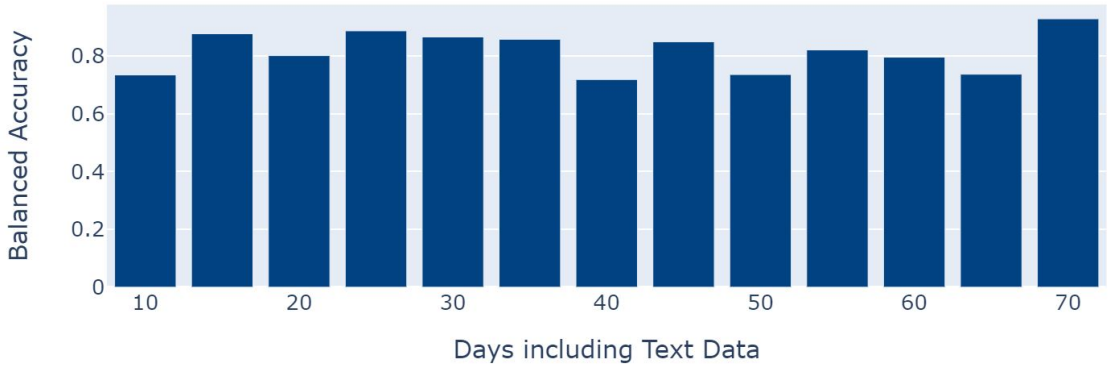

Figure 7.6: Stacked Ensemble Model BAcc by Days Including Text Data. The figure shows the BAcc for the stacked ensemble model. The results are clustered by the amount of days including text data each patient has. The value on the x-aches shows the start for the range of each bin. The main finding is that there is no correlation between the amount of texts and model performance.

7.2.3 Concatenation Model

The concatenated data baseline performs worse than expected for both, short and long term prediction tasks. A higher expectation bases on the models' ability to learn context between demographic, time variant and text data. A possible reason for the bad performance is that concatenation is an early fusion approach that forces the model to use the same training duration and hyperparameters for all features. For example, the MLP Text training has a duration of more than 100 epochs and the other MLP models with a single data source just around 15 to 30 epochs. In contrast, the concatenated model trains for about 50 epochs, which is not optimal, neither for text nor for numerical features.

For further investigation, we conducted an experiment with the MLP model. The model uses demographic data and a hyperparameter, that enables the HPO to choose to concatenate the time variant data. The model achieves better results without time variant data, further indicating that the concatenation does not perform well as a data fusion method.

7.2.4 Ensemble Importances and Generalization

The baseline Stacked Ensemble achieves worse results in the short term prediction task than some of the baselines it uses as input. The MLP ensemble gets outperformed by MLP Text (BAcc of 0.63 vs. 0.68) and outperformed by RF (BAcc of 0.68 vs. 0.75). A possible reason for this behavior can be found by comparing the feature importances. The feature importances for MLP Ensemble trained for the short term task are shown in Figure 7.7 and for Stacked Ensemble in 7.8. The MLP ensemble model puts a higher importance on demographic and time variant data than to text data. This can be a result of a correlation between the importances and the performance on the training dataset. Demographic data achieves a BAcc of 0.85 on the training dataset, time variant data achieves 0.82 and text data just 0.66. Since the MLP Text model has a similar performance on the training dataset and test dataset, it does generalize well. The results for each split are shown in tables A.10, A.12 and A.14 for demographic, time variant and text data respectively.

Figure 7.7: Importances MLP Ensemble Short Term Task. The figure shows the feature importances for the SVM model that creates the baseline MLP Ensemble for short term prediction. This data is not available for long term prediction, because the SVM model for MLP Ensemble trains with a polynomial kernel in this case, which does not support feature importance.

What seems to happen, is that the ensemble MLP model learns the generalization error from MLP Demographic and MLP Time Variant. Table 7.8 gives an overview about the estimation of the generalization error of MLP Ensemble. The model optimized for the short term task has a big difference of recall between training and test splits, thus it has a generalization error. The generalization error becomes very small when we optimize the model for long term prediction (LT Recall). This is most likely due to the difference of class imbalance between the short and long term prediction tasks. The model has no generalization error on the negative class.

	ST Recall	ST TNR	LT Recall	LT TNR
Training	0.99	0.81	0.72	0.86
Validation	0.64	0.84	0.72	0.84
Test	0.42	0.84	0.70	0.86

Table 7.8: MLP Ensemble generalization. The table shows the results for each class of the MLP ensemble Model for each Split. ST Recall indicates that the model trained for short term prediction does not generalize well for prediction of the positive class. All other classes seem to generalize well.

This behavior can also be found in the stacked ensemble, but more input variables make it more complex to analyze. Figure 7.8 shows the importances of the stacked ensemble model for the short and long term tasks. This data is available, since the HPO chooses a linear kernel for both classes. For the short term prediction task, RF has by far the highest importance, which does correlate with the outcome. But LR does not correlate, it has the 3rd highest importance, while it has the worst performance.

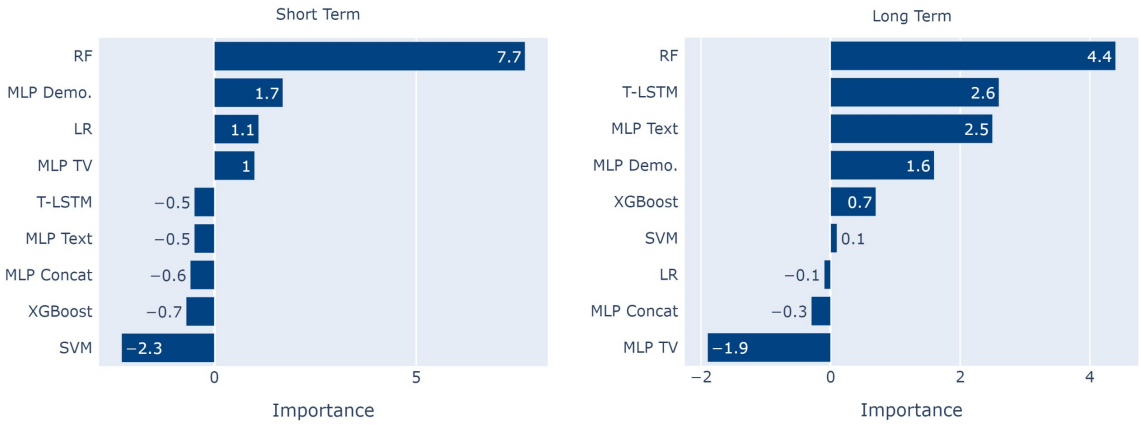

Figure 7.8: Stacked Ensemble Importances. The figure shows the feature importance for the SVM model that creates the Ensemble baseline. The left diagram shows the importances for short term and the right one the importances for long term. MLP TV is the MLP Time Variant Model.

Table 7.9 presents an estimation of the generalization error of the stacked ensemble. It shows a similar behavior as in the MLP ensemble. Short term has a large generalization error for recall and a small one for TNR. Long term does seem to generalize well.

	ST Recall	ST TNR	LT Recall	LT TNR
Training	1.00	0.94	0.86	0.91
Validation	0.69	0.89	0.84	0.91
Test	0.47	0.89	0.86	0.89

Table 7.9: Stacked Ensemble generalization. The table shows the results for each class of the Stacked Ensemble Model for each Split. ST Recall indicates that short term prediction does not generalize well for prediction of the positive class. All other classes seem to generalize well.

Concluding the analysis of generalization, it is most likely that the bigger class imbalance of the short term prediction leads to a weak generalization for the kidney graft loss class. The positive result on the other hand is that the data balance techniques seem to work for the long term prediction. It does generalize well even if there still is a big data imbalance.

7.2.5 Random Forest Feature Importances

One major advantage of random forest is that internal estimates are also used to measure variable importance [Breiman 2001]. The random forest importances are shown in Figure 7.9. A full list of feature importances can be found in the appendix, Figure A.2 for short term and Figure A.1 for long term.
The diagrams show similar feature importances with both having the creatinine months as top features. A reason that each month is a stronger indicator then the value throughout the whole year could be, that the model is able to so see rises in creatinine. An assumption by the supervising medical expert is that the rises indicate diseases like infections, loss of fluid due not drinking enough, diarrhea, vomiting, or medication effects. These diseases would harm the kidney and increase the chance of a kidney graft loss. Protein is the second most important time variant feature. On the contrary to creatinine, the mean over the whole year has a higher importance, than the means for each month. The most important demographic features are either related to the age of the donator or recipient, or they are related to dialysis. In short term the age of the donor is more important, while in long term the age of the recipient is more important. The medical expert assumes that a kidney of an older patient leads kidney sickness that happen early

after transplantation like renal vein thrombosis, arterial occlusion or renal artery stenosis. Long term graft loss is stronger connected to the recipient's age, since younger patients are overall healthier.

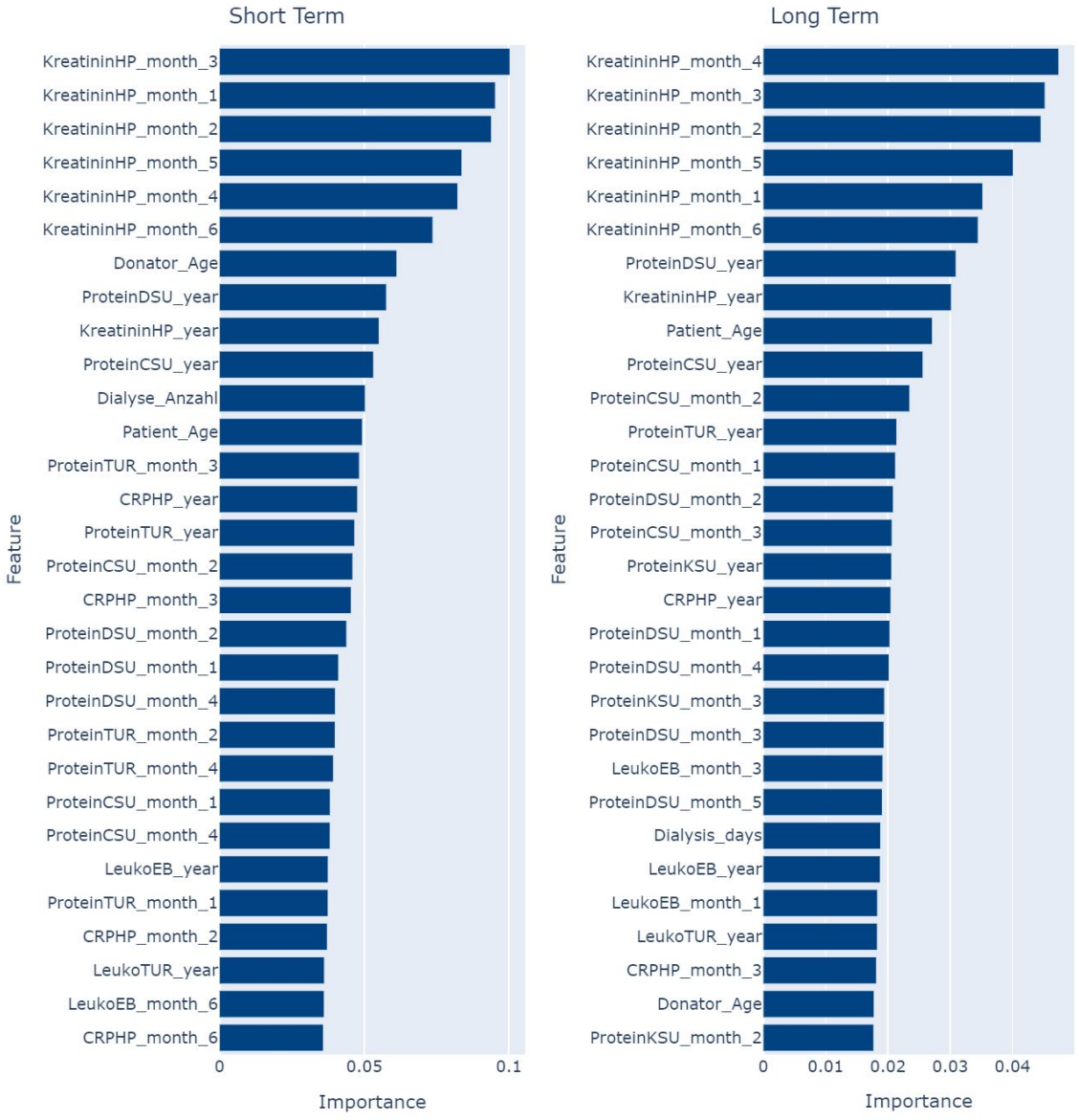

Figure 7.9: Random Forest Feature Importances. The left diagram shows the importance for the short term task and the right one for the long term task.

7.3 Discussion

We have conducted many experiments throughout this thesis. Starting with 7 baselines to find a suitable model objective and 12 Baselines for each prediction time frame result in 31 baselines. This section discusses some findings and challenges we encountered.

Balanced Accuracy

Optimizing all models for the objective that resulted with the highest recall turned out to be the correct decision. BAcc evaluates the confusion matrix independently of the class imbalance, which in our case sets the focus stronger on the positive class compared to the other metrics. The positive class is more important from a medical perspective as shown in section 5.1. An increased focus on the positive class is also relevant from a technical perspective, since we have enough samples to train the negative class to achieve a good TNR. A good recall for the long term task was only achieved by combining all ML models to a stacked ensemble. Which is not the case for the short term task as shown in the next paragraph.

Task Comparison

The ensemble model for the short term task achieves a recall of 0.47, while the ensemble model for the long term task achieves a recall of 0.83. The positive class for the short term prediction task only contains 38 samples, which turned out not to be enough to train the models. Summing up the samples from the 3-fold cross-validation, the stacked ensemble model predicted 9 out of 19 positive samples in the test dataset correctly. For these patients, we have the more accurate information that they will undergo kidney graft loss during next year instead of the next five years. But the models also predicts 148 false positive samples, thus increasing the medical care for 9 patients with a high risk, comes with the downside of increasing it for 148 patients who don't need it at that time. Overall, this seems to be a bad trade and shows that the ensemble model for short term prediction is not very helpful. The discussion focuses on the long term prediction task from now on, since the performance of the short term predictions models is underwhelming.

Fusion Methods

The stacked ensemble is the best performing fusion method compared to concatenation and mean. Section 7.2.3 shows why concatenation performs badly. Creating a mean is a very simple attempt to get a result from multiple single data models, but the results exceed our expectations. The mean method is able to increase the BAcc from 0.56 for MLP Demographic, 0.54 for MLP Time Variant and 0.64 for MLP Text to 0.74 for MLP mean.

The stacked ensemble shows that a late fusion approach has an advantage over early fusion. It is able to combine the advantages of each Model with individual importances. It combines the strength of MLP for text data, the good results of RF and XGBoost for structured data and the ability of T-LSTM to process sequential data. Our good results of RF and XGBoost for structured data are in accordance with the findings of [Grinsztajn et al. 2022]. Overall, a well working fusion method seems to be a key element for this task.

Class Imbalance

Besides a small dataset and a multimodal data structure is the class imbalance the main challenge for this thesis. Different loss functions, optimizers, and data balancing techniques create the hyperparameters to counter this challenge. The impact of each hyperparameter is difficult to quantify, since the HPO automatically chooses the best fitting ones. MLP experiments showed that some hyperparameters are clearly better in handling the class imbalance than others. Therefore, we removed optimizer and loss function from the HPO in Section 5.2.5 and continued with AdamW as loss function and BCE as optimizer.

The most impactful hyperparameters to handle class imbalance seem to be downsampling and pos_weight. Comparing all HPO results shown in the Appendix A the smallest value for downsampling is 2 and the highest is 18. High values like 18 are picked by the ensembles, since they do not have other possibilities to handle the class imbalance.

Downsampling seems to be a good way to handle the class imbalance.

7.4 Summary

This chapter presents and evaluates all results of the experiments we conducted. First, the chapter shows results of the MLP demographic model that has been optimized for different evaluation metrics. BAcc is the metric that achieves the best recall and gives the trade-off according to our medical specialist. Thus, we chose it as the objective to maximize during the tuning process. The next section shows the results for all MLP baselines that are conducted throughout the experiments. MLP Ensemble achieves the best result for the long term task, with a BAcc of 0.78. MLP Text achieves the best result for the short term task, with a BAcc of 0.68. Section 7.1.3 presents the results of all conducted ML models. The stacked ensemble model scores the best result for the long term task with a BAcc of 0.86. The stacked ensemble model has a score of 0.68 BAcc for the short term task. The next part compares the RF and MLP Text results between the dataset MO and MV. MV achieves better results with RF in short term prediction. Dataset MO achieves better results in the other comparisons. The new preprocessing of dataset MV does not achieve higher scores, mainly, because we defined the labels in a cleaner way. ML models using dataset MV have a better generalization estimation due to the 3-fold cross validation.
The next section analyses the results from a qualitative perspective. First, we compare the short and long term prediction task with the stacked ensemble model. The main finding is that the model performs badly in predicting the positive class in the short term task. Next, we compare the data quality and show that the EHR from location Mitte have a better quality than those from Virchow. Another finding is that the amount of texts does not influence the chance of a sample being predicted correctly. An analysis of the concatenation model shows that it does not perform well as a fusion method, because demographic, time variant and text data need different training duration and hyperparameters. The next part investigates the connection between feature importances and the generalization estimation of the ensemble model. We discover that a generalization error from ML models used as input for the ensemble model lead to feature importances that do not correlate to the performance of the input predictions. The effect leads the ensemble model to perform worse on the short term task. The last part of this section shows the feature importances of the RF model. Creatinine is the feature with the highest importance for both, short and long term prediction.
The last section discusses findings and challenges about balanced accuracy, the short term task, the stacked ensemble and the class imbalance.

Chapter 8

Conclusion and Future Outlook

This thesis aims to improve the health of patients that received a kidney transplant by predicting graft loss. A medical physicist can use this information to increase the care for high risk patients and ideally postpone or even prevent the graft loss. The goal is divided in two tasks. The first task based on the work of [Islam 2021] and it has the goal to predict graft loss in long term (1 to 6 years after transplantation). Our model can successfully learn this task and is able to predict more than 8 out of 10 graft losses. The second task based on the work of [Reuter 2021] and it has the goal to predict graft loss in short term (1 to 2 years after transplantation). The final model for this task is not able to learn this task properly, since the short term time frame contains 72% less patients undergoing kidney graft loss. It predicts less than 5 out of 10 graft losses correctly. We achieved more precise predictions and a better generalization estimation through our contributions. The main contributions are a new data preprocessing pipeline, a more precise and rigorous definition of the labels, new ML model approaches like SVM, XGBoost, T-LSTM and stacked ensemble, implementation of 3-fold cross-validation, early stopping, HPO for all models, and a study about the performance of different evaluation metrics. A detailed evaluation about the contributions can be done by reviewing the hypotheses that we stated at the beginning of the thesis:

Hypothesis 1: Data centric approaches improve all baselines
We created a new data baseline (dataset MV) using the data centric approach. This baseline creates predictions that more trustful compared to those that have been created with the dataset MO. This trust mainly results from two major changes. First, redefining the labels so that the model only predicts kidney graft loss instead of kidney graft loss and death. Second, by an improved generalization estimate that is created through 3-fold cross validation. We also improved the preprocessing so that a decrease in performance caused by more precise labels could be compromised.

Hypothesis 2: Model centric approaches improve all baselines
Adding different loss functions, optimizers, and techniques to handle class imbalance with the combination of a HPO greatly improves the performance of our models. The stacked ensemble creates the best baseline by combining strengths of all conducted machine learning models. Comparing the stacked ensemble with the final outcome of [Islam 2021] we are able to improve the BAcc from 0.71 to 0.89. This is a significant improvement, which has to result from the model centric approach, since the data centric approach did not increase the outcome in BAcc.

Hypothesis 3: Including the time-dimension creates a better baseline
We choose T-LSTM to be able to process the time-dimension of the sequential data. The T-LSTM itself does not create a better baseline. But its predictions serves as an input for the stacked ensemble. It has the second-highest importance for long term prediction, showing that this ap-

proach substantially contributed to the final outcome of the thesis. Furthermore, time constrains did not allow the HPO to train nearly as many trials for T-LSTM as for the other models. There is probably still a lot of unused potential in this model.

Hypothesis 4: Redefining the target metric improves the results from a medical perspective
The chosen model objective balanced accuracy is discussed in Section 7.3. The metric focuses stronger on the positive class compared to all other conducted evaluation metrics, but still takes the negative class into account. Comparing the stacked ensemble with the final outcome of [Islam 2021] we are able to improve the recall by from 0.65 to 0.83 and the TNR from 0.78 to 0.89. The recall increased by 0.18 and the TNR by 0.11, showing that the improvement is stronger on the positive class. Recall is the more important metric from a medical perspective, since a high recall means, that the prediction misses fewer patients undergoing kidney graft loss. Conducting different evaluation metrics was an important step to achieve results that fulfill the medical requirements for a practical implementation.

Concluding the analysis of the hypotheses, all the hypothesis are proven. This achievement and a decent performance in the long term prediction task makes this thesis a success. It still has a lot of potential for improvements, but the sufficient performance leads to the question of practical use:

How can we further improve the performance, and how can we implement the model for a practical use?

Domain-specific BERT model. The MLP experiments show that the text data holds the most information for our task. We created the embeddings with a gBERT Model that [Reuter 2021] and [Islam 2021] finetuned. They also tested a medical German BERT (MedBERT) with the expectation that the predictions benefits from a language model that is pre-trained on domain-specific data. MedBERT does not achieve better results, most likely due to a small text training corpus consisting of the International Classification of Diseases (ICD-10) [Shrestha 2021]. gBert in comparison uses the German corpus of OSCAR dataset, Wikipedia, news, speeches, and legal data which is a big corpus of about 163 GB of text data [Chan et al. 2020]. [Bressem et al. 2020] presents a pretrained BERT for classification of chest radiographic reports. Their current research has the goal to enhance the model for more generic medicine purposes by including texts of different medical fields. The corpus will also incorporate texts from nephrology, which will most probably improve the performance on tasks such as ours.

Fusion learning approaches. The conducted fusion approaches are rather simple. We either use the early fusion approach and concatenate the features before training, or aggregate the predicted probabilities using late fusion. The late fusion approach stacked ensemble outperformed concatenating by a large margin, indicating that the right fusion method is a key element of for our task. More complex fusion approaches might have more potential. We recommend further experiments with late fusion approaches or hybrid fusion approaches:

- **Concatenate features to probabilities:** The stacked ensemble only uses seven probabilities as an input. Adding more data by concatenating the all or a combination of demographic, time variant or text data to the input vector could improve the performance.
- **T-LSTM late fusion:** Single data baselines strongly contributed to the stacked ensemble. The T-LSTM currently uses a concatenated input vector including demographic, time variant and text data. Training the T-LSTM separately for both sequential data sources time-variant and text could create more valuable inputs for the stacked ensemble.

- **Concatenate layers instead of probabilities:** Probabilities just have the information about the certainty of the prediction, while layer of a NN carry information about semantic meanings [Pérez-Rúa et al. 2019]. Concatenate the last layer instead of the probabilities would allow the ensemble to learn contexts between semantic meanings.

- **Multimodal Fusion Architecture Search (MFAS):** [Pérez-Rúa et al. 2019] propose MFAS, an algorithm that finds the optimal fusion architecture for a given dataset. It uses a generic search space that spans many possible fusion architectures.

Practical use and continuous prediction. We only see a practical use for a model that is optimized for the long term task. Unfortunately, we have too few examples to train a reliable model for the short term task. Therefore, short term prediction can only achieve a good performance by either collecting more data or a redefining the labels, e.g., increasing the short term prediction time frame to a range of two years, instead of one.
Long term prediction on the other hand performs well enough to put the model into practical use. The easiest way to give medical physicists access to our prediction would be an implementation in TBase. TBase is described in Section 3.1, it does not only serve as a database, but also as an interface that accesses hospital information systems, transplant-specific data and medication lists for drug-drug interactions [Osmanodja et al. 2021]. ML predictions would be a useful addition to the TBase landscape.
The functionality of the model is currently limited to one prediction that we create one year after the transplantation. Data collected after this point is not taken into account, and after some time the prediction does not represent the actual risk of kidney graft loss anymore. In order to create a model that is able to predict the risk at anytime, we would need to develop a model that is able to do continuous prediction. Instead of a data collection time frame of one year, it would be designed flexible. A prediction could be done anytime, once enough data has been collected. This model could create regularly predictions for each patient, which enables monitoring about the risk of kidney graft loss. This finally helps to adjust the amount of medical care and also provides important information for further research into preventing the loss of kidney transplants.

Bibliography

[Akiba et al. 2019] Akiba, T., Sano, S., Yanase, T., Ohta, T., und Koyama, M., Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[Ashenden 2021] Ashenden, S. K., *The Era of Artificial Intelligence, Machine Learning, and Data Science in the Pharmaceutical Industry*. Elsevier Inc.

[Baytas et al. 2017] Baytas, I. M., Xiao, C., Zhang, X., Wang, F., Jain, A. K., und Zhou, J., Patient subtyping via time-aware lstm networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, Seite 65–74, New York, NY, USA. Association for Computing Machinery.

[Bradley 1997] Bradley, A. P., The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159.

[Breiman 2001] Breiman, L., Random forests. *Machine learning*, 45(1):5–32.

[Bressem et al. 2020] Bressem, K., Adams, L., Gaudin, R., Tröltzsch, D., Hamm, B., Makowski, M., Schüle, C.-Y., Vahldiek, J., und Niehues, S., Highly accurate classification of chest radiographic reports using a deep learning natural language model pretrained on 3.8 million text reports. *Bioinformatics (Oxford, England)*, 36.

[Chan et al. 2020] Chan, B., Schweter, S., und Möller, T., German's next language model. In *Proceedings of the 28th International Conference on Computational Linguistics*, Seiten 6788–6796, Barcelona, Spain (Online). International Committee on Computational Linguistics.

[Chawla et al. 2002] Chawla, N. V., Bowyer, K. W., Hall, L. O., und Kegelmeyer, W. P., SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357.

[Chicco und Jurman 2020] Chicco, D. und Jurman, G., The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation. *BMC Genomics*, (21).

[Christoph et al. 2015] Christoph, J., Maier, C., Schmidt, D., Ganslandt, T., und Sedlmayr, M., Erschließung komplexer nephrologischer daten – erfahrungen bei der transformation tbase nach i2b2. *GMDS 2015)*.

[Cruz et al. 2007] Cruz, D., Bolgan, I., Perazella, M. A., Bonello, M., de Cal, M., Corradi, V., Polanco, N., Ocampo, C., Nalesso, F., Piccinni, P., und Ronco, C., North east italian prospective hospital renal outcome survey on acute kidney injury (neiphros-aki): targeting the problem with the rifle criteria. *Clinical journal of the American Society of Nephrology : CJASN*, 2 3:418–25.

[Devlin et al. 2018] Devlin, J., Chang, M., Lee, K., und Toutanova, K., BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

[DSO 2021] DSO, Jahresbericht organspende und transplantation in deutschland. *Deutsche Stiftung Organtransplantation*.

[Efraimidis 2010] Efraimidis, P. S., Weighted random sampling over data streams. *CoRR*, abs/1012.0256.

[Fernández et al. 2018] Fernández, A., García, S., Galar, M., Prati, R. C., Krawczyk, B., und Herrera, F., *Learning from Imbalanced Data Sets*. Springer, Cham.

[Gandhi 2018] Gandhi, R., Support vector machine — introduction to machine learning algorithms. https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47.

[García et al. 2015] García, S., Luengo, J., und Herrera, F., *Data Preprocessing in Data Mining*. Springer, Cham.

[Gentile und Warmuth 1998] Gentile, C. und Warmuth, M. K. K., Linear hinge loss and average margin. In Kearns, M., Solla, S., und Cohn, D., Editoren, *Advances in Neural Information Processing Systems*, Volume 11. MIT Press.

[Grinsztajn et al. 2022] Grinsztajn, L., Oyallon, E., und Varoquaux, G., Why do tree-based models still outperform deep learning on tabular data?

[Gupta 2019] Gupta, T., Deep learning: Feedforward neural network. https://towardsdatascience.com/deep-learning-feedforward-neural-network-26a6705dbdc7.

[Haldar 2020] Haldar, N., Artificial neural network for the prediction of short-term graft failure of trans-planted kidneys. Mastersthesis, Technische Universität Berlin, Einsteinufer 17, 10587 Berlin.

[Hienen 2021] Hienen, M., Clustering-based prediction of long-term renal transplant failure risks. Mastersthesis, Technische Universität Berlin, Einsteinufer 17, 10587 Berlin.

[Hochreiter und Schmidhuber 1997] Hochreiter, S. und Schmidhuber, J., Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

[Hoerl und Kennard 2000] Hoerl, A. E. und Kennard, R. W., Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 42(1):80–86.

[hopkinsmedicine.org 2021] hopkinsmedicine.org, Kidney transplant. https://www.hopkinsmedicine.org/health/treatment-tests-and-therapies/kidney-transplant.

[Horev 2018] Horev, R., Bert explained: State of the art language model for nlp. https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270.

[IQWiG 2016] IQWiG, Cancer: What do the codes in the doctor's letter mean? https://www.ncbi.nlm.nih.gov/books/NBK279426. Germany: Institute for Quality and Efficiency in Health Care (IQWiG).

[Islam 2021] Islam, R., A multi-modal deep learning strategy for long term kidney graft loss prediction on imbalanced dataset. Mastersthesis, Beuth Hochschule für Technik Berlin, Luxemburger Straße 10, 13353 Berlin.

[Jordan und Mitchell 2015] Jordan, M. und Mitchell, T., Machine learning: Trends, perspectives, and prospects. *Science (New York, N.Y.)*, 349(6245):255—260.

[Khanna 2020] Khanna, C., What and why behind fit_transform() and transform() in scikit-learn! https://towardsdatascience.com/what-and-why-behind-fit-transform-vs-transform-in-scikit-learn-78f915cf96fe.

[Kingma und Ba 2014] Kingma, D. und Ba, J., Adam: A method for stochastic optimization. *International Conference on Learning Representations*.

[Lin et al. 2017] Lin, T.-Y., Goyal, P., Girshick, R., He, K., und Dollár, P., Focal loss for dense object detection.

[Loshchilov und Hutter 2017] Loshchilov, I. und Hutter, F., Decoupled weight decay regularization.

[Mathew et al. 2020] Mathew, A., Amudha, P., und Sivakumari, S., Deep learning techniques: An overview.

[Molnar 2022] Molnar, C., *Interpretable Machine Learning - A Guide for Making Black Box Models Explainable*. 2 Auflage.

[Morde 2019] Morde, V., Xgboost algorithm: Long may she reign! https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d.

[Ng 2021] Ng, A., Mlops from model centric to data centric ai.

[Osmanodja et al. 2021] Osmanodja, B., Schmidt, D., Budde, K., und Mayrdorfer, M., Tbase - an integrated electronic health record and research database for kidney transplant recipients. *Journal of Visualized Experiments*, 2021.

[Palmisano et al. 2021] Palmisano, A., Gandolfini, I., Delsante, M., Cantarelli, C., Fiaccadori, E., Cravedi, P., und Maggiore, U., Acute kidney injury (aki) before and after kidney transplantation: Causes, medical approach, and implications for the long-term outcomes. *Journal of Clinical Medicine*, 10:1484.

[Pérez-Rúa et al. 2019] Pérez-Rúa, J., Vielzeuf, V., Pateux, S., Baccouche, M., und Jurie, F., MFAS: multimodal fusion architecture search. *CoRR*, abs/1903.06496.

[Prakash 2021] Prakash, A., Working with sparse features in machine learning models. https://www.kdnuggets.com/2021/01/sparse-features-machine-learning-models.html.

[Reuter 2021] Reuter, R., Prediction of permanent kidney graft loss via deep ensemble learning and nlp using a biased dataset of electronic health records. Mastersthesis, Beuth Hochschule für Technik Berlin, Luxemburger Straße 10, 13353 Berlin. https://prof.bht-berlin.de/loeser/teaching/masterthesis/.

[Ruder 2016] Ruder, S., An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747.

[Shinohara et al. 2013] Shinohara, E. Y., Aramaki, E., Imai, T., Miura, Y., Tonoike, M., Ohkuma, T., Masuichi, H., und Ohe, K., An easily implemented method for abbreviation expansion for the medical domain in japanese text. a preliminary study. *Methods of information in medicine*, (52).

[Shrestha 2021] Shrestha, M., Development of a language model for medical domain. masterthesis, Hochschule Rhein-Waal.

[Srivastava et al. 2014] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., und Salakhutdinov, R., Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

[Tibshirani 1996] Tibshirani, R., Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288.

[Vaswani et al. 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., und Polosukhin, I., Attention is all you need. *CoRR*, abs/1706.03762.

[Wekerle et al. 2017] Wekerle, T., Segev, D., Lechler, R., und Oberbauer, R., Strategies for long-term preservation of kidney graft function. *Lancet (London, England)*, 389(10084):2152—2162.

[Wolf et al. 2020] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., Drame, M., Lhoest, Q., und Rush, A. M., Transformers: State-of-the-Art Natural Language Processing. Seiten 38–45. Association for Computational Linguistics.

[Xing et al. 2018] Xing, L., Krupinski, E., und Cai, J., Artificial intelligence will soon change the landscape of medical physics research and practice. *Medical Physics*, 45.

Appendix A

Hyperparameters and Results

Logistic Regression

Table A.1 shows the hyperparameter picks for LR. The time variant data is excluded for both, short and long term HPO. In order to handle data imbalance, short term uses SMOTE, while long term uses a low threshold and a higher downsampling.

Hyperparameter	Range / Categories	Step	ST Pick	LT Pick
threshold	0.1 to 0.8	0.02	0.66	0.22
l1_ratio	0 to 1	0.05	0.7	0.45
c_value	100 / 10 / 1 / 0.1 / 0.01	-	1	100
DS-Rate	0 to 12	1	7	10
SMOTE	True / False	-	True	False
Drop_lab	True / False	-	True	True

Table A.1: The table shows the hyperparameters for LR, their ranges or categories and the step size if is a range. The columns ST pick shows the chosen hyperparameter for short term and LT pick, the chosen hyperparameter for long term.

The final results for the LR model are shown in table A.2. Short and long term have similar BAccs with 0.55 and 0.51. This is a general low result, since a BAcc of 0.5 can be achieved with random predictions. The Recall for long term is more than twice as high, thus LR predicts the positive class better in short term and the negative class better in long term.

		BAcc	Recall	Precision	F1	MCC	AUC	PR AUC	Var(BAcc)
ST	**Train**	0.82	0.76	0.54	0.53	0.21	0.9	0.08	0.001
	Val	0.64	0.37	0.52	0.51	0.1	0.57	0.03	0.003
	Test	0.55	0.21	0.51	0.49	0.04	0.66	0.02	0.005
LT	**Train**	0.7	0.89	0.54	0.41	0.17	0.77	0.12	0.012
	Val	0.62	0.68	0.52	0.42	0.10	0.63	0.07	0.04
	Test	0.51	0.49	0.50	0.38	0.01	0.53	0.05	0.005

Table A.2: The table shows the results for the LR model. The ST rows shows the results for short term and the LT rows show the results for long term.

Support Vector Machine

The HPO results for the SVM model can be seen in table A.3. The regularization strength 'c' is missing, because it was set to the default value of '1' by mistake. This might lead to a suboptimal set of hyperparameters. The LR model also uses the 'c_value' as regularization strength, and the HPO picked a value of '1' for short term and a value of '100' for long term. Therefore, it is likely that at least long term needs a higher 'c' value in SVM.

Hyperparameter	Range / Categories	Step	ST Pick	LT Pick
Threshold	0.1 to 0.9	0.02	0.58	0.46
Shrinking	True / False	-	True	False
Kernel	linear / poly / rbf / sigmoid	-	sigmoid	rbf
DS-Rate	0 to 12	1	8	9
SMOTE	True / False	-	True	False
Drop_lab	True / False	-	False	False

Table A.3: The table shows the hyperparameters for SVM, their ranges or categories and the step size if is a range. The columns ST pick shows the chosen hyperparameter for short term and LT pick the chosen hyperparameter for long term.

The results for the SVM model are listed in table A.4. They are very similar to the LR outcome, with BAcc just above 0.5. Short Term a higher F1 score of 0.51, but only achieves a Recall of 0.16, which still results in a very weak model.

		BAcc	Recall	Precision	F1	MCC	AUC	PR AUC	Var(BAcc)
ST	**Train**	0.71	0.63	0.52	0.48	0.12	0.78	0.06	0.0
	Val	0.63	0.32	0.53	0.54	0.13	0.67	0.09	0.006
	Test	**0.55**	0.16	0.51	0.51	0.05	0.62	0.02	0.000
LT	**Train**	0.73	0.51	0.66	0.67	0.37	0.9	0.41	0.047
	Val	0.60	0.68	0.52	0.40	0.09	0.64	0.10	0.054
	Test	**0.52**	0.53	0.50	0.37	0.01	0.52	0.06	0.040

Table A.4: The table shows the results for the SVM model. The ST rows shows the results for short term and the LT rows show the results for long term.

Random Forest

The HPO picks for the RF model are presented in table A.5. The hyperparameter are similar for short and long term. Short term uses a higher DS-rate, because it has a higher data.

Hyperparameter	Range / Categories	Step	ST Pick	LT Pick
threshold	0.1 to 0.5	0.02	0.22	0.32
criterion	Gini / Entropy	-	Entropy	Gini
bootstrap	True / False	-	True	True
max_depth	0 to None	1	26	None
min_samples_split	2 / 5 / 10 / 20	-	10	10
min_samples_leaf	1 / 2 / 4	-	1	4
max_features	sqrt / log2	-	log2	sqrt
n_estimators	200 to 1500	1	893	1133
DS-Rate	0 to 12	1	11	7
SMOTE	True / False	-	False	False
Drop_lab	True / False	-	False	False

Table A.5: The table shows the hyperparameters for RF, their ranges, or categories and the step size if is a range. The columns ST pick shows the chosen hyperparameter for short term and LT pick the chosen hyperparameter for long term.

Table A.6 shows the RF results. RF achieves very good results with a BAcc of 0.75 in short term and 0.64 in long term. The short term model performs better in F1 than the long term model, despite having a higher class imbalance. Both models perform better on the negative class, because their BAcc is higher than recall. But they predict at least half of the positive samples correctly.

		BAcc	Recall	Precision	F1	MCC	AUC	PR AUC	Var(BAcc)
ST	**Train**	0.95	1.0	0.56	0.58	0.33	1.0	0.83	0.0
	Val	0.78	0.63	0.55	0.58	0.24	0.86	0.11	0.002
	Test	**0.75**	0.57	0.54	0.56	0.2	0.92	0.1	0.007
LT	**Train**	0.78	1.0	0.55	0.44	0.23	1.0	0.97	0.009
	Val	0.7	0.62	0.55	0.54	0.21	0.74	0.17	0.019
	Test	**0.64**	0.5	0.53	0.51	0.14	0.74	0.14	0.039

Table A.6: The table shows the results for the RF model. The ST rows shows the results for short term and the LT rows show the results for long term.

XGBoost

The hyperparameter picks for XGBoost are presented in table A.7. Long term uses a high DS-Rate of '10' and SMOTE to handle the class imbalance. Short just uses a DS-rate of '4' but compensates this with a threshold of '0.3'.

Hyperparameter	Range / Categories	Step	ST Pick	LT Pick
threshold	0.1 to 0.5	0.02	0.3	0.85
colsample_bylevel	0.1 to 1	0.1	0.8	0.8
colsample_bytree	0.1 to 1	0.1	0.9	0.6
learning_rate	0.001 / 0.01 / 0.1 / 1 / 10 / 100	-	0.1	0.01
max_depth	1 to 10000	1	7440	2204
n_estimators	100 to 1000	1	508	433
objective	binary:logistic / binary:hinge	-	logistic	hinge
rate_drop	0.1 to 0.9	0.1	0.7	0.6
skip_drop	0.1 to 0.9	0.1	0.7	0.8
reg_alpha	0 / 0.001 / 0.01 / 0.1 / 1 / 10 / 100	-	100	10
reg_lambda	0 / 0.001 / 0.01 / 0.1 / 1 / 10 / 100	-	100	1
subsample	0.1 to 1	0.1	1	0.8
DS-Rate	0 to 12	1	4	10
SMOTE	True / False	-	False	True
Drop_lab	True / False	-	False	False

Table A.7: The table shows the hyperparameters for XGBoost, their ranges or categories and the step size if is a range. The columns ST pick shows the chosen hyperparameter for short term and LT pick the chosen hyperparameter for long term.

Table A.8 shows the results for the XGBoost model. Long term performs a bit better with a BAcc of 0.7 compared to a BAcc of 0.66 in short term. The recall of long term is 41% higher than short term. A similar BAcc but stronger Recall shows, that long term is better in predicting the positive class and short better in predicting the negative class.

		BAcc	Recall	Precision	F1	MCC	AUC	PR AUC	Var(BAcc)
ST	**Train**	0.77	0.59	0.57	0.61	0.29	0.85	0.3	0.0
	Val	0.79	0.62	0.57	0.6	0.28	0.76	0.2	0.011
	Test	**0.66**	0.37	0.54	0.55	0.15	0.7	0.06	0.003
LT	**Train**	0.84	0.93	0.57	0.56	0.32	0.84	0.54	0.017
	Val	0.73	0.68	0.56	0.54	0.23	0.73	0.41	0.042
	Test	**0.7**	0.63	0.55	0.53	0.2	0.7	0.39	0.012

Table A.8: The table shows the results for the XGBoost model. The ST rows shows the results for short term and the LT rows show the results for long term.

MLP Demographic Data

The hyperparameter of MLP demographic are presented in table A.9. Both, short and long term, use WRS, decremental layers, but no SMOTE. Short term compensates the higher class imbalance with a high pos_weight of '95'.

Hyperparameter	Range / Categories	Step	ST Pick	LT Pick
WRS	True / False	-	True	True
decremental	True / False	-	True	True
learning rate	1e-5 to 1e-2	1e-5	7.59e-3	6.83e-3
layers	1 to 6	1	5	2
dropout	0.4 to 0.9	0.1	0.4	0.7
batch size	8 / 16 / 32	-	32	8
nb_units	32 to 512	1	156	443
pos_weight	1 to 100	1	95	23
DS-Rate	0 to 12	1	2	3
SMOTE	True / False	-	False	False

Table A.9: The table shows the hyperparameters for MLP using demographic data, their ranges or categories and the step size if is a range. The columns ST pick shows the chosen hyperparameter for short term and LT pick the chosen hyperparameter for long term.

The results for MLP demographic are listed in table A.10. The gap in BAcc and recall between validation dataset and test dataset is high. The recall is 28% lower for short term and 26% lower for long term, indicating that the model might overfit on the validation dataset.

		BAcc	Recall	Precision	F1	MCC	AUC	PR AUC	Var(BAcc)
ST	**Train**	0.85	0.93	0.53	0.48	0.19	0.92	0.27	0.003
	Val	0.67	0.57	0.51	0.46	0.09	0.66	0.03	0.006
	Test	**0.59**	0.41	0.51	0.46	0.05	0.69	0.03	0.002
LT	**Train**	0.8	1.0	0.57	0.47	0.28	0.94	0.42	0.107
	Val	0.67	0.72	0.54	0.46	0.16	0.65	0.1	0.024
	Test	**0.56**	0.53	0.51	0.41	0.05	0.57	0.07	0.009

Table A.10: The table shows the results for the MLP demographic data model. The ST rows shows the results for short term and the LT rows show the results for long term.

MLP Time Variant Data

Table A.11 presents the hyperparameters for MLP time variant. Short term HPO has chosen the maximum layers of 6 layers. A HPO with a higher layer range could find better hyperparameters, but tests showed that more layers increase training duration rather than the actual results.

Hyperparameter	Range / Categories	Step	ST Pick	LT Pick
WRS	True / False	-	False	True
decremental	True / False	-	True	False
learning rate	1e-5 to 1e-2	1e-5	1.24e-3	4.18e-3
layers	1 to 6	1	6	5
dropout	0.4 to 0.9	0.1	0.5	0.6
batch size	8 / 16 / 32	-	8	8
nb_units	32 to 512	1	264	98
pos_weight	1 to 100	1	74	65
DS-Rate	0 to 12	1	2	2
SMOTE	True / False	-	False	False

Table A.11: The table shows the hyperparameters for MLP using time variant data, their ranges or categories and the step size if is a range. The columns ST pick shows the chosen hyperparameter for short term and LT pick the chosen hyperparameter for long term.

The outcome of the MLP time variant model is shown in table A.12. A better performance on the validation datasets than on the test datasets indicates validation overfitting.

		BAcc	Recall	Precision	F1	MCC	AUC	PR AUC	Var(BAcc)
ST	**Train**	0.82	0.97	0.53	0.44	0.19	0.96	0.34	0.014
	Val	0.78	0.74	0.52	0.5	0.16	0.75	0.15	0.001
	Test	**0.58**	0.36	0.51	0.46	0.05	0.6	0.05	0.006
LT	**Train**	0.71	0.89	0.54	0.42	0.19	0.82	0.26	0.09
	Val	0.66	0.64	0.55	0.48	0.17	0.62	0.13	0.042
	Test	**0.54**	0.4	0.51	0.45	0.04	0.56	0.08	0.042

Table A.12: The table shows the results for the MLP time variant data model. The ST rows shows the results for short term and the LT rows show the results for long term.

MLP Text Data

Table A.13 lists the hyperparameters that HPO picked for MLP text. Short term uses a higher pos_weight and long term a higher DS-Rate and SMOTE to handle data imbalance. Compared to MLP demographic and MLP time variant, the pos_weight is very low.

Hyperparameter	Range / Categories	Step	ST Pick	LT Pick
WRS	True / False	-	True	True
decremental	True / False	-	False	True
learning rate	1e-5 to 1e-2	1e-5	7.95e-3	9.82e-3
layers	1 to 6	1	1	2
dropout	0.4 to 0.9	0.1	0.8	0.7
batch size	8 / 16 / 32	-	8	8
nb_units	32 to 512	1	244	78
pos_weight	1 to 100	1	5	1
DS-Rate	0 to 12	1	5	10
SMOTE	True / False	-	False	True

Table A.13: The table shows the hyperparameters for MLP using text data, their ranges or categories and the step size if is a range. The columns ST pick shows the chosen hyperparameter for short term and LT pick the chosen hyperparameter for long term.

The results for MLP text are shown in table A.14. The text data outperforms the other single MLP models, with a BAcc of 0.68 for short term and 0.64 for long term. MLP text is only using few layers with one for short term and two for long term prediction.

		BAcc	Recall	Precision	F1	MCC	AUC	PR AUC	Var(BAcc)
ST	**Train**	0.66	0.64	0.51	0.43	0.08	0.74	0.05	0.0
	Val	0.74	0.79	0.52	0.44	0.12	0.69	0.04	0.001
	Test	**0.68**	0.69	0.51	0.43	0.09	0.7	0.05	0.001
LT	**Train**	0.67	0.74	0.54	0.44	0.15	0.74	0.12	0.021
	Val	0.69	0.78	0.54	0.45	0.17	0.68	0.11	0.037
	Test	**0.64**	0.7	0.53	0.43	0.13	0.67	0.08	0.08

Table A.14: The table shows the results for the MLP text data model. The ST rows shows the results for short term and the LT rows show the results for long term.

MLP Mean

The MLP mean baseline has no hyperparameters, since it is mean results of MLP demographic, MLP time variant and MLP text. The results are shown in table A.15.

		BAcc	Recall	Precision	F1	MCC	AUC	PR AUC	Var(BAcc)
ST	**Train**	0.9	0.97	0.54	0.52	0.24	0.97	0.57	0.002
	Val	0.69	0.53	0.52	0.5	0.12	0.84	0.07	0.001
	Test	**0.61**	0.36	0.51	0.49	0.07	0.68	0.1	0.007
LT	**Train**	0.74	0.88	0.55	0.46	0.21	0.85	0.38	0.039
	Val	0.73	0.86	0.55	0.47	0.2	0.82	0.27	0.054
	Test	**0.74**	0.89	0.54	0.45	0.2	0.84	0.3	0.029

Table A.15: The table shows the mean results of the MLP single models. The ST rows shows the results for short term and the LT rows show the results for long term.

MLP Concatenated Results

Table A.16 presents the chosen hyperparameters for the MLP concatenate. Short term uses higher pos_weight and DS-Rate to handle the larger class imbalance.

Hyperparameter	Range / Categories	Step	ST Pick	LT Pick
WRS	True / False	-	False	True
decremental	True / False	-	True	True
learning rate	1e-5 to 1e-2	1e-5	6.57e-3	8.78e-3
layers	1 to 6	1	2	1
dropout	0.4 to 0.9	0.1	0.7	0.9
batch size	8 / 16 / 32	-	16	8
nb_units	32 to 512	1	52	273
pos_weight	1 to 100	1	19	7
DS-Rate	0 to 12	1	10	5
SMOTE	True / False	-	False	False

Table A.16: The table shows the hyperparameters for MLP using the concatenated data, their ranges or categories and the step size if is a range. The columns ST pick shows the chosen hyperparameter for short term and LT pick the chosen hyperparameter for long term.

		BAcc	Recall	Precision	F1	MCC	AUC	PR AUC	Var(BAcc)
	Train	0.65	0.99	0.51	0.24	0.07	0.9	0.24	0.003
ST	**Val**	0.76	0.73	0.52	0.49	0.16	0.68	0.15	0.002
	Test	**0.56**	0.36	0.51	0.45	0.04	0.6	0.02	0.0
	Train	0.69	0.78	0.54	0.46	0.17	0.78	0.15	0.014
LT	**Val**	0.67	0.68	0.53	0.47	0.15	0.64	0.09	0.014
	Test	**0.57**	0.5	0.52	0.44	0.06	0.56	0.06	0.051

Table A.17: The table shows the results for the MLP concatenated data model. The ST rows shows the results for short term and the LT rows show the results for long term.

MLP Ensemble

Hyperparameter	Range / Categories	Step	ST Pick	LT Pick
threshold	0.1 to 0.5	0.02	0.1	0.66
shrinking	True / False	-	False	True
kernel	linear / poly / rbf / sigmoid	-	linear	poly
c_value	100 / 10 / 1 / 0.1 / 0.01	-	0.1	0.01
degree	1 to 3	1	-	2
DS-Rate	0 to 18	1	18	9
SMOTE	True / False	-	False	True

Table A.18: The table shows the hyperparameters for the MLP ensemble, their ranges or categories and the step size if is a range. The columns ST pick shows the chosen hyperparameter for short term and LT pick the chosen hyperparameter for long term.

		BAcc	Recall	Precision	F1	MCC	AUC	PR AUC	Var(BAcc)
	Train	0.9	0.99	0.54	0.52	0.24	0.98	0.62	0.001
ST	**Val**	0.74	0.64	0.52	0.5	0.14	0.84	0.12	0.004
	Test	**0.63**	0.42	0.51	0.48	0.08	0.68	0.09	0.001
	Train	0.79	0.72	0.6	0.63	0.35	0.86	0.38	0.037
LT	**Val**	0.78	0.72	0.6	0.61	0.33	0.83	0.27	0.061
	Test	**0.78**	0.7	0.59	0.61	0.31	0.84	0.26	0.049

Table A.19: The table shows the results for the MLP ensemble model. The ST rows shows the results for short term and the LT rows show the results for long term.

T-LSTM

Table A.20 presents the HPO picks for T-LSTM. Due to time constrains, the HPO only completed 12 trails for long term and 18 trials for short term, instead of 200 each. T-LSTM takes by far the longest time for each HPO trial, with approximately three to five hours per run.

Hyperparameter	Range / Categories	Step	ST Pick	LT Pick
learning rate	1e-6 to 1e-3	1e-6	3.56e-4	5.16e-4
dropout	0.2 to 0.6	0.1	0.2	0.3
hidden_dim	64 / 128 / 256 / 512	-	512	256
fc_dim	32 / 64 / 128 / 256	-	256	256
pos_weight	1 to 100	1	64	23
DS-Rate	0 to 16	1	6	10

Table A.20: The table shows the hyperparameters for T-LSTM, their ranges or categories and the step size if is a range. The columns ST pick shows the chosen hyperparameter for short term and LT pick the chosen hyperparameter for long term.

T-LSTM is able to achieve good results in short term, even if the HPO only run for a few times. A worse outcome for long term is likely due to even less HPO runs.

Table A.21: The table shows the results for the T-LSTM model. The ST rows shows the results for short term and the LT rows show the results for long term.

		BAcc	Recall	Precision	F1	MCC	AUC	PR AUC	Var(BAcc)
ST	**Train**	0.63	0.42	0.51	0.48	0.08	0.77	0.08	0.006
	Val	0.7	0.52	0.52	0.51	0.13	0.7	0.05	0.004
	Test	**0.69**	0.52	0.52	0.5	0.12	0.76	0.07	0.001
LT	**Train**	0.64	0.64	0.53	0.46	0.13	0.71	0.11	0.04
	Val	0.64	0.62	0.53	0.47	0.13	0.63	0.09	0.031
	Test	**0.59**	0.54	0.52	0.45	0.08	0.63	0.09	0.042

Ensemble

Hyperparameter	Range / Categories	Step	ST Pick	LT Pick
threshold	0.1 to 0.8	0.02	0.14	0.58
shrinking	True / False	-	False	False
kernel	linear / poly / rbf / sigmoid	-	linear	linear
c_value	100 / 10 / 1 / 0.1 / 0.01	-	10	1
degree	1 to 3	1	2	3
DS-Rate	0 to 18	1	16	11
SMOTE	True / False	-	True	True

Table A.22: The table shows the hyperparameters for the stacked ensemble, their ranges or categories, and the step size if is a range. The columns ST pick shows the chosen hyperparameter for short term and LT pick the chosen hyperparameter for long term.

		BAcc	Recall	Precision	F1	MCC	AUC	PR AUC	Var(BAcc)
ST	**Train**	0.97	1.0	0.61	0.66	0.45	1.0	0.97	0.0
	Val	0.79	0.69	0.54	0.54	0.2	0.83	0.06	0.004
	Test	**0.68**	0.47	0.52	0.52	0.14	0.85	0.06	0.002
LT	**Train**	0.86	0.81	0.64	0.68	0.44	0.92	0.65	0.021
	Val	0.84	0.77	0.64	0.68	0.43	0.9	0.65	0.069
	Test	**0.86**	0.83	0.62	0.66	0.42	0.93	0.66	0.042

Table A.23: The table shows the results for the stacked ensemble model. The ST rows shows the results for short term and the LT rows show the results for long term.

Feature Importances

	Feature	Value
0	KreatininHP_month_3	0.100357
1	KreatininHP_month_1	0.095301
2	KreatininHP_month_2	0.093952
3	KreatininHP_month_5	0.083721
4	KreatininHP_month_4	0.082320
5	KreatininHP_month_6	0.073695
6	s_Alter	0.061248
7	ProteinDSU_year	0.057652
8	KreatininHP_year	0.055134
9	ProteinCSU_year	0.053206
10	Dialyse_Anzahl	0.050321
11	p_Alter	0.049368
12	ProteinTUR_month_3	0.048303
13	CRPHP_year	0.047676
14	ProteinTUR_year	0.046699
15	ProteinCSU_month_2	0.046002
16	CRPHP_month_3	0.045468
17	ProteinDSU_month_2	0.043906
18	ProteinDSU_month_1	0.041136
19	ProteinDSU_month_4	0.040022
20	ProteinTUR_month_2	0.039936
21	ProteinTUR_month_4	0.039341
22	ProteinCSU_month_1	0.038242
23	ProteinCSU_month_4	0.038171
24	LeukoEB_year	0.037499
25	ProteinTUR_month_1	0.037469
26	CRPHP_month_2	0.037224
27	LeukoTUR_year	0.036236
28	LeukoEB_month_6	0.036148
29	CRPHP_month_6	0.035874
30	p_Koerpergroesse	0.035673
31	LeukoEB_month_5	0.035463
32	LeukoEB_month_1	0.034747
33	CRPHP_month_1	0.034435
34	ProteinCSU_month_3	0.034100
35	ProteinTUR_month_5	0.033651
36	ProteinCSU_month_5	0.033382
37	ProteinDSU_month_5	0.033226
38	CRPHP_month_5	0.032667
39	LeukoTUR_month_5	0.032572
40	LeukoTUR_month_3	0.031747
41	Dialysis_days	0.031556
42	LeukoEB_month_2	0.030810
43	ProteinDSU_month_3	0.030612
44	Ischaemie_kalt	0.029618
45	ProteinCSU_month_6	0.029240
46	ProteinDSU_month_6	0.028145
47	s_Spendergewicht	0.028126
48	LeukoEB_month_3	0.027983
49	ProteinKSU_year	0.027502
50	s_Spendergroesse	0.027438
51	CRPHP_month_4	0.026901
52	LeukoTUR_month_1	0.026447
53	LeukoEB_month_4	0.026303
54	ProteinKSU_month_5	0.024668
55	ProteinKSU_month_2	0.024371
56	ProteinKSU_month_3	0.022117
57	LeukoTUR_month_4	0.021901
58	ProteinTUR_month_6	0.021836
59	ProteinKSU_month_4	0.021763
60	Primaerfunktion_nein	0.021305
61	MMB_broad	0.020736
62	s_Todesursache_ICB	0.020697
63	LeukoTUR_month_2	0.020164
64	LeukoTUR_month_6	0.018375
65	Anzahl	0.018041
66	Primaerfunktion_ja	0.017989
67	ProteinKSU_month_1	0.017790
68	MM_broad	0.017088
69	p_Grunderkrankung_Analgetikanephropathie	0.016862
70	ProteinKSU_month_6	0.015570
71	MMA_broad	0.014243
72	p_Grunderkrankung_Streptokokken-GN	0.013797
73	s_Todesursache_SAB	0.013557
74	MMDR_broad	0.013087
75	p_Geschlecht_w	0.012583
76	p_Blutgruppe_O	0.012411
77	s_Spenderart_Hirntot	0.012232
78	s_Blutgruppe_O	0.012029
79	p_Blutgruppe_A	0.012015
80	p_Grunderkrankung_FSGS	0.011973
81	p_Geschlecht_m	0.011733
82	s_Blutgruppe_A	0.010965
83	p_Grunderkrankung_Chronische Glomerulonephritis	0.010748
84	s_Geschlecht_w	0.010710
85	p_Grunderkrankung_IgA Nephropathie	0.010544
86	s_Todesursache_Lebendspende	0.010542
87	Ort_CCM	0.009993
88	p_Grunderkrankung_unknown	0.009868
89	s_Geschlecht_m	0.009543
90	s_Spenderart_unknown	0.009240
91	p_Blutgruppe_AB	0.008953
92	s_Todesursache_SHT	0.008737
93	p_Blutgruppe_B	0.008578
94	Ort_CVK	0.008351
95	Organqualitaet_unknown	0.008240
96	s_Blutgruppe_unknown	0.008199
97	p_Grunderkrankung_Alport	0.008062
98	p_Grunderkrankung_Zystennieren	0.008041
99	s_Blutgruppe_B	0.007637
100	p_Grunderkrankung_Einzelniere	0.007280
101	s_Spenderart_living unrelated	0.007191
102	s_Spenderart_living related	0.007007
103	Organqualitaet_Good	0.006792
104	p_Grunderkrankung_Amyloidose	0.006684
105	s_Blutgruppe_AB	0.006665
106	s_Todesursache_unknown	0.005913
107	p_Grunderkrankung_GN	0.005717
108	p_Grunderkrankung_Hypertension	0.005442
109	s_Todesursache_Apoplex	0.004938
110	p_Grunderkrankung_interstitielle Nephritis	0.004722
111	Organqualitaet_Acceptable	0.004319
112	s_Todesursache_hypoxämischer Hirnschaden	0.003918
113	s_Todesursache_Hirnödem	0.003469
114	p_Grunderkrankung_andere	0.003103
115	p_Grunderkrankung_SLE	0.002929
116	p_Grunderkrankung_Pyelonephritis	0.002811
117	p_Grunderkrankung_Diabetes mellitus	0.001953
118	p_Grunderkrankung_Nephrolithiasis	0.001073
119	s_Todesursache_SDH	0.000930
120	p_Grunderkrankung_MPGN	0.000790
121	p_Grunderkrankung_Morbus Wegener	0.000682
122	p_Grunderkrankung_Terminale Niereninsuffizienz	0.000535
123	p_Grunderkrankung_Nierenzellkarzinom	0.000473
124	p_Grunderkrankung_Refluxnephropathie	0.000411
125	s_Todesursache_kardiovaskulär	0.000324
126	Primaerfunktion_unknown	0.000296
127	p_Grunderkrankung_ZYstennieren	0.000280
128	s_Todesursache_andere	0.000272
129	Ort_CBF	0.000264
130	s_Todesursache_Trauma	0.000146
131	s_Geschlecht_unknown	0.000116
132	p_Grunderkrankung_Gichtnephropathie	0.000088
133	p_Blutgruppe_unknown	0.000080
134	p_Grunderkrankung_chron. Glomerulonephritis	0.000080
135	p_Grunderkrankung_HUS	0.000076
136	p_Grunderkrankung_Mesangioproliferative Glomer...	0.000026
137	p_Grunderkrankung_Diabetische nephropathie	0.000000
138	p_Grunderkrankung_ANV	0.000000
139	p_Grunderkrankung_Hämolytisch-urämisches Syndrom	0.000000
140	p_Grunderkrankung_Chron. Niereninsuffizienz	0.000000
141	p_Grunderkrankung_Fokal-Segmentale Glomerulosk...	0.000000
142	p_Grunderkrankung_Fokal segmentale Glomerulosk...	0.000000
143	p_Grunderkrankung_Ciclosporin A	0.000000
144	p_Grunderkrankung_Diabetes	0.000000
145	p_Grunderkrankung_Hypertensive Nephrosklerose	0.000000
146	p_Grunderkrankung_hydronephrose	0.000000
147	p_Grunderkrankung_IGA Nephropathie	0.000000
148	p_Grunderkrankung_IGA nephropathie	0.000000
149	p_Grunderkrankung_hypoplastische Nieren bds.	0.000000
150	p_Grunderkrankung_membranoproliferative GN	0.000000
151	p_Grunderkrankung_membranöse GN	0.000000
152	p_Grunderkrankung_nephrosklerose	0.000000
153	p_Grunderkrankung_polyzystische Nierenerkrankung	0.000000
154	p_Grunderkrankung_refluxnephropathie	0.000000
155	p_Grunderkrankung_schrumpfnieren	0.000000
156	p_Grunderkrankung_glomerulonephritis	0.000000
157	p_Grunderkrankung_diabetische Nephropathie	0.000000
158	s_Spenderart_living unknown	0.000000
159	p_Grunderkrankung_chronische Niereninsuffizienz	0.000000
160	s_Todesursache_non traumatic	0.000000
161	s_Todesursache_suicide	0.000000
162	s_Todesursache_tumor	0.000000
163	p_Grunderkrankung_chronische GN	0.000000
164	p_Grunderkrankung_hypertensive Nephropathie	0.000000
165	p_Grunderkrankung_Vaskulitis	0.000000
166	p_Grunderkrankung_Term. Niereninsuff.	0.000000
167	p_Grunderkrankung_Rapid progressive Glomerulon...	0.000000
168	p_Grunderkrankung_RPGN	0.000000
169	p_Grunderkrankung_Polyzystische Nierendegenera...	0.000000
170	p_Grunderkrankung_Membranöse Glomerulonephritis	0.000000
171	p_Grunderkrankung_Membranoproliferative Glomer...	0.000000
172	p_Grunderkrankung_IgA- nephropathie	0.000000
173	p_Grunderkrankung_IgA nephropathie	0.000000
174	p_Grunderkrankung_chron. Pyelonephritis	0.000000

Figure A.1: All Random Forest Short Term Feature Importance

	Feature	Value
0	KreatininHP_month_3	0.100357
1	KreatininHP_month_1	0.095301
2	KreatininHP_month_2	0.093952
3	KreatininHP_month_5	0.083721
4	KreatininHP_month_4	0.082320
5	KreatininHP_month_6	0.073695
6	s_Alter	0.061248
7	ProteinDSU_year	0.057652
8	KreatininHP_year	0.055134
9	ProteinCSU_year	0.053206
10	Dialyse_Anzahl	0.050321
11	p_Alter	0.049368
12	ProteinTUR_month_3	0.048303
13	CRPHP_year	0.047676
14	ProteinTUR_year	0.046699
15	ProteinCSU_month_2	0.046002
16	CRPHP_month_3	0.045468
17	ProteinDSU_month_2	0.043906
18	ProteinDSU_month_1	0.041136
19	ProteinDSU_month_4	0.040022
20	ProteinTUR_month_2	0.039936
21	ProteinTUR_month_4	0.039341
22	ProteinCSU_month_1	0.038242
23	ProteinCSU_month_4	0.038171
24	LeukoEB_year	0.037499
25	ProteinTUR_month_1	0.037469
26	CRPHP_month_2	0.037224
27	LeukoTUR_year	0.036236
28	LeukoEB_month_6	0.036148
29	CRPHP_month_6	0.035874
30	p_Koerpergroesse	0.035673
31	LeukoEB_month_5	0.035463
32	LeukoEB_month_1	0.034747
33	CRPHP_month_1	0.034435
34	ProteinCSU_month_3	0.034100
35	ProteinTUR_month_5	0.033651
36	ProteinCSU_month_5	0.033382
37	ProteinDSU_month_5	0.033226
38	CRPHP_month_5	0.032667
39	LeukoTUR_month_5	0.032572
40	LeukoTUR_month_3	0.031747
41	Dialysis_days	0.031556
42	LeukoEB_month_2	0.030810
43	ProteinDSU_month_3	0.030612
44	Ischaemie_kalt	0.029618
45	ProteinCSU_month_6	0.029240
46	ProteinDSU_month_6	0.028145
47	s_Spendergewicht	0.028126
48	LeukoEB_month_3	0.027983
49	ProteinKSU_year	0.027502
50	s_Spendergroesse	0.027438
51	CRPHP_month_4	0.026901
52	LeukoTUR_month_1	0.026447
53	LeukoEB_month_4	0.026303
54	ProteinKSU_month_5	0.024668
55	ProteinKSU_month_2	0.024371
56	ProteinKSU_month_3	0.022117
57	LeukoTUR_month_4	0.021901
58	ProteinTUR_month_6	0.021836
59	ProteinKSU_month_4	0.021763
60	Primaerfunktion_nein	0.021305
61	MMB_broad	0.020736
62	s_Todesursache_ICB	0.020697
63	LeukoTUR_month_2	0.020164
64	LeukoTUR_month_6	0.018375
65	Anzahl	0.018041
66	Primaerfunktion_ja	0.017989
67	ProteinKSU_month_1	0.017790
68	MM_broad	0.017088
69	p_Grunderkrankung_Analgetikanephropathie	0.016862
70	ProteinKSU_month_6	0.015570
71	MMA_broad	0.014243
72	p_Grunderkrankung_Streptokokken-GN	0.013797
73	s_Todesursache_SAB	0.013557
74	MMDR_broad	0.013087
75	p_Geschlecht_w	0.012583
76	p_Blutgruppe_O	0.012411
77	s_Spenderart_Hirntot	0.012232
78	s_Blutgruppe_O	0.012029
79	p_Blutgruppe_A	0.012015
80	p_Grunderkrankung_FSGS	0.011973
81	p_Geschlecht_m	0.011733
82	s_Blutgruppe_A	0.010965
83	p_Grunderkrankung_Chronische Glomerulonephritis	0.010748
84	s_Geschlecht_w	0.010710
85	p_Grunderkrankung_IgA Nephropathie	0.010544
86	s_Todesursache_Lebendspende	0.010542
87	Ort_CCM	0.009993
88	p_Grunderkrankung_unknown	0.009868
89	s_Geschlecht_m	0.009543
90	s_Spenderart_unknown	0.009240
91	p_Blutgruppe_AB	0.008953
92	s_Todesursache_SHT	0.008737
93	p_Blutgruppe_B	0.008578
94	Ort_CVK	0.008351
95	Organqualitaet_unknown	0.008240
96	s_Blutgruppe_unknown	0.008199
97	p_Grunderkrankung_Alport	0.008062
98	p_Grunderkrankung_Zystennieren	0.008041
99	s_Blutgruppe_B	0.007637
100	p_Grunderkrankung_Einzelniere	0.007280
101	s_Spenderart_living unrelated	0.007191
102	s_Spenderart_living related	0.007007
103	Organqualitaet_Good	0.006792
104	p_Grunderkrankung_Amyloidose	0.006684
105	s_Blutgruppe_AB	0.006665
106	s_Todesursache_unknown	0.005913
107	p_Grunderkrankung_GN	0.005717
108	p_Grunderkrankung_Hypertension	0.005442
109	s_Todesursache_Apoplex	0.004938
110	p_Grunderkrankung_interstitielle Nephritis	0.004722
111	Organqualitaet_Acceptable	0.004319
112	s_Todesursache_hypoxämischer Hirnschaden	0.003918
113	s_Todesursache_Hirnödem	0.003469
114	p_Grunderkrankung_andere	0.003103
115	p_Grunderkrankung_SLE	0.002929
116	p_Grunderkrankung_Pyelonephritis	0.002811
117	p_Grunderkrankung_Diabetes mellitus	0.001953
118	p_Grunderkrankung_Nephrolithiasis	0.001073
119	s_Todesursache_SDH	0.000930
120	p_Grunderkrankung_MPGN	0.000790
121	p_Grunderkrankung_Morbus Wegener	0.000682
122	p_Grunderkrankung_Terminale Niereninsuffizienz	0.000535
123	p_Grunderkrankung_Nierenzellkarzinom	0.000473
124	p_Grunderkrankung_Refluxnephropathie	0.000411
125	s_Todesursache_kardiovaskulär	0.000324
126	Primaerfunktion_unknown	0.000296
127	p_Grunderkrankung_ZYstennieren	0.000280
128	s_Todesursache_andere	0.000272
129	Ort_CBF	0.000264
130	s_Todesursache_Trauma	0.000146
131	s_Geschlecht_unknown	0.000116
132	p_Grunderkrankung_Gichtnephropathie	0.000088
133	p_Blutgruppe_unknown	0.000080
134	p_Grunderkrankung_chron. Glomerulonephritis	0.000080
135	p_Grunderkrankung_HUS	0.000076
136	p_Grunderkrankung_Mesangioproliferative Glomer...	0.000026
137	p_Grunderkrankung_Diabetische nephropathie	0.000000
138	p_Grunderkrankung_ANV	0.000000
139	p_Grunderkrankung_Hämolytisch-urämisches Syndrom	0.000000
140	p_Grunderkrankung_Chron. Niereninsuffizienz	0.000000
141	p_Grunderkrankung_Fokal-Segmentale Glomerulosk...	0.000000
142	p_Grunderkrankung_Fokal segmentale Glomerulosk...	0.000000
143	p_Grunderkrankung_Ciclosporin A	0.000000
144	p_Grunderkrankung_Diabetes	0.000000
145	p_Grunderkrankung_Hypertensive Nephrosklerose	0.000000
146	p_Grunderkrankung_hydronephrose	0.000000
147	p_Grunderkrankung_IGA Nephropathie	0.000000
148	p_Grunderkrankung_IGA nephropathie	0.000000
149	p_Grunderkrankung_hypoplastische Nieren bds.	0.000000
150	p_Grunderkrankung_membranoproliferative GN	0.000000
151	p_Grunderkrankung_membranöse GN	0.000000
152	p_Grunderkrankung_nephrosklerose	0.000000
153	p_Grunderkrankung_polyzystische Nierenerkrankung	0.000000
154	p_Grunderkrankung_refluxnephropathie	0.000000
155	p_Grunderkrankung_schrumpfnieren	0.000000
156	p_Grunderkrankung_glomerulonephritis	0.000000
157	p_Grunderkrankung_diabetische Nephropathie	0.000000
158	s_Spenderart_living unknown	0.000000
159	p_Grunderkrankung_chronische Niereninsuffizienz	0.000000
160	s_Todesursache_non traumatic	0.000000
161	s_Todesursache_suicide	0.000000
162	s_Todesursache_tumor	0.000000
163	p_Grunderkrankung_chronische GN	0.000000
164	p_Grunderkrankung_hypertensive Nephropathie	0.000000
165	p_Grunderkrankung_Vaskulitis	0.000000
166	p_Grunderkrankung_Term. Niereninsuff.	0.000000
167	p_Grunderkrankung_Rapid progressive Glomerulon...	0.000000
168	p_Grunderkrankung_RPGN	0.000000
169	p_Grunderkrankung_Polyzystische Nierendegenera...	0.000000
170	p_Grunderkrankung_Membranöse Glomerulonephritis	0.000000
171	p_Grunderkrankung_Membranoproliferative Glomer...	0.000000
172	p_Grunderkrankung_IgA- nephropathie	0.000000
173	p_Grunderkrankung_IgA nephropathie	0.000000
174	p_Grunderkrankung_chron. Pyelonephritis	0.000000

Figure A.2: All Random Forest Long Term Feature Importance