Prediction of Permanent Kidney Graft Loss via Deep Ensemble Learning and NLP using a biased Dataset of Electronic Health Records

submitted by

Rouven Reuter

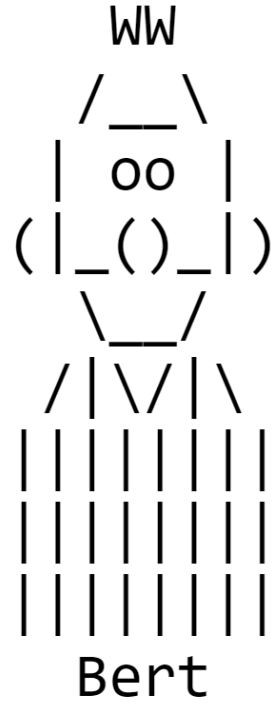
Matr.-Nr.: 896327

of the Faculty VI – Informatik und Medien –
of the Beuth Hochschule für Technik Berlin presented Master Thesis
to acquire of the academic degree
Master of Science (M. Sc.)
in the filed of
Mediainformatics

Date of delivery May 25, 2021

Supervisor / Surveyor

Prof. Dr. Alexander Löser	Beuth Hochschule für Technik
Prof. Dr. Henrik Tramberend	Beuth Hochschule für Technik
Dipl.-Ing. Jens-Michalis Papaioannou	Beuth Hochschule für Technik
Dr. Marcel Naik	Charité Berlin


WW
/__\
| oo |
(|_()_|)
__/
/|\/|\
Bert

Kurzfassung

Hintergrund: Der Beitrag technologischer Errungenschaften im medizinischen Sektor wächst in jeglicher Hinsicht. Auch Anwendungen, die des Machine Learnings entsprungen sind fördern den Fortschritt in diesem Bereich. Die Vorhersage vom Verlust einer Niere als folge einer Nierentransplantation ist Teil dieses Fortschritts. Das Ziel dieser Arbeit ist es eine Deep Learning Model zu implementieren, welches textuelle und tabellarische Daten vereint um den Verlust einer Niere in einem kurz zeit Szenario vorherzusagen. Folglich ist unsere Hypothese, dass die Vorhersage, gegenüber einer Random Forest Baseline, welche nur tabellarische Daten nutzt verbessert werden kann, da wir der Ansicht sind, dass der Inhalt von medizinischem Text komplementär zu dem von tabellarischen Daten ist. Mit der Aufgabe einhergehende Herausforderungen ist die Balance der Klassen des Datensets, sowie unstrukturierter Text.

Methodik: Die Daten wurden durch einer Vielzahl von Maßnahmen gesäubert und aufbereitet. Des Weiteren wurde das State of the Art NLP model BERT genutzt um Embeddings zu erstellen. Als ersten Schritt haben wir zwei Multi-Layer Perceptrons, für die jeweiligen Datentypen, programmiert und dieser eine Mehrzahl von Data-Balancing-Experimenten unterzogen, um schlussendlich ein ensemble model zu erzeugen.

Resultate: Usere Resultate zeigen deutlich, dass das Miteinbeziehn von medizinischem Text die Vorhersage verbessert. Es war uns möglich den F1-Score unserer Baseline um 6% zu schlagen und damit auf einen neuen Höchstwert von 59% F1-Score anzuheben. Zusätzlich konnten wir eine inhaltlich starke Tendenz der Text Daten zum Abstoßen von Nieren offenlegen.

Fazit: In dieser Arbeit war es uns möglich aufzuzeigen, dass klinischer Text komplementäre zu tabellarischen Daten ist und somit eine Verbesserung der Vorhersage von Nierenverlust erreicht werden kann.

Abstract

Background: As technology's contributions to the medical sector rises in several fields, applications of machine learning have also made vital contributions. The prediction of future kidney graft loss following a transplantation is one of them. The goal of this work is it to deliver a deep learning model for the prediction of graft loss in a short-term scenario via using text and tabular data. We hypothesise that combining the text and tabular medical data will improve the prediction over a random forest baseline only using tabular data. Challenges associated with this task are the strong bias towards success in the dataset and the individualistic nature of unstructured text.

Methods: We extensively cleaned and prepared the tabular and text data using several measures. Furthermore, the state of the art NLP model BERT was used to create text embeddings. We first created two separate multi-layer perceptrons and conducted multiple data balancing technique experiments, to lastly build an ensemble model.

Results: Our results show clearly that adding text to the prediction increases the capabilities of our model and outperforms the baseline by 6% to an overall 59% F1-Score. Moreover, we found a strong bias in the text data towards failure, since physicians tend to only note down observations contributing to a disease.

Conclusion: In our work we were able to show that adding clinical text to the prediction of kidney graft loss increases the the scores of the prediction overall.

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

Code Listings

Index of abbreviations

FNN	Feedforward Neural Networks
RNN	Recurrent neural network
ML	Machine Learning
TP	true positive
TN	true negative
FN	false negative
FP	false positive
MCC	Matthews correlation coefficient
AKI	Acute Kidney Injury
ROC	Receiver Operating Characteristics
AUC	Area under curve
BERT	Bidirectional Encoder Representations from Transformers
MLM	Masked Language Model
NDC	National Drug Code
ICU	Intensive care unit
NLP	Natural Language Processing
NER	Name entity recognition
MLP	Multi-Layer Perceptron
EHR	Electronic Health Record
RF	Random Forest
UI	User Interface
WRS	Weighted Random Sampler
CW	Class Weighting
PCA	Principal Component Analysis

1 Introduction

Technology found its way into the medical domain through several successful implementations. Recent examples include the surgical system Da Vinci, supporting surgeons during surgery [9], 3D printed hip prostheses that grant every recipient a replacement that is easily adjusted to their own hip [10] and also the Smart Health Record, which aims to collect the medical history of patients. [11] More specific disciplines like Machine Learning (ML) have proven successful, with ambitions like diagnostics and disease prediction, which are only possible with a vast amount of collected patient data. [12, 13] The methods of machine learning are a valid choice for diagnostics, since making a diagnosis depends on evaluating data, finding coherences and drawing a conclusion from them. Generally speaking a classification task with ML works similarly.

A disease that is commonly targeted by the data science community is the Acute Kidney Injury (AKI). When experiencing an AKI the functionalities of the kidney abruptly stop working, putting the patient into a life-threatening situation. An AKI is traditionally diagnosed through the creatinine blood level and the urine output of a patient. Depending on these metrics it can occur in different stages defined by the RIFLE score. [14] However the technology of machine learning is able to incorporate more variables than the ones used by the RIFLE score and is therefore capable of determining an AKI in either real-time or the future. Several papers on this topic have already been published with varying success, [15] showing that with the right data and algorithm a solid model can be created.

1.1 Motivation

Like most internal organs the kidney is a vital one. Besides synthesising urine and routing it through the body, the kidney conducts several more tasks. Filtering toxins from the bloodstream, as well as excess water to eject as urine. Furthermore, the kidney regulates the acidity measured in the cardio vascular system and also produces hormones like Erythropoietin to control the production of red blood cells in the bone marrow. Additionally, the concentration of minerals such as potassium, sodium and calcium are influenced by the kidney. [16] However to support a sustainable cohabit with an owner and its kidney some resources are needed.

The Resource Physician According to the OECD, 357.401 individuals, or roughly 0.43% of the German population, where actively practicing as a physician in 2018. Even though Germany has a high per capita number of medical doctors compared to other European countries (Italy 0.33%, France 0.31%, UK 0.28%), there is still a shortage of medical personnel in the country. The overall

number rose by about 50% since 1993, which is unproportional compared to the population growth of 2.7%. However, due to an ever aging population, more doctors work part-time and the burden of bureaucracy, a growth rate of 50% is not sufficient. Technology could be the key to compensating for the lack of medical professionals. [17, 18]

Escaping Death has its Cost Considering the vast duties the kidney fulfills on a daily basis, a End-Stage Renal Disease is a life-threatening condition. Nevertheless the lifespan of a patient can be extended through the process of dialysis, which substitutes the kidney and supplements the tasks a damaged kidney can no longer perform by purifying the blood. [16] While the artificial blood purification will extend the life expectancy of a patient by 10 to 20 years, the annual costs for dialysis treatment are around 40,000€. Alternatively, a kidney transplant allows most patients to experience double the lifetime and the annual costs are only one third as those of dialysis treatment (except for during the year of the transplantation). [19, 20] Additional life time is the main reason for choosing transplant over of a permanent dialysis treatment, but transplantation can have several other drawbacks.

The Resource Kidney Regarding the risks and resources of a kidney transplant, the 2017 Annual Data Report for kidney shows optimistic numbers related to short- and long-term unadjusted allograft survival and a decline of the adult waiting list. Likewise, the number of successful deceased donor kidney transplants grew. Moreover, the number of living donor transplantation, which offer a higher survivability than a deceased donor transplant, keeps falling, but the absolute number of transplants rose. Despite the waiting list length dropping from a all time high, an shortage of donor kidneys still persists and the average time frame for a patient to receive treatment is three years. [21]

Lucky enough to get one, but... Even when a patient is able to receive a kidney transplant, acute kidney injury, which is associated with allograft loss and patient mortality, is common. The reasons for an AKI after a transplantation are manifold and include, for example, acute cellular rejection or a renal artery thrombosis. [22] The probability for a hospitalized patient to develop an AKI is between 2% and 18%, whereas these numbers are even higher for patients in the intensive care unit, ranging from 22% to 57%. Alarmingly high is the number of cases that go undetected, ranging from 57% to 75.6%. [23] Direct 11% of patients decease due to the fact of not detecting an AKI early. [24] However, detecting of an AKI is difficult and mostly possible by observing a loss of organ function. [25]

Conclusion Despite the fact that the absolute number of kidney transplants is at a historic high and the waiting list is shrinking, transplant success is still vital, since a scarcity of available organs persists. Also the number of living donor transplants dropped, which represents a decline in the quality of transplants since the majority of donor organs are deceased donor kidneys. The high percentage of patients who develop AKI is as alarming as the amount of undetected AKIs. The high rate of undetected AKIs, however, is understandable as a wide multitude of factors play a role in the development of an AKI and most commonly damage can only be detected with certainty when a loss of organ function is present. On top of all of this, the lack of full-time medical experts makes predicting graft loss even more difficult. The motivation of this thesis is it to reduce the number of undetected AKI cases and to compensate for the low number of medical staff through

the means of a data driven prediction system that helps prevent a possible demise of patients and in doing so, saves cost and possibly extends patients' life expectancies.

1.2 Hypothesis

Observing the baseline model used in the task prior to this thesis. [26] A prediction was performed using only numerical and categorical features. However T-Base holds additional data, as the data used in the Random Forest baseline approach. The spared data includes a vast amount of unstructured clinical text. Taking this into account the hypothesis of this thesis are:

1. That medical text holds information complementary to the tabular data of the patient and will therefore, by combining the two data types, improve the performance of the model.
2. An MLP model using a balancing technique will outperform a RF model on the same imbalanced dataset.
3. Further pre-training BERT with AKI risk-factors that will improve graft loss prediction on clinical notes.

The hypothesis will further be refereed to as fist, second or third hypothesis.

1.3 Problem Definition

There is already a multitude of papers on AKI prediction published. However most of the published papers concentrate on categorical and numerical values, such as age, gender, ethnicity and creatinine, urine output, blood pressure. [15] Few papers do a prediction based on text corpora form electronic health records and numerical, as well as categorical values. [12, 27] An absolute absence of research can be observed when it comes to German clinical text for the prediction of AKI. This might be because of strict European data privacy laws and a insufficient digital infrastructure, however for this thesis the Charité Berlin offered to provide a data collection of patients that underwent a kidney transplantation.

The Baseline Prior to this thesis, work has already been conducted on the dataset in terms of Graft loss prediction. [26] This work is a continuation of it. The task stays the same, but the results will be improved through the addition of further methods and features.

Challenges Given the dataset the goal is to predict the last possible stage of an AKI - Graft loss - after a kidney transplantation for a short term scenario. The short term time span is defined as 12-18 Months after the date of transplantation. This means that data only from the beginning of a patients medical history, until one year after his/her transplantation took place can be used and a prediction of six months into the future needs to be made.

The dataset provided is strongly imbalanced towards successful transplantation's, making it difficult to learn the failure class. Further the number of examples in the dataset it rather small compared with the task and expected input vector.

The appropriated text corpus is unstructured and drafted by multiple people, which leads text that mean the same but are structured and written completely different. Additionally spelling mistakes or personal text stylistic choice might keep the model from learning. Lastly unwanted, nonsensical or repeating text chunks can also be a obstacle, rising the size text embedding or dilute the important information.

Solutions To prove our fist hypothesis a model that is able to handle both types of data is needed. We decided to use an ensemble architecture, to combine the embeddings. Further a technique to handle the imbalance of the dataset is need and a text cleaning pipeline to make the text as uniform as possible. The second hypothesis will be proven by working towards the first one, as the tabular MLP model will be part of the ensemble model. For proving our third hypothesis we will fine-tune a BERT model on the RIFLE risk-factors and evaluate of the predictions made will improve.

Main Contributions In the course of the thesis we provide multiple contributions (i) An approach to clean medical text data, (ii) An MLP model to combine text and tabular data, (iii) introduce a new NLP fine-tune task using AKI risk-factors.

1.4 Methodology

In this part the methodology of this thesis will be briefly outlined. A full-fledged overview of the used methods can be found in chapter 4.

Data Preparation Measures When exploring the text corpora of the available data, the state of the data was insufficient for conversion to text features. Several measures have been taken to reduce the found flaws. We used various regular expressions to remove information that holds no value for the prediction. Furthermore, we replaced or removed unknown or nonsensical characters and replaced medical abbreviations with their fully worded counterparts and much more. For the preparation of the tabular data, a pipeline provided by our baseline was used.

Baseline As our baseline we chose a thesis prior to ours, which uses the same source data. [26] The model of choice in this case was a Random Forest.

Experiments To generate text embeddings a model is needed that transforms plain text into processible representations for machine learning models. There are several different methods to conduct such a conversion. Well known models include Word2Vec [28], FastText [29] or GloVe [30].However we decided to go for the state-of-the-art model BERT [3]. BERT can be retrained for different languages as well as fine-tuned for specific tasks. To find the best fitting variety of BERT, a comparison of the BERT models gBERT-base [6], base-bert-german [5] and MedBert[1] has been conducted on the medical corpus at free disposal during the project. Several data balancing techniques and strategies to handle long text have been compared. The same balancing techniques have been applied to the tabular model.

Models Furthermore, we built individual models for both of the data types given and applied the best-suited techniques evaluated through the experiments. To finally combine their embeddings to

[1] https://huggingface.co/smanjil/German-MedBERT

form an ensemble model which considers both data types.

1.5 Thesis structure

The following passage describes the content and structure of this thesis.

In **Chapter 2** the fundamental important to understand the contents of this thesis are covered. First, an explanation of metrics to evaluate classification algorithms is given. Next, a short description of model architectures like Random Forest and feedforward networks is given. Finally, we will cover the relevant balancing techniques used in the experiments.

The **3rd Chapter** is primarily concerned with the data used. We first go over the database provided and then cover our cohort selection as well as the selection of features. During this thesis a temporary task was given to us as we were unable to work on the actual data. We analysed the MIMI-III dataset and our findings are laid out in section 3.4.

Chapter 4 is the main part of this work. Here we cover all the approaches we implemented. Starting with the baseline model and going on with the implementation of our tabular model to the preparation of the text data and model, as well as fine-tuning, and ending it with our attempt of the ensemble model.

Chapter 5 covers the environments used by us and their capacity. Furthermore, a full data flow diagram will be explained.

In **Chapter 6** the results of all experiments will be examined. Furthermore, an error analysis will be conducted. For all results we first explain them and then give an interpretation. Firstly, the results produced by the tabular model will be investigated and interpreted, followed by the text and ensemble model. Finally, the results of iteration two text and the fine-tuning will be revealed.

Chapter 7, the final chapter, provides a discussion of the whole project covering things that went well and not so well. Furthermore, we give an extensive future outlook and suggestions for improvement.

2 Fundamentals

In this chapter the fundamental knowledge needed to understand this thesis is discuss. First metrics relevant for the evaluation of a classification model are explained and followed up by the types of model architectures. Lastly a deep dive into the data balancing techniques, considered to solve the task, is conducted. Many of the concepts introduced in this section will reappear throughout the thesis.

2.1 Evaluation Metrics for Classification

When a binary classification Machine Learning (ML) model has been implemented, its effectiveness and efficiency need to be measurable to make the performance of the ML model comparable. There are established metrics which are approved by the scientific community, however, there is no consensus on a unified measure. [4] With this being the case, the chosen evaluation methods cover widely used metrics like the F_1-Score but rarer metrics that add more points to the evaluation, like the Matthews correlation coefficient, in hopes of creating a more sustainable evaluation.

2.1.1 Confusion matrix

The confusion matrix forms the foundation of the evaluation of binary classification. To define the confusion matrix, consider a dataset where every element of the data is either classified as positive or negative. The given classifications are deemed to be the ground truth. Further consider a classifier operating on the given dataset. The classifier's task is to estimate which label (positive or negative) to assign to an element of the set. If the label assigned by the classifier is equal to the ground truth label, the comparison between these labels will be considered true. Should the classifier miss-classify (confuse) an example, that example will be considered false. Table 2.1 can be derived form this definition. [31]

The values established by the confusion matrix (TP, TN, FN, FP) can be further used to calculate relevant metrics for the assessment of classification algorithms. Table 2.2 shows the basic metrics and their formulas.

In the context of this paper, only recall and precision will be relevant. Recall can be defined as the probability that a randomly chosen instance will be predicted to be positive, whereas precision can be described as the probability that a randomly chosen, predicted instance (positive) will be relevant. Using recall and precision it is possible to calculate the first metric - accuracy. [32]

	Predicted positive	Predicted negative
Actual positive	True positives TP	False negatives FN
Actual negative	False positives FP	True negatives TN

True positives (TP) and true negatives (TN) are the correct predictions, while false negatives (FN) and false positives (FP) are the incorrect predictions

Table 2.1: Confusion matrix [4]

Sensitivity, recall, true positive rate	$= \frac{TP}{TP+FN} = \frac{TP}{n^+}$	Specificity, true negative rate	$= \frac{TN}{TN+FP} = \frac{TN}{n^-}$
Positive predictive value, precision	$= \frac{TP}{TP+FP}$	Negative predictive value	$= \frac{TN}{TN+FN}$
False positive rate, fallout	$= \frac{FP}{FP+TN} = \frac{FP}{n^-}$	False discovery rate	$= \frac{FP}{FP+TP}$

TP: true positives. TN: true negatives. FP: false positives. FN: false negatives

Table 2.2: Classwise performance measures [4]

2.1.2 Accuracy

Accuracy can be calculated through a formula that uses all the values provided by the confusion matrix.

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

The value provided by this formula ranges from $[0, +1]$ (1 equals a perfect classification rate) and can be interpreted as the ratio between all data instances that have been labeled true by the classifier and all instances of a given dataset. As simple and logical as this metric seems, it fails to provide an accurate score for unbalanced datasets (an unbalanced dataset is one which houses a significantly larger number of data pieces that are labeled with the same label) since, when a dataset is strongly biased towards one label, the algorithm will learn to classify those labels effectively, hence the true positive or true negative value will be high and predominate in the overall calculation. [4]

2.1.3 F1-Score

Statistics defines the F_1-Score the measurement of the accuracy of a test. It is widely used to measure the performance of binary and multi classifications of machine learning algorithms. [4] The result of the F_1-Score ranges between $[0, +1]$ and can be interpreted as the weight average of the metric's precision and recall. Mathematically the F_1-Score can be calculated as the harmonic mean of precision and recall. The following is the formula of the F_1-Score. [33]

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

Although the F_1-Score is frequently used to evaluate confusion matrix base Machine Learning models, it has two deficiencies. Firstly, the F_1-Score does not take into account samples correctly classified as negative. Secondly, the F_1 is susceptible to class swapping (class swapping refers to swapping the positive and negative labels of an dataset). The second issue can be overcome by a macro/micro averaging procedure, nevertheless this technique is biased. [4]

Despite the flaws of accuracy and the F_1-Score, the decision was made to include them as metrics since they are still frequently used and appear as measurements in the baseline.

2.1.4 Fbeta-Score

For some evaluations recall might be more important than precision. For example, when predicting disease, where more attention is directed towards predicting the disease rather then a healthy individual, since miss-classifying a healthy person as sick has no repercussions in most cases.

$$F_\beta = (1 + \beta^2) \cdot \frac{precision \cdot recall}{(\beta^2 \cdot precision) + recall}$$

The Fβ-Score offers the possibility to either weight recall or precision higher. By increasing β the importance of recall rises and would be weighted twice as important as precision. Lowering the β has a contrary effect - less weight to recall more to precision. [34]

2.1.5 ROC-AUC

The ROC (Receiver Operating Characteristics) curve visualizes the ratio of the sensitivity and specificity (see table 2.2) for a classification task. The values of the curves coordinate system range between $[0, 1]$, whereby the x-axis represents the specificity and the y-axis represents the sensitivity. To compute the curve a threshold is used to determine how an element should be classified. Calculating the sensitivity and specificity for every threshold and plotting the results will issue the ROC curve. [35]

The AUC (Area under curve) score can be used to evaluate the efficiency of classification models and it can be calculated through the means of the ROC curve. [36] The highest possible AUC score is a 1, which can be interpreted as a perfect classification. While the lowest number to score

is a 0.5 which is the equivalent of random guessing. [35] Realistically, the value cant be smaller than 0.5, since random guessing produces a diagonal line between $[0, 0]$ and $[1, 1]$. The important statistical quality of the AUC is that the AUC of a classifier is equivalent to the probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance. [37] According to Fawcett et al. the AUC score can be calculated using a trapezoidal rule base algorithm shown in [37].

The ROC-AUC addresses the problems mentioned for accuracy [38] and was also included in the baseline.

2.1.6 Matthews correlation coefficient

To address the issues connected to the F_1-Score Chicco and Jurman suggest the use of the Matthews correlation coefficient (MCC) for binary classification algorithms. According to their work, the MCC is the only metric that produces a high score only when classifications, both positive and negative data instances, were predominantly predicted correctly.

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

The result of the formula ranges from $[-1, +1]$. A 1 is consider to be a perfect classification, while a -1 is a complete misclassification. [4]

Since the dataset that was used in this thesis also has a strong tendency towards patients with a successful transplantation, the decision was made to include the MCC.

2.2 Random Forest

The general idea of decision trees is to split a dataset into two portions based on a certain rule until a predefined condition has been met. The task of a decision tree is dependent on the predefined stopping condition and the rule set on how to split the data, so it can be used for regression or classification. A strong flaw of decision trees is that they are prone to overfit, thus are not able to generalize very well. [39] Following up on the concept of the decision tree, Breiman published the random forest paper in 2001. [40] In his work he proposed an ensemble decision tree learning algorithm that averages the prediction of a multitude of trees to calculate its final decision. Furthermore, the trees use a bootstrap aggregation strategy (bagging), which makes overfitting less likely. This strategy is based on using bootstrap samples rather than the originals. Figure 2.1 shows the structure of a random forest. [39]

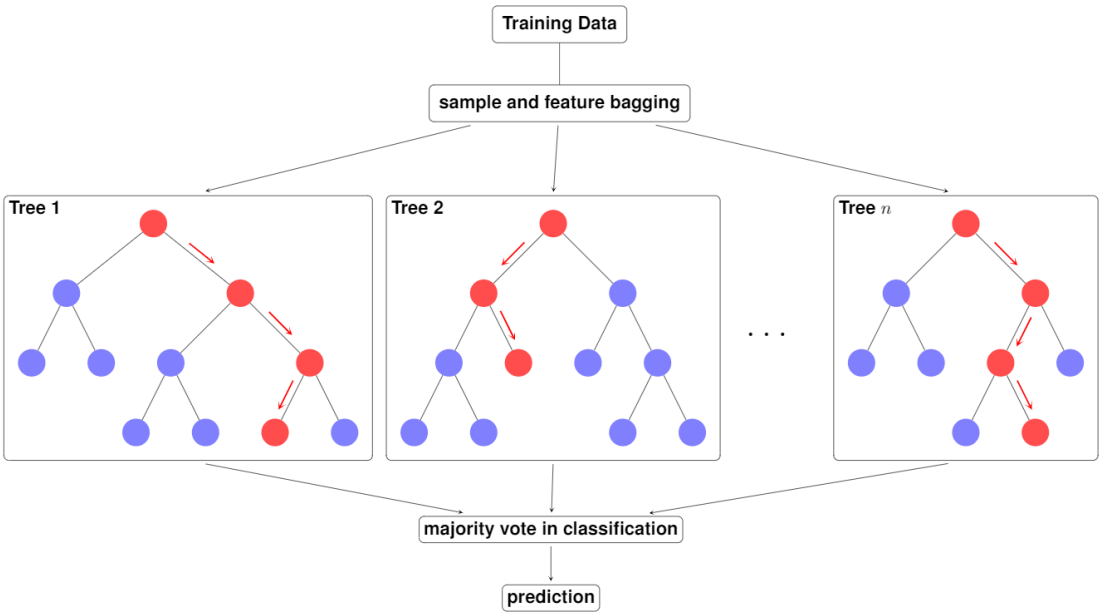


Figure 2.1: Illustration showing the mechanism of a random forest. The figure shows how the data flows through a RF model and finally does a voting on the classification.

2.3 Deep Feedforward Neural Networks

The term deep feedforward neural network, also synonymously used as multilayer perceptron (MLP) or just feedforward neural network, covers a concept which aims to approximate a function f*. MLPs are the principle deep learning models. The first instance of an application using an MLP can be dated to the year 1943, by Warren McCulloch and Walter Pitts. [41] [] Since then MLPs have been used for countless tasks. A generic example includes a classifier that categorises the input x to a category y using the function y = f*(x). More specifically, an MLP learns the values of the input θ for a mapping y = f(x;θ) to approximate the most fitting function. To differentiate them from recurrent neural networks, FNNs have no feedback connections. [42]

Feedforward Neural Networks Architecture MLPs are called networks, because they are comprised of a multitude of functions that follow in succession. The functions hosted by an MLP are referred to as layers, the first layer is called the input layer, while the last layer is considered to be the output layer. Figure 2.2 visualizes the architecture of a simple MLP. The individual layers are made up of perceptions or neurons. The input layer usually has as many perceptions as a given task provides parameters and it can not derive from that. On the other hand, the output layer, at least of a classification task, has the same number of perceptions and classes. In between those two are what is called the hidden layers. The hidden layers can be of any number of perceptions, however with higher numbers comes a higher processing time. To let data flow to the network, each layer is fully connected to the next one.

Furthermore, each neuron has a weight and activation function and each connection also features a weight. Taking all the outputs of the previous layer, the product of the weights and inputs are passed to the activation function which "squashes" the result. [42] There are several possible

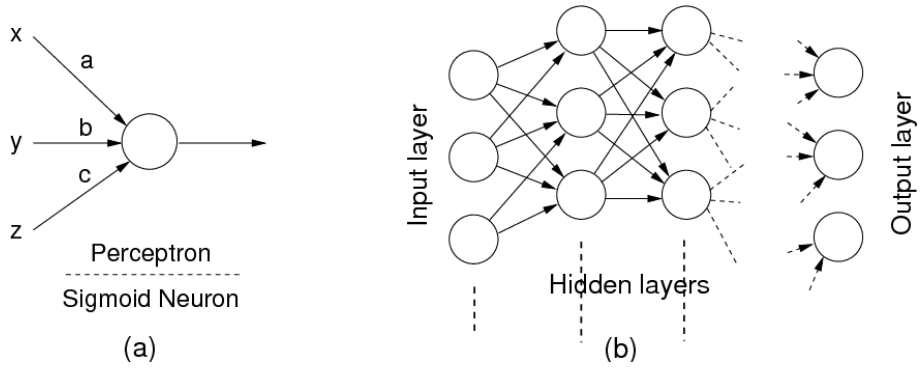


Figure 2.2: (a) Perceptron or Sigmoid neuron, (b) Model for Artificial Neural Net. [1]

activation functions that can be applied. The most common ones are ReLu, Sigmoid and Softmax. [43] A full illustration of the multi-layer perceptron can be seen in figure 2.2.

Neuroscience inspired the idea for the feedforward network, as it is supposed to mimics the neurons in the human brain. [42]

Cost Function As calculating the perfect weights for a FNN is impossible since there are too many unknown variables, a cost function, also referred to as loss function, is used to calculate the model's error rate and adjust the weights via back propagation. [43]

Gradient-Based Learning Adjusting the weights of the perceptions is possible by using an optimization algorithm that is referred to as gradient descent, which arranges the weights of the neuron by minimizing the result of the cost function. An opportunity for a gradient descent based learning approach always arises when applying linear algebra fails. The search for the optimal weights is an iterative process, which, when repeated often enough, will lead to the minimum cost for the task and thus will produce the optimum weights.

2.4 Transformers

The model architecture of a transformer (figure 2.3) draws on the dependencies between the input and output of an encoder-decoder structure and self-attention mechanisms. Transformers are feedforward networks that are highly parallelizable, thus abolishing the constraint of sequential computation seen in recurrent neural networks (RNN). The original implementation by Vaswani et al. beat multiple state-of-the-art translation models while only needing a fraction of the processing power and time for training their model. [2]

The attention mechanism has been introduced by Bahdanau et al. to deal with the major bottleneck of fixed-length vectors in the context of machine translation seen in RNN sequence to sequence models. The problem usually appears when translating sentences that are longer then most sentences in a training set. Bahdanau mitigated the problem by calculating hidden states during the encoder phase for each part of the sequence and passing those to the decoder stage. The decoder is then able to create context vectors from the hidden states and is thus able to compute long sentences. [44]

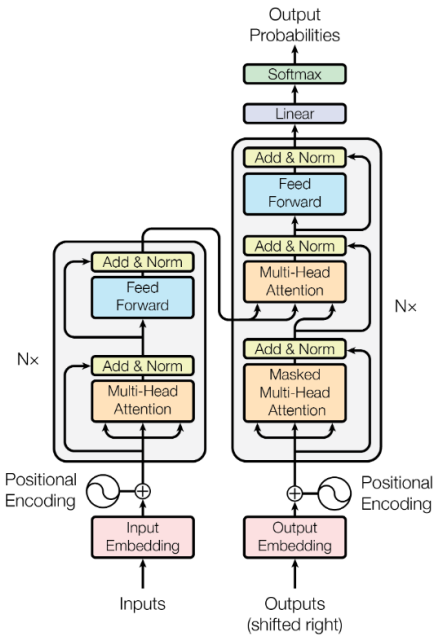


Figure 2.3: Concept of the transformer architecture as proposed by [2]

2.5 BERT

Building upon the blocks of Vaswani's work [2], Devlin et al. published the NLP model bidirectional encoder representations from transformers (BERT) in 2018. [3] Arguing that techniques used for per-taining by language models back then were not optimal due to their unidirectionality, Devlin proposed a bidirectional fine-tuning approach. Fine-tuning usually refers to a pre-training practice where a model first gets trained on a general task and, in a second step, fine-tuned on a particular task. [45]

The architecture of BERT is almost identical to the design of Vaswani (figure 2.3), differences can be noted on the output layers or heads. However, the major difference to [2] can be found in its approach to training the model. Two unsupervised tasks are assigned to the model.

- **Masked language model** - Applying MLM for training includes masking 15% of tokens randomly selected with the [MASK] token. Then BERT is tasked with predicting those tokens, based on their surrounding context. However, a discrepancy between fine-tuning and pre-training is that during fine-tuning the [MASK] token is not available. To alleviate the negative effects of this, selected tokens do not always get replaced wit the [MASK] token. Instead only 80% of selected tokens get replaced with [MASK], 10% get substituted with a random token and 10% are left unchanged.
- **Next sentence prediction** - The objective of next sentence prediction consists of predicting the next sentence from a pair of sentences. 50% of the time the given sentence is actually the subsequent sentence of the sentence that has been looked at and the other 50% of the time it is a random sentence from the corpus.

Another peculiarity is how BERT treads word embeddings. When encoding corpora, BERT has four more tokens that are used to point out certain conditions. As seen in figure 2.4 BERT uses the special token [CLS] to announce the beginning of a sequence and to separate the beginning of a new sentence. In a series of multiple sequences the [SEP] token is used. Furthermore, the [CLS] token can be used for text classification tasks. [46] Additionally, a segment embedding is added to every token to state the belonging of that token to a sentence. Other special tokens include the [UNK] token, which indicates that BERT is unable to tokenize a word due to character illiteracy problems and the [PAD] token used for padding when the lengths of token embeddings in a batch are not uniform. Lastly BERT uses a sub-word representation for unseen words whereby words get split into smaller sub-words or characters. To mark a partial representation of a word two ## are used. An example is displayed in figure 2.4 by the tokens ”play” and ”##ing”. Only words that are not part of the maximum of 30,000 word vocabulary of BERT are represented as sub-words.

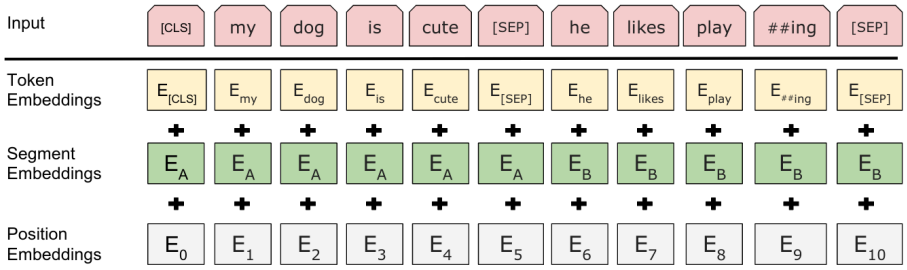


Figure 2.4: Representation of words embedded by BERT [3]

At the time point of publishing BERT achieved several state-of-the-art performance tasks, including GLUE (General Language Understanding Evaluation), SQuAD (Stanford Question Answering Dataset) v1.1 and v2.0 and SWAG (Situations With Adversarial Generations). [3]

2.6 Data Balancing Techniques

Datasets that emanate from real world scenarios tend to be imbalanced. This is especially true for medical dataset that label their examples for certain illnesses. Most of the time medical data collections have a high imbalance towards healthy individuals. Trying to predict the minority in such datasets is reasonably difficult since every model will have a strong bias towards the negative class. To remedy the imbalance, two approaches are possible - algorithmic and data-based. While algorithmic approaches mostly try to manipulate the loss of the model towards the minority class, the data-based approaches use techniques over-, under- and combined sampling. Since our dataset also has a strong imbalance towards success, managing this imbalance was inevitable.

2.6.1 Focal Loss

The focal loss is an imbalance-managing adaptive loss function based on the cross entropy loss. As the certainty of predicting a class rises, the scaling factor can decrease to zero. This leads to a lower learning contribution of the network for those classes. Through that the focal loss allows for

scaling up the focus on difficult-to-predict classes, thus making it a suitable choice for handling imbalanced datasets. The focal loss is defined as:

$$\mathbf{FL}(pt) = -(1-pt)^{\gamma} log(pt)$$

γ is the additional factor that differentiates it form the cross entropy loss. According to the original paper $\gamma > 0$ increases the loss for difficult-to-classify classes while reducing it for simple ones. We included this method because, in its original paper, it shows promising results. [47]

2.6.2 Class Weighting

Class weighting is another approach to balancing the learning of classes through manipulating the loss based on class rarity. It is also based on the cross entropy function, however, its formula is different from the focal loss.

$$\mathbf{WCE} = -(\beta y log(\hat{y}) + (1-y) log(1-\hat{y}))$$

Generally, the idea of the focal loss and the weighted class entropy (WCE) loss is the same - learn difficult classes by penalizing frequent classes. The weighting factor β can either be scalar or a vector, depending on the count of classes to predict. By utilizing the factor as > 1, false positives get favored, whereas setting it to < 1 favors false negatives. This technique was included in our research since it is easy to implement and has already seen its use in the medical sector. [48]

2.6.3 Weighted Random Sampling

When randomly selecting items from a data collection where each item has the same chance to be selected this is called uniform sampling. However, there are instances where groups or specific items should be selected more or less frequently when sampling. To achieve this behaviour, weights can be assigned to each data point, thus making a selection more or less likely. This procedure is called weighted random sampling. Using this technique, the bias in imbalanced datasets can be reduced by assigning a higher probability of selection to the minority classes and, in doing so, increasing the likelihood of learning those classes in their entirety. This technique was implemented in our research to have a statistical mean of balancing a dataset. [49]

2.6.4 SMOTE (Synthetic Minority Over-Sampling Technique)

While the previously mentioned techniques were based on loss or statistical means, SMOTE is a technique that can be used to generate samples for a dataset and therefore balance it out. Work prior to SMOTE has shown that oversampling by replacement does not increase prediction of the minority class, since the decision region of the classifier gets smaller. Therefore, SMOTE oversamples the dataset by creating synthetic examples, which it does by combining data from its

k nearest neighbors of the rare class. More specifically, the SMOTE algorithm works by taking the k nearest neighbors, creating a sample in course of those. Then the difference between the newly created sample point and its neighbors is taken, followed by a multiplication of the difference with a random value from 0 to 1. The difference then needs to be added to the synthetic feature vector. Through this approach, a new example is created between two already existing examples, causing the oversampling to be more general and the decision region extensive. [50] Further research on this method came up with variances of SMOTE.

SMOTE BL In a classification dataset most of the examples (if the dataset is of good quality) can be clustered into a region. The outer rim of this region is called the borderline. Examples located on and close to the borderline are difficult to classify since they diverge strongly from the core. SMOTE BL (BorderLine) focuses on creating synthetic examples only with borderline data points. Experiments conducted by Han et .al show that SMOTE BL outperforms the original SMOTE. [51]

SVM SMOTE Support Vector Machines (SVM) are a common tool used in the machine learning realm, however the SVM algorithm suffers significantly from imbalanced datasets. To alleviate this weakness, Akbani et al. [52] implemented a SMOTE flavor that uses the different error costs algorithm by Veropoulos et al. [53] The implementation accomplishes the following idea: The different error costs function pushes examples of the minority class further away from the borderline so that examples can be more easily classified and SMOTE takes the densely distributed examples to better define the borderline.

SMOTE ENN Edited Nearest Neighbor (ENN) is a technique which eliminates classes that are a minority in their neighborhood. It removes examples that have a different label from at least two of their three nearest neighbors, thereby making the decision region more homogeneous and easier to differentiate from other classes. ENN gets used prior to SMOTE. [54, 55]

SMOTE Tomek The Tomek algorithem, as ENN, is a technique used prior to SMOTE to clean singular data points from the dataset. To achieve this, Tomek uses two examples of different classes (a and b) and estimates the distances (d) between them and one other randomly selected sample (z). If the criteria $d(x, y) < d(y, z)$ or $d(x, y) < d(x, z)$ are met, a Tomek-Link gets established and both of the examples are removed. [56]

2.6.5 ADASYN (Adaptive Synthetic Sampling)

As a basic idea, ADASYN uses a weighted distribution individually for every minority class example, depending on their complexity when it comes to predicting the example. For examples that tend to be difficult to predict, ADASYN generates more synthetic data to reduce the bias on subsets of the minority classes that are very well represented. [57]

2.7 Summary

In this chapter, we covered all the necessary fundamentals for understanding this thesis. We first took a look at relevant metrics for evaluation classification algorithms and examined them from the most simple, like the confusion matrix, to the complex, like the F1-Score. We then conducted a deep dive into the different models used and investigated the Random Forest, MLP and Transformer models. Lastly, a description of data balancing approaches we covert.

3 Dataset Description

This chapter covers the data sources and the data itself. The first section will refer to the to the Charité provided database and explain which tables and entries have been used in this work. The following section describes and reasons the process of sectioning the cohort for this project. Lastly, an analysis of two datasets has been conducted, the dataset derived from T-Base, on which this thesis is based, and the Mimic-III dataset, which was given to us as a temporary task.

3.1 T-Base

The T-Base database was provided by the Charité of Berlin to explore the possibility of predicting kidney graft loss after a transplantation in long- and short-term scenarios. The database explicitly holds data of patients who underwent a transplantation of any sort. Not only does the database keep track of the recipient of the transplant but also the donor, as information about the donor is also valuable to guarantee a successful transplantation. In detail, the database hosts personal information about the receiver and donor, like height or age. Furthermore, the medication a patient receives, the diagnoses made by physicians, laboratory blood and urine values, information about the transplant procedure and much more. Figure 3.1 shows the database in its entirety.

Tabular Data The tabular data that has been used for the prediction task comes from the tables Transplantation, Patient, Donor, Laborwert, Medication, Diagnose, Verlauf and Dialyse. All data extracted from these tables was either numerical or categorical.

Text Data The text extracted from T-Base can be located in two tables, Untersuchung and Verlauf. Textually, the Verlauf corpus holds information about doctors' assessments. It consists mostly of observations a doctor made and is apportioned into three columns of the table, Beurteilung, BeurteilungAerztlich and BeurteilungIntern. Untersuchung, on the other hand, only has one column of text, however the text comes in multiple categories and is vastly different for each of those. Its contents can be highly structured, like a machine-written laboratory report mostly consisting of tables, but also highly unstructured in the form of continuous text originating from a physician's survey.

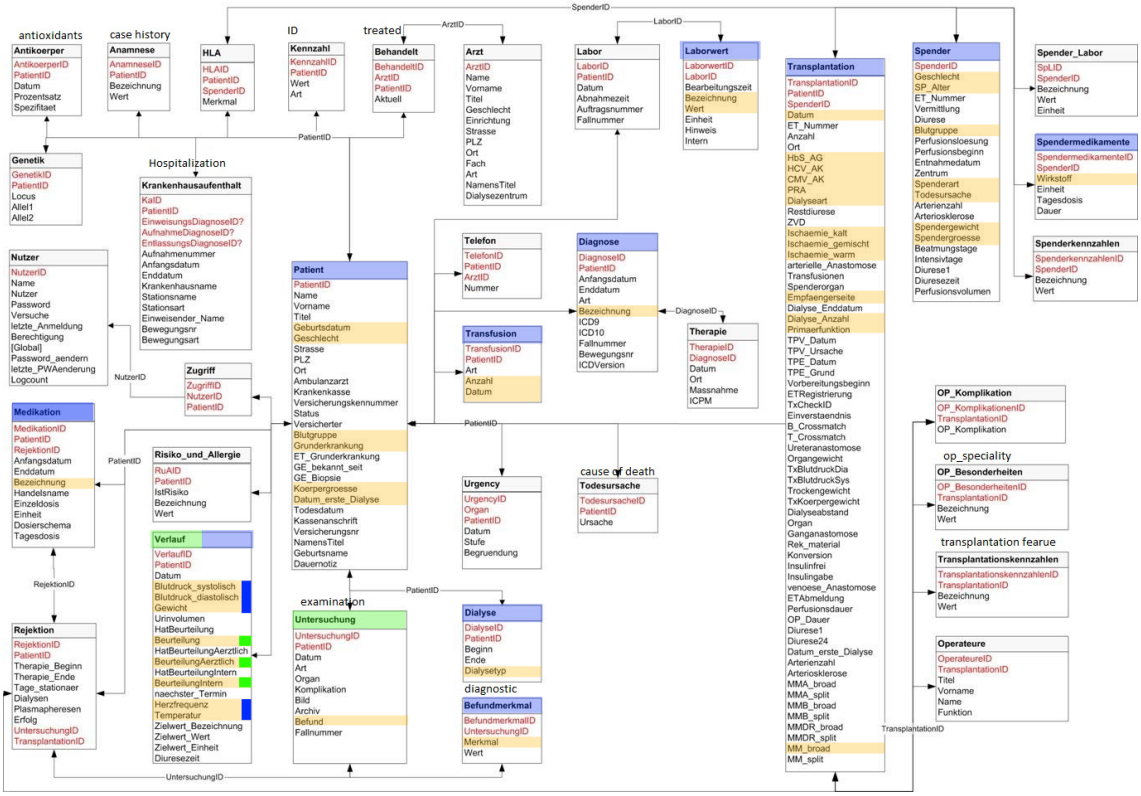


Figure 3.1: T-Base database structure. This figure shows the entire database. To simplify it the headers of tables used as features for the tabular model are colored blue, while tables used for the text model are colored green. Individual columns used are colored orange.

3.2 Cohort Selection

The decision on which patients should be part of the cohort was made by medical staff. The rules created to form the cohort were made with medical and practical intentions. At the time when this project was conducted, the total number of kidney patients found in T-Base was 19642. We applied the following selection rules to them:

- Only patients with one or more kidney transplants, but no other kind of organ transplant.
- Discard all patients younger than 18
- Transplants before the year 2000 were not included
- The transplant must have been conducted at the location of Charité Mitte or Virchow
- Patients who had a failing transplant within 12 months after transplantation were not considered

Applying those rules 1263 patients were a fit, which resulted in 1265 possible transplants to predict. Patients under 18 were excluded, since aftercare for them is provided by the pediatrics department. The choice of not including transplants before 2000 is the result of incompatible text data. Patients having a failing kidney within 12 months after the surgery were excluded since the definition we chose for labeling the data would label them incorrectly.

3.3 Data Labeling and Feature Selection

A classification task needs labels or classes. For our task we decided to set the label as short-term failure when the patient died or the kidney was lost between 12-18 months after the surgery. The first 12 months after transplantation were not considered as graft loss does not commonly occur during this time. Illustration 3.2 shows all time periods of the transplantation process.

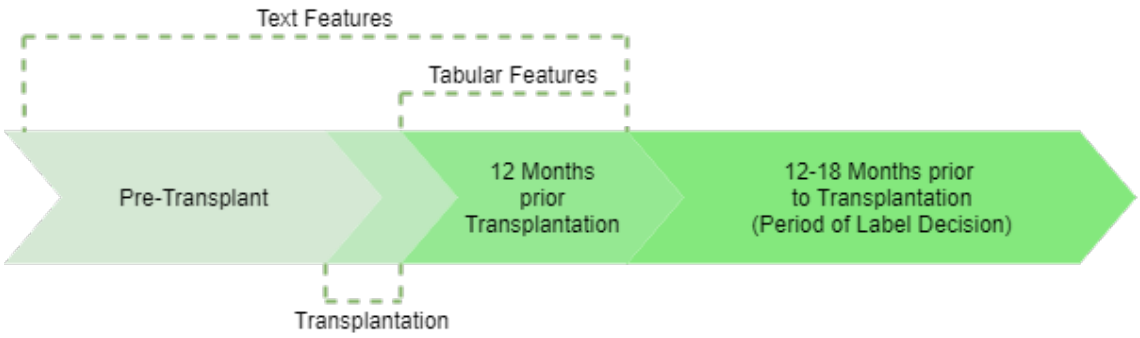


Figure 3.2: Label selection time frame. The illustration shows the time frames that are important to select the label for a patient. The decision of which label to allocate is made in the time period 12-18 months after the surgery.

Furthermore, a stratified train-test-split was conducted with a ratio of 70/30. Splitting the dataset resulted in a training set of 886 and a test set of 379 transplants whose class balance can be observed in figure 3.3.

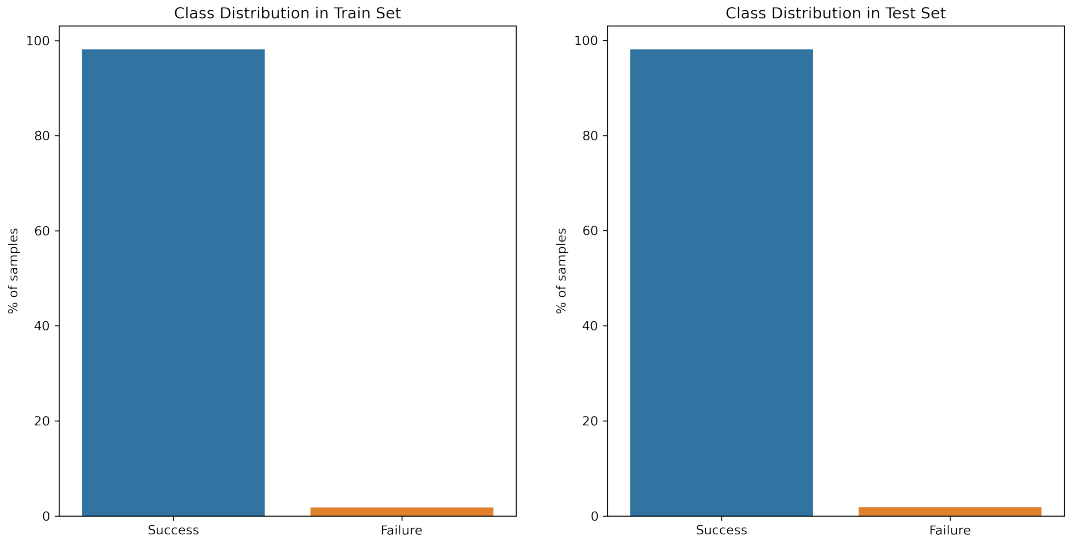


Figure 3.3: Class distribution plot. Shows the class distribution in percent after the train-test-split. It can be clearly seen that the dataset is highly biased towards success.

Feature Selection For the feature selection of the tabular data, we carried on with the selection period provided by the predecessor thesis. [26] In this work, tabular features were selected from the day of the surgery up until 12 months past it. The selection of text, however, starts at the first entry of a given patient until 12 months after the transplant. We considered data prior to the transplant to be valuable to the prediction, thus we included it.

3.4 A Temporary Task - Mimic-III Dataset Analysis

At the beginning of this thesis the access to the T-Base database had been delayed. As it was not clear when the access would be granted an alternative task was proposed. Instead of predicting AKI failure through the T-Base data collection, the prediction task of readmission to the ICU (intensive care unit) within 30, 15 and 7 days of discharge utilizing the Mimic-III dataset [58] was investigated. This task has already received significant dedication by the scientific community. [59–63]

The planned contribution was to implement text embeddings provided by a BERT model. Unfortunately, the temporary assigned task never led to a full implementation of a model, but a deep dive into the Mimic-III dataset had been conducted.

Overall the dataset holds 33,204 total patients and an absolute count of 44,026 ICU admissions. The average number of ICU visits per patient is 1.33, while the mean length of a stay is calculated to be 63.66 days. In total, 762,417 notes are contained in the dataset, per patient, 23 notes being the average per patient and 17.31 the average per visit. The Mimic-III data also includes 1947 unique ICU-9 codes for procedures and 210,376 in total, further it hosts 6,184 unique ICU-9 codes for diseases and a total of 521,786 codes can be found. Another important part of an EHR (Electronic Health Record) is the history of medication. Mimic collects this type of information as National

Drug Codes (NDC). There are 3725 unique NDCs found and in total 1,885,865 medications have been administered.

To be able to predict a readmission, a patient needs to be readmitted, meaning a patient needs to have at least two visits to the ICU to be a valid choice for the cohort. Another condition for an example to become part of the cohort is that the readmission has to take place in a time frame less than 30 days. The distribution of visits per patient, seen in figure 3.4, shows that most patients only experience one visit to the ICU during their illness. However the dataset still holds more than 1,000 patients with two visits and several patients with multiple visits.

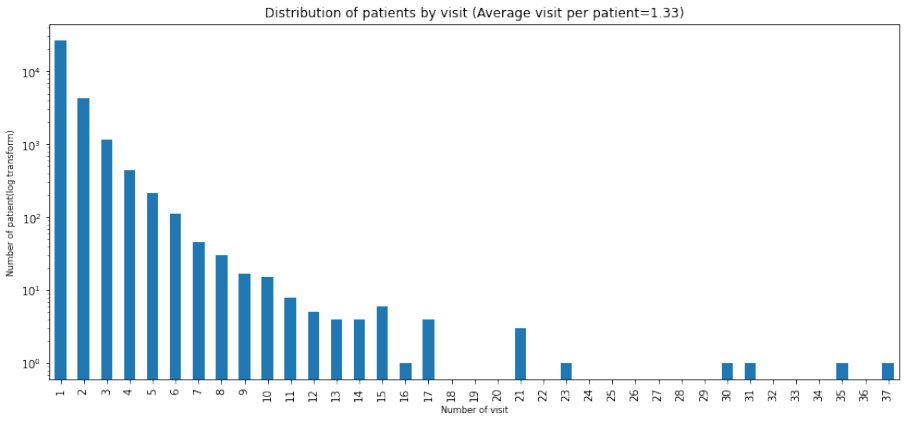


Figure 3.4: Number of visits per patient (Mimic-III).

Figure 3.5 shows the distribution of patients by readmission within 30, 15 and 7 days on the left. It can be seen that patients are much more likely to be readmitted within 7 days than during the longer time periods. Since the longer time periods also include the 7 days readmission, it might look like patients are more like to be readmitted within 30 days. However, every 5th patient has been readmitted within the time frame of 30 days which, in terms of class balance, definitely means that prediction of readmission is possible. The same pattern can be observed when readmission is displayed by visits, shown by the right graph of Figure 3.5.

The figure 3.6 shows on the left the average time spent during an admission (LOS) in hours by the count of visits and, on the right, the average time between visits by visit count. In the first diagram a volatility can be noticed when looking at the majority distribution of the stay length. A major outlier can be spotted with a count of two visits and an average stay length of over 4000 days. On the right diagram, a downtrend is visible from a stay count of 1 to 9, after which the length between the stays turns volatile again. A downtrend would have been expected, because the more serious an illness is, the more likely you are to be readmitted in general and in a short time frame.

To assess if the time spans, shown in figure 3.6, hold any impact towards the classes, a linear regression has been performed. For the length of a stay it can be concluded that it has a positive impact for all three classes (30, 15 and 7 days). The longer their stay, the more likely a patient is to be readmitted (see figure 3.7) which is expected because a shorter stay would indicate a less severe condition. The length of time between ICU stays is not significant, except in the 30 days class. There is no correlation between the outpatient time and a readmission.

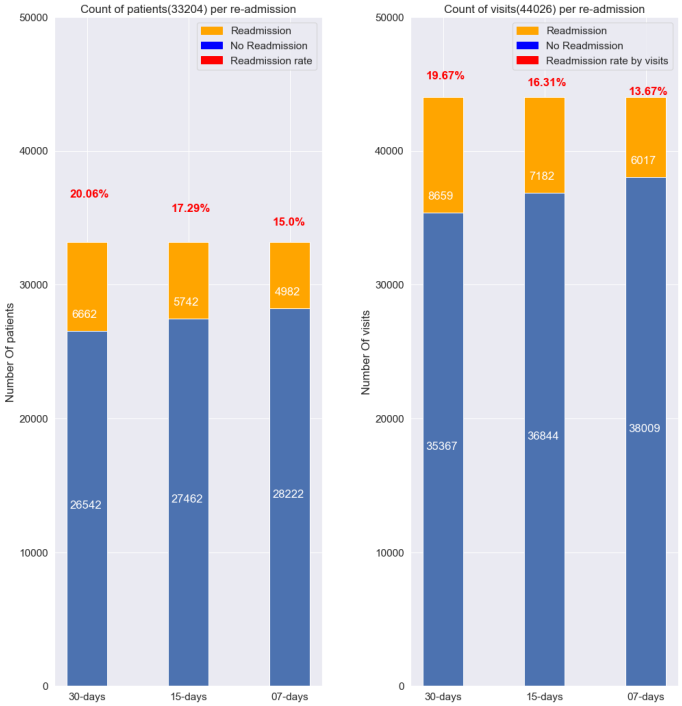


Figure 3.5: Count of patients or visit per readmission (Mimic-III).

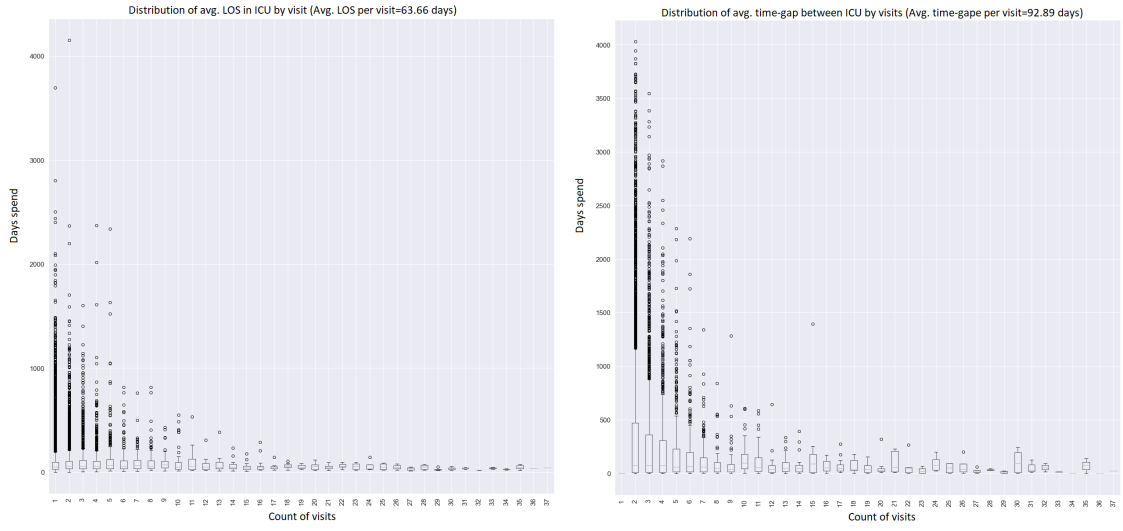


Figure 3.6: Average time spent in ICU and time between ICU admissions (Mimic-III).

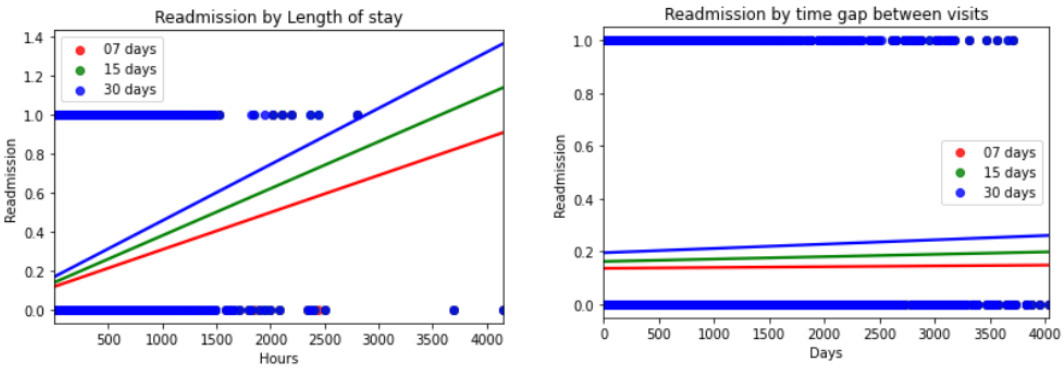


Figure 3.7: Linear-Regression Stay length and readmission (Mimic-III).

Furthermore, an investigation on the relationship between readmission and insurance type, as well as number of transfers, has been carried out. The investigation was conducted to find evidence for the differentiability of the classes. Figure 3.8 shows the relationship between transfers and readmissions. The x-axis represents the number of transfers a patient has gone through, while the y-axis displays the number of readmissions. Observing the graph shows that a higher transfer rate fosters a lower readmission rate. However, transfer numbers smaller than nine are abetting higher numbers of readmission.

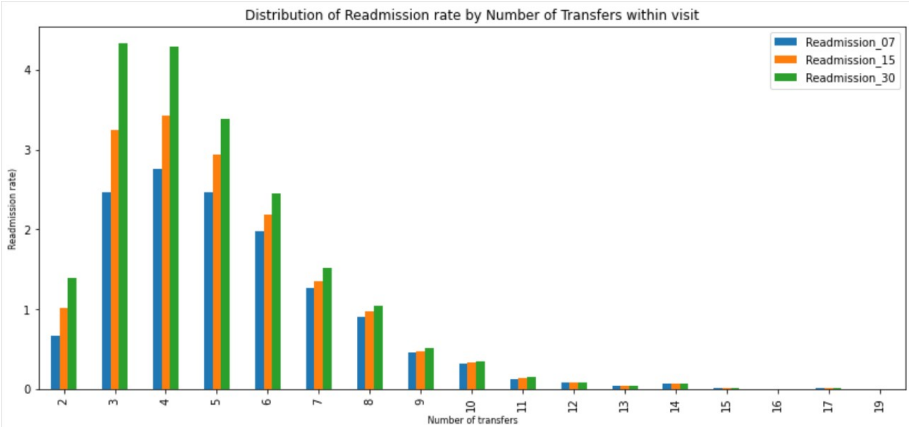


Figure 3.8: Distribution of Readmission rate by Number of Transfers (Mimic-III).

The correlation between the type of insurance and the number of readmissions can be seen in figure 3.9. The bar diagram displays the type of insurance on the horizontal axis, while the count of readmission is displayed on the vertical axis. The classes are differentiated by three different colored bars. Being insured with Medicare gives the patient an up to three times as high chance of being readmitted to the ICU within 30 days than being privately insured which gives a patient the second highest chance. This trend keeps up for every insurance type found in the Mimic-III. Governmental and no insurance (Self Pay) patients have the tendency to get readmitted fewer than ~1 times. The prediction between admission and no admission can be more efficient when implementing the insurance type as a feature since insurances like Medicare and Private hold a much higher chance for readmission than a governmental or no insurance.

Figure 3.10 shows the top 11 diseases found in the dataset in percent. It is apparent that the

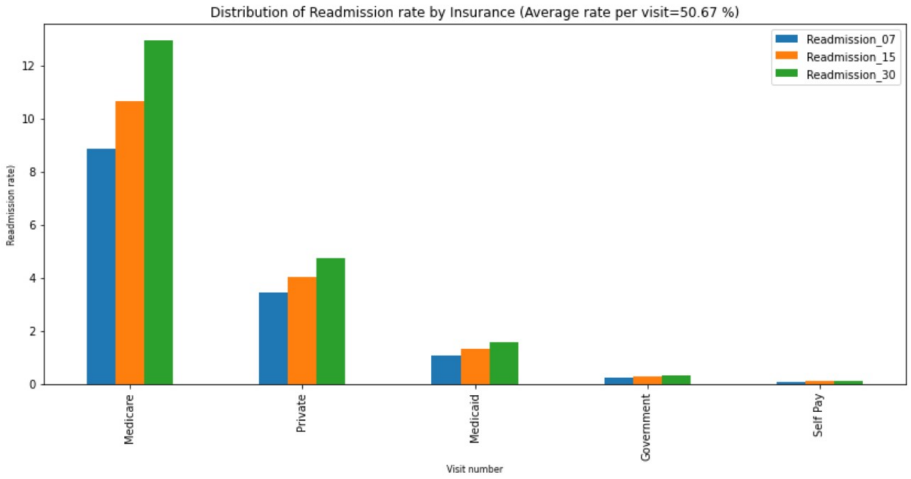


Figure 3.9: Distribution of Readmission rate by insurance type (Mimic-III).

most common diseases distribute equally between the three classes, excluding Disorders of lipoid metabolism and Systemic inflammatory response syndrome, which are only found in one or two of the three classes. Since the classes are equally distributed, it might be difficult to distinguish between them through this feature.

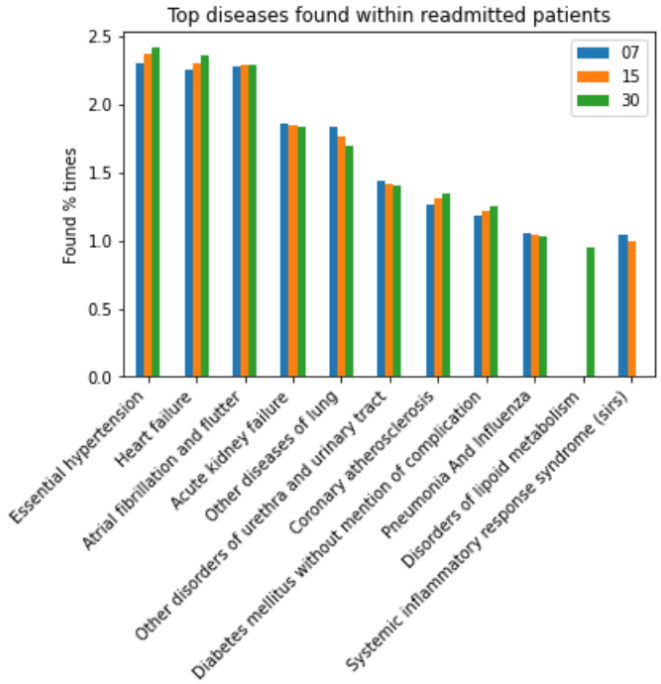


Figure 3.10: Distribution of top diseases found by readmission period (Mimic-III).

To see which impact the disease, procedure and medical codes have, a linear regression was conducted between them and the readmission count which can be seen in figure 3.11. The p-value for the correlation between a readmission and the number of diagnoses lies at 0.003, which makes the correlation, with a confidence interval of 5%, statistically significant. A negative impact can be noticed after the 25th diagnosis, however a count of up to 10 diagnoses benefits further readmissions. A proportional rise of the number of admissions and the number of diagnoses should be expected

since more diseases could also increase the risk of a severe illness. However, this is only the case until a count of 10 diagnoses. After the 11th diagnosis the number of diagnoses falls to ~250, does a dead cat bounce and descends to ~1. For the middle graph displayed in figure 3.11, the p-value is estimated to be 6.02e-12. The graph shows the correlation between the number of performed procedures and the rate of readmissions. A declining trend of admissions by rising procedures can be observed. An explanation for this trend might be that patients who receive more treatments than necessary are actually contributing to their health. The last graph situated on the right shows the relation between the rate of medicine administered and the readmission count. Calculating the p-values resulted in a statistically significant value of 1.68e. A negative impact can be noticed upon taking around 50 medications which leads to a steep decline in readmissions.

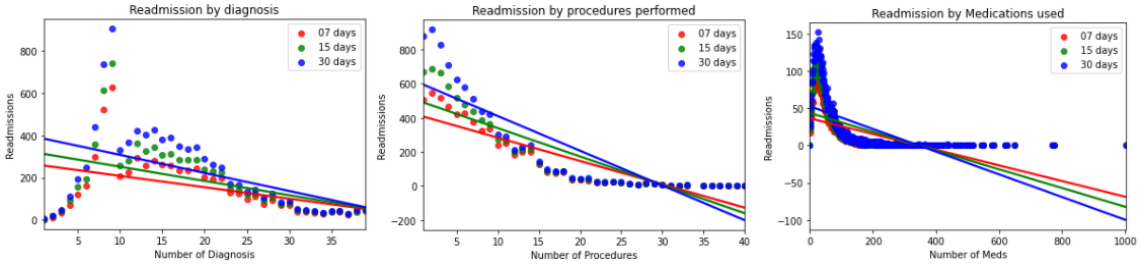


Figure 3.11: Linear regression of disease, procedure and medical codes (Mimic-III).

In conclusion, the investigation shows that the dataset can indeed be used to predict readmission. This result is also backed by the fact that positive results have already been achieved by the scientific community. [59–63] The dataset provides sufficient amounts of data with a total of 44,026 readmissions. Even though the class balance is 1 to 4 for the largest class, a plausible discriminability thought features is given

3.5 Summary

This chapter dealt with the data used in this project. It first covered the structure of the database T-Base and exposed all features used for each of the two data types (figure 3.1). Then the bias towards failure of the dataset was shown and explained followed by the feature selection and the labeling. The chapter closed out with a section regarding the Mimic-III dataset. A small analysis of the dataset was conducted.

4 Methodology

In this chapter we cover all the approaches done to prove our hypothesis. To begin with, we will first cover out hypothesis and then state the problem definition. Next we will give an introduction to our chosen baseline and the tabular data preparation measures. This is followed up by the tabular model architecture, the text data preparation and the implementation of the text model. Lastly, the measures taken to create the ensemble model will be laid out.

4.1 Random Forest Baseline

Most machine learning tasks have a predecessor that already did work on that particular task and delivered results. To compare our work and see if we were able to improve upon the past product, we use the best model by Haldar et al. [26] In the prior work, a variety of models have been evaluated which are Random Forest (RF), Multilayer Perceptron (MLP) and LSTM mono- and bidirectional. The architecture that beat all others was determined to be the RF.

4.1.1 Data Selection

The baseline only considers categorical and numerical data as an input for the prediction. However, the full feature vector was 229 features long. Haldar chose all features in correspondence with medical staff and refined the selection after conducting error analysis. The following tables show the final selection.

Table 4.1 shows a collection of patient detail and their selection for the receiver and donor. Some of the features might be self-explanatory, while others are not. So only features that might not explain themselves to the reader are described in the following section. The Primary Function states if the kidney transplanted was already working during the surgery. The side at which the kidney was taken from or transplanted to is described by "Side of Kidney", this is important since the radius, and thus the perfusion, of the arteries differ from left to right kidney. Medication might not be unknown, but it is worth noting that it is not a single feature but can be itemized into all medicines available. However, listing all of them would be excessive. The abbreviations found in the middle and last third of the table are antibody occurrences. Lastly, the donor type refers to the state of aliveness of the donor when receiving the kidney.

The features got selected from the Patient and Donor tables, as well as the Transplantation Table.

Details	Receiver	Donor
Sex	✓	✓
Body Height	✓	✓
Age	✓	✓
Body Weight	✓	✓
Blood type	✓	✓
Primary Function	✓	X
Side of Kidney	✓	✓
Medication	✓	X
Blood pressure	✓	X
Hearth rate	✓	X
CMV AK pos	✓	X
CMV IGG pos	X	✓
HBC AK pos	X	✓
HBC AB pos	X	✓
HCV AB pos	X	✓
HCV AK pos	✓	✓
HBS AG pos	✓	✓
Dialysis type	✓	X
Donor type	X	✓
Degree of Kinship	X	✓
Time Δ first Dialysis and Transp.	✓	X

Table 4.1: Selected features for receiver and donor; The table gives an overview of the features selected for receiver and donor. A ✓states that the feature has been selected for that participant, while an X determines that this feature is absent.

Further a features for diagnosis and primary disease were selected, both as categorical values consisting of 30 and 26 categories.

Lab values	Kidney
Creatinin	Ischämie kalt
CPR	BANFF
Leukozyten - Blood	PRA
Leukozyten - Urine	MM Board
Protein Concentration	
Protein Dipstick	
ProteinTUR	
Daily Protein output	
NitritTUR	

Table 4.2: Selected features for Lab values and the kidney; The table gives an overview of the features selected for Lab values and the kidney.

The Lab values seen in table 4.2 are mostly different methods of measuring protein. All the values were collected from the day of transplantation until one year after. The values were originally a time series, however, for the RF model, mean values with different time spans have been calculated like overall, after 1, 2 or 3 months, to create a uniform input vector length. All values in this column are numerical.

Values concerning the kidney can be seen in the second column. The BANFF is a scalar that shows the quality of the kidney, thus is a categorical value. [64] MM Board states how well the receiver and donor match in terms of kidney transplantation.

4.1.2 Data Preparation

To transform the data into a format useful for ML, several measures have been taken by Haldar et al. Categorical features are usually given in a string format, however ML models can only process numbers. To convert them to coherent numbers that still represent the original categories, several techniques are available. [65] Haldar chose one-hot-encoding for transforming the values. Furthermore, a normalization was implemented to set values with high ranges, like age, between zero and one. This keeps certain model architectures (Deep learning, logistic regression, etc.) from weighting values with high numbers more strongly. [26] Following up the normalization is an imputation, which fills in missing values based on values that are present in the example of a patient. [66] The last step in the data preparation comes with the oversampling technique SMOTE to balance out the dataset. The balance was restored to 50/50. Figure 4.1 shows the full workflow of the data preparation.

4.2 Tabular Model

Despite the fact that Haldar already implemented a MLP model for the tabular data, we were under the perception that the Principal Component Analysis (PCA) [67] he applied before passing the

Figure 4.1: Tabular data preparation workflow. The figure shows the sequence in which the data transformations are applied to the data. It is important to note the order in which the transformations are performed.

vector to the MLP model would impair the results of the model.We came to this conclusion since a decremental architecture would give the model an opportunity to condense that vector itself and not lose data before inputting it. The architecture of the final model can be seen in figure 4.2. Furthermore, we decided to apply additional balancing techniques to the data. The data however runs through the same process described in section 4.1.2 We did not apply any changes to it other than leaving out the PCA, which was anyways only applied to the MLP and not the RF model by Haldar [26]. In total, 12 experiments were conducted.

4.2.1 Tabular Model Experimental Setup

As the dataset used was highly imbalanced, a suitable technique to handle this imbalance was needed. To find this technique we ran a multitude of experiments. We included mostly balancing methods that do not use oversampling as their core idea since oversampling was already used in the baseline. Furthermore, we differentiated between a homogeneous and decremental layer architecture. The decremental architecture was evaluated to make the previously used PCA redundant.

Choosing Hyperparameters Since time constraints were a limiting factor in this work, we decided to run all experiments with the same hyperparameters. We based our choice on prior experience and the review of papers. [68] Our input layer was determined to be 229, as this was the feature length resulting from the previous work. While our output layer consists of one neuron, the hidden layers were interchangeably a linear layer followed by a batch norm layer. Final selection of hyperparameters can be seen in the following list.

- Epochs: 100
- Batch Size: 64
- Learning Rate: 1e-4
- Hidden Layer Size: 32
- Layers: 3
- Decemental: True & False

Balancing Techniques. Included in the experiments were six methods. As a minimal attempt we used the model architecture without modifying the data or loss function. Furthermore, we investigated balancing methods that respected the ratio of the classes like class weighting [69], weighted random sampler [49] and focal loss [70], as well as a combination of class weighting and weighted random sampler. Lastly, the baseline technique of SMOTE was examined [50].

4.2.2 Tabular MLP Model

Evaluating the previously conducted experiments, we concluded that weighted random sampler and a decremental model architecture would deliver the best results, as well as a model which uses no balancing technique.

Model Architecture. The full architecture can be seen in figure 4.2. As a result of the previously conducted work by Haldar [26] the input layer was determined to be 229. Since the size of the feature vector was high in comparison with the number of examples in our dataset, we decided to use a decremental architecture to avoid the curse of dimensionality. Furthermore, the smaller the last layer of the hidden unit is, more like an evasion of the curse of dimensionality on the ensemble model. The hidden layers start with a size of 256 and end with a size of, followed by an output layer of 1 for the binary prediction. Moreover, we chose alternating layers of linear and batch norm. Batch norm layers were used to quicken the training and to smooth the learning. [71] We used binary cross entropy loss as our loss function.

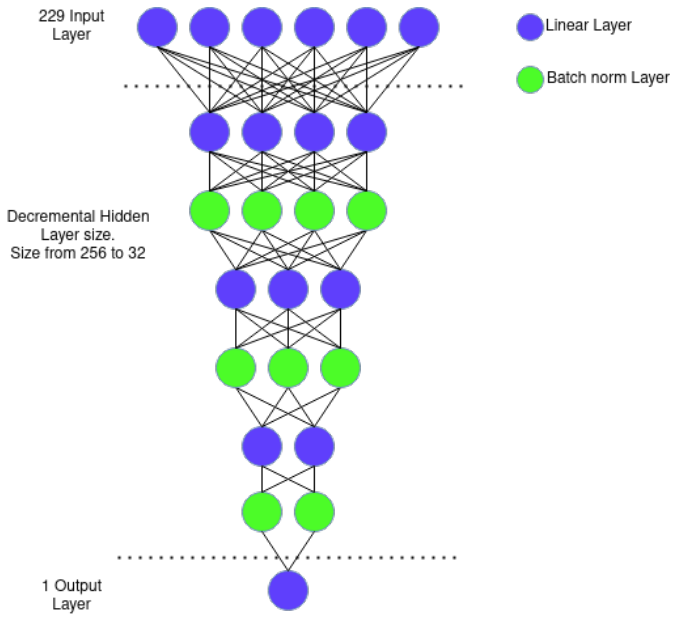


Figure 4.2: Tabular model architecture. Linear layers are displayed as blue, while green neurons are batch norm layers. Important to note is the decremental structure of the model.

Hyperparameter selection. To find fitting hyperparameters we conducted a hyperparameter search. The ranges and parameters used for the grid search are displayed in table 4.3. We used a wide range of values which made the grid search take four days. The last column of the table shows the selected hyperparameters, while the second, third and fourth columns show the minimum, maximum and the steps the optimizer can take.

4.3 Text Model

This section will cover the setup of all experiments related to the text model investigation. Several experiments were conducted to find the best-suited BERT model for the text corpus. This includes

Hyperparameter	Grid Search Minimum	Grid Search Maximum	Grid Search Steps	Final Hyperparameters
Epochs	50	1000	10	650
Batch Size	8	192	8	96
Learning Rate	1e-8	1e-4	1e-1	1e-5
Layers	2	5	1	5
Hidden Dimension	16	512	8	256
Decremental	-	-	-	True
Class Weights	-	-	-	0.0102, 0.5525

Table 4.3: Hyperparameter Optimization and resulting hyperparameters. The table shows from left to right first the selected ranges and steps for the grid search and in the last two columns the hyperparameter resulting from the grid search for the tabular model.

multiple BERT models, text sampling techniques and data balancing treatments. Upon finding the best fitting combination an MLP model has been trained to predict short term AKIs. Additionally, the steps taken to train and evaluate the models are mentioned. This section will also cover the measures to prepare the text are presented.

4.3.1 Text Preparation Measures

Clinical notes are usually written by a doctor and feature his/her impressions and observations of the patient's health condition. Most of the time, text for a single patient even comes from a multitude of medical personnel. Knowing this, the expected text will be unstructured and with high variance. This leads to a high amount of noise in the data. Even if a doctor decides to structure their text well, other doctors might not follow their patterns.They might have their own ways of structuring the notes or just write continuous text, meaning that a foundation for uniformly well-structured medical text is not given. Furthermore, the medical domain features a vast amount of medical abbreviations, some of which are ambiguous and fall out of context for a NLP model if not properly trained. Text featured in an EHR might also be structured when created by an algorithm or a UI used by the user. Those reports usually originate from laboratory results. Generally, text data mostly contains unwanted or nonsensical symbols, like HTML tags, that need to be addressed. Thus, prepossessing the text is necessary.

This work features two iterations of text preprocessing variances. A general flow of the processing pipeline can be seen in figure 4.3. The first iteration has a minimalist approach to preprocess the text and is used for most of the experiments and results in this thesis. Iteration two, on the other hand, has been created after the first error analysis and is therefore much more refined. If iteration two is used for one of the tasks, it will be explicitly stated.

We were provided two sources of text from the Untersuchung table and the Verlauf table. As for the Verlauf table, we concatenated the texts of the columns Beurteilung, BeurteilungAerztlich and BeurteilungIntern. For the most part, the three parts follow repeating patterns and are comparably short. Untersuchung, on the other hand, features long and unstructured text but is clustered into multiple categories..

Category driven measures. The provided Untersuchung text was clustered in 105 categories. Having the text labeled by category was helpful to determine characteristics and structures of cer-

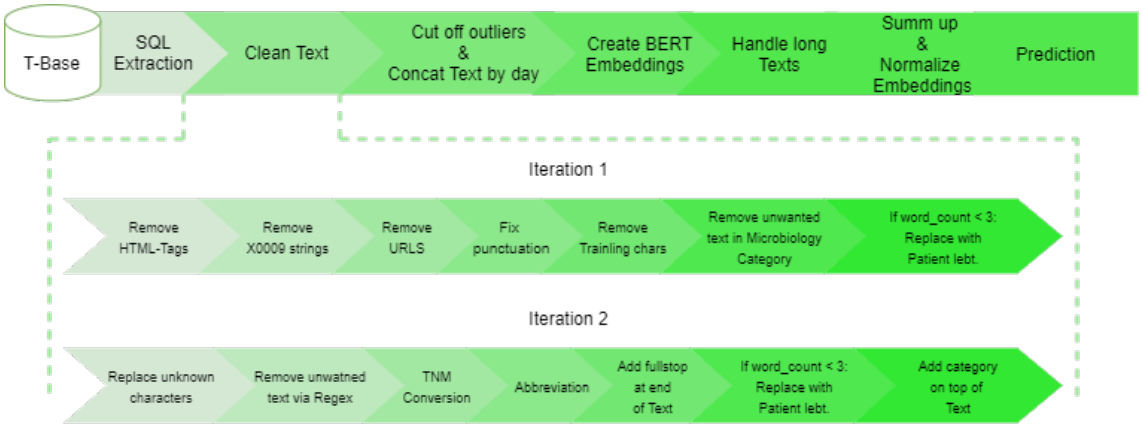


Figure 4.3: Generalized Text Pipeline. The figure shows a generalization of data processing for text. At the lower third of the figure the text cleaning steps can be seen for iteration one and two. The text cleaning steps are the same in number, but the first three steps of iteration one are contained in the first step of iteration two.

tain texts.We used this convenience to apply measures selectively to certain categories. Texts of the category Mikrobiologie were found to always store their relevant information in between the words "Material:" and "medizinisch validiert", so we were able to extract this part via a regular expression seen in table 4.4. For the second iteration we applied additional category-driven measures, like removing reappearing street names and the JNr for pathology reports. In the Verlauf category we threw out empty parts that would usually state which doctor the patient had been treated by. The text of the KOD category had TNM classification codes, which are codes to determine the state of a tumor. [72] We translated those codes to readable text describing the state of the tumor. Furthermore, we used the category as a headline for each text to give the text model more context about it and reduced the variance of categories by aggregation. The categories were aggregated as some of them were duplicates like "Sono" and "Sonographie". For both iterations we used quantile values by category to trim outliers. This measure was taken since some of the texts appeared to be repeating themselves multiple times in one entire.

On Handling Medical Abbreviations. Text of both the tables Untersuchung and Verlauf are studded with abbreviations that refer to medical vocabulary. Using abbreviations with a text model that has not been fine-tuned to understand them might lead to a loss of information, especially when the abbreviations are ambiguous. [73] To handle these abbreviations, we decided to create a dictionary containing the abbreviation and its corresponding full phrases. Wikipedia offers a German article for medical abbreviations ordered in a table structure. We captured this data via web request.[1] This gave us 2822 translations of which about 300 were ambiguous. For the ambiguous abbreviations, we let a medical doctor who has insight into the data decide which of the translations to keep. This condensed the vocabulary to 2514. Some abbreviations could be written out using either a colloquial or a professional term. Whenever possible, we decided to go with the colloquial term, since our BERT model is neither pretraind nor fine-tuned on medical text. Examples include the abbreviation "RR", which translates to "Riva-Rocci" and means blood pressure. To locate and

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

replace the abbreviations, we used a regular expression. Conversion of the abbreviations was only done on the second iteration of text.

Replacing Unknown Characters. Whenever BERT encounters a character it cannot encode, it is going to replace the whole word in which the character is found with the [UNK] token. Losing whole phrases because of a single character not found means losing information, while not being able to encode Greek or Scandinavian letters like μ or $\varnothing$, which themselves hold information, also results in information loss. To prevent this loss of information, we decided to apply different methods for replacing or removing unknown characters depending on the case.

We found that some combinations of characters that were unknown to BERT would always be equivalent to the same letters. For example, if an "Ä" followed by a "^{2}" was observed, the "^{2}" would need to be replaced with an "r" for the word to be readable. We were able to track down 10 of those instances. Furthermore, we replaced accented letters, like é, with their unaccented counterparts and spelled in full symbols like $\varnothing$. Some characters that were absolutely not classifiable were completely removed.

Using Regular Expression to remove unwanted Text Chunks. The text found in the database came with repetitive and impractical information for the prediction target. To remove the unwanted information, we implemented regular expressions. Table 4.4 shows the regular expressions used and provides information about which iterations they are used in. Common information found in text we deemed as not important to the prediction are HTML tags, URLs, telephone numbers and street names. Further clinical specific findings we removed in agreement with medical staff were the admission number, the date at which a "Befund" was received, the laboratory number, the case number and the J-Nr. We also removed zip codes followed by a city name with a list of 9178 city names. Lastly, we found some instances of text where a "\" followed by an "X" and a combination of numbers were present. We removed those as well.

Description	Regex	Iteration 1	Iteration 2
URL	(http|ftp|https://([\w_-]+(?:(?:\.[\w_-]+)+))([\w.,@?^=%&:/~+-]*[@?^=%&/~+-])	✓	X
URL V2	(?:(?:ftp|http|https)?(?::///)?(?:[Ww]{3,3}\.|)[A-Za-z0-9-+]+\.[A-Za-z]{2,5}(?:/[A-Za-z0-9?+-_=&.~#]|/|))	X	✓
X0009	\\X(.*?)\\	X	✓
Street names	((?:|Am)([a-zA-Zäüö-]{1,}(?: |)(?:**street_name_suffixes**)(|)([1-90]{1,3}(?:|)[a-zA-Z]|[1-90]{1,3}),*))	X	✓
Zip code and City	(\b[1-90]{5,5} (?:**name_of_city**)\b)	X	✓
HTML-Tags	<.*?>	✓	✓
Telefon number	(\(?:\b(?:**tele_prefixes**)(?:| |:)(?:[0-9]{0,3}(?:-| |/|))(?:(?:-| |/|)[0-9]+)+\)?)	X	✓
All Caps and Split Words	\b(?:[A-Z]{1,1}(?:|:|\.))){2,}	X	X
General Split Words	\b(?:[A-Za-z]{1,1}(?:\s|:|\.))){2,}	X	X
Abbreviations	(?:(?:\b(|/|\s) + **Abbreviation** + [.,;:!?|\s])	X	✓
Admission number	((?:Alte Aufnahmenummer:|Aufnahmenummer:)\s[1-90]{3,13})	X	✓
the word bold	(bold(?: |))	X	✓
Befundsempfänger	(Befundempfänger: Berlin, den [0-9]{0,2}\.[0-9]{0,2}\.[0-9]{0,4})	X	✓
Laboratory Nr.	(Labor./Auftrags-Nr.: [0-9]*)	X	✓
Case number	(Fall-Nr.: [0-9]*)	X	✓
Empty seen through	(?:gesehen durch:(?: |)Therapie)	X	✓
Empty therapy through	(?:Therapie durch\:$)	X	✓
J-Nr.	(J-Nr\.:)((\s[A-Z]{0,1}[0-9]*(-|)[0-9]{0,2})|([A-Z]{0,1}[0-9]*(-|)[0-9]{0,2}))	X	✓
TNM-Codes	(?:(?:(?:p|c|y){0,3}T(?:[0-4]|is)|(?:(?:p|c|y){0,3}N(?:[0-4X])))1,2(?:[A-Za-z0-9(+)\/]|)(?:\b|))	X	✓
Microbiology cutout	(?:\sMaterial:.medizinisch validiert)	✓	✓

Variable names can be found in the appendix (street_name_suffixes, tele_prefixes, name_of_city). They have been neglected to keep the table lucid.

Table 4.4: Regular Expression; The table shows all considered regular expressions in the middle column and their use in the first column. Important to note are the last two columns that state which iteration of text cleaning what regular expressions are used in.

Empty and Short Texts. Entries in the text were sometimes empty or featured fewer than 5 words. As those entries would not deliver any or very little information, we, in correspondence with the

medical staff, decided to replace those with the phrase "Patient lebt.". The decision was made on the premise that entries that short do not feature any problematic occurrences in the patient and thus were only made to state the visit and well-being of the patient. This effort was made for both iteration one and two.

Text Aggregating and Selection. At last we decided to aggregate the text by day. Texts are already organized by day and transplantation ID, however, if two or more texts appear on the same date for one transplantation ID, we concatenated them. Furthermore, to aggregate one corpus for a prediction, we selected all text prior to the transplant date and all texts up until one year after the transplant.

4.3.2 BERT Models - Overview

This section gives an overview of the selected BERT model for the evaluation. Three BERT models were open for selection. The models are partially pretrained on different datasets and partially fine-tuned on different tasks. We included a selection to investigate which combination of pretraining and fine-tuning would provide the best results on our text corpus.

Base-BERT-German

The Base-BERT-German-Cased model [5][2] has been published in 2019.It was trained on three different text corpora, including 6GB of German Wikipedia, the OpenLegalData[3] dump contributing 2.4 GB and news articles with 3.6 GB. The training was realised within 810.000 steps, a batch size of 1024 and a sequence length of 128, as well as 30.000 steps with sequence length 512. The model was evaluated on five German NLP tasks using the F1-Score as a metric and comparing it to multilingual models as shown in table 4.5.

Model	GermEval18 (coarse)	GermEval18 (Fine)	GermEval14	CONLL03	10kGNAD
multilingual cased	0.710	0.441	0.0834	0.792	0.888
multilingual uncased	0.731	0.461	0.823	0.786	0.901
German Bert cased	**0.747**	**0.488**	**0.840**	**0.804**	**0.905**

Table 4.5: Base-BERT-German Evaluation of Deepset. [5] The table shows a comparison of Deepsets German Base-BERT-cased with multilingual models and five tasks. Important to note is that the BERT model outperforms. The score compared is the F1-Score.

The first two tasks mentioned in table 4.5 are classification tasks (multi- and binary classification), the second and third one are NER tasks and the last one is a document classification task. As the table shows, it was possible for the Base-Bert-German model to outperform the multilingual models.

Since the medical notes found in TBase are written in the German language, Base-Bert-German has been considered a suitable choice for the kidney prediction task.

[2] https://deepset.ai/german-bert

[3] http://openlegaldata.io/research/2019/02/19/court-decision-dataset.html

gBERT-Base

Released by Deepset in the last quarter of 2020, the gBERT model was trained on significantly more data then the Base-Bert-German model. [6] A German version of the OSCAR dataset [74] was used, which was obtained by a comman crawl, sorted by language and preprocessed, leading to 145GB of German text. Further, a Wiki dump was collected using the WikiExtractor repository[4] resulting in 6GB. Moreover, the ORUS dataset[5] consisting of subtitle text was added to the corpus and contributed 10GB, [75] as well as the already mentioned Open legal data dump with 2.4 GB. The overall size of the dataset adds up to 163.4 GB of German text.

The training was conducted with a sequence size of 512, a batch size of 128 and a maximum of 4000 training steps.

	GermEval18 (Coarse)	GermEval18 (Fine)	GermEval14	Averaged F1
GBERT$_{Data}$	74.51	48.01	87.41	69.97
GBERT$_{WWM}$	76.48	49.99	87.80	71.42
GBERT$_{Data+WWM}$	**78.17**	**50.90**	**87.98**	**72.35**

$_{Data}$: Full dataset without Whole Word Masking
$_{WWM}$: OPUS and Wiki dump data without Whole Word Masking
$_{Data+WWM}$: Full dataset with Whole Word Masking

Table 4.6: gBert Evaluation of Deepset. [6] The table shows Deepset next generation language model in comparison on different training data and tasks. It is worth noting that it outperforms the BERT model of table 4.5 in all overlapping tasks. The evaluation metric is the F1-Score

The scores seen in table 4.6 are macro average F1-Scores. Comparing table 4.6 and table 4.5 on the overlapping tasks, gBERT outperforms Base-BERT-German by 2-4% points on all tasks. So, we also deemed this model to be a fit for the given task.

German MedBert

During research, a German MedBERT model had been found. [7][6] The model is based on the Base-BERT-German model and has been fine-tuned on medical text with the task of document multilabel classification for ICD-10 codes.

Models	Precision	Recall	F1
German BERT	86.04	75.82	80.60
German MedBERT-256	87.41	77.97	82.42
German MedBERT-512	87.75	78.26	82.73

Table 4.7: MedBert Evaluation. [7] While not fully comparable with the Base-BERT and gBERT model, due the different datasets and tasks, the model still shows decent results.

Table 4.7 shows that the model performed decently on its task and was able to outperform the German Base-BERT model. Even though this model is based on German Base-BERT, we decided to include it in the experiments since it was fine-tuned on predicting diseases.

[4] https://github.com/attardi/wikiextractor

[5] http://opus.nlpl.eu

[6] https://huggingface.co/smanjil/German-MedBERT

4.3.3 BERT Model Experimental Setup

Given the medical text corpus and the imbalance of classes associated with the dataset, we conducted a series of experiments. The experiments are based on three parameters: the chosen BERT model, the technique to handle the imbalance of the data and the way BERT tokens longer than 512 were managed. In total, 13 balancing techniques, three BERT models and three token handling algorithms were considered. The Models and techniques were all fully compared, however, we decided to only test the different token aggregate techniques on the best combination of the previously best-performing factors. In total, 65 experiments have been run to find a suiting model.

Choosing Hyperparameters. Due to time constraints we ran all experiments with equal hyperparameters. We choose the hyperparameters by prior experience and paper reviews. [68] Our MLP architecture consisted of one hidden linear and one batch normalisation layer. The input size was determined to be 768 nodes.

- Epochs: 150
- Batch Size: 64
- Learning Rate: 1e-4
- Hidden Layer Size: 64

Balancing Techniques. The method used for imbalance handling can be sorted into two categories - Weighting- and Oversampling techniques. The first category covers the loss function Focal Loss [70], as well as the class balancing rudiments class weighting [69] and weighted random sampler [49]. Covering the oversampling techniques, we used SMOTE [50] and its varieties, as well as Adasyn [57]. We also did crossovers of the two categories mainly using Focal loss.

Managing long Texts with BERT. For long texts with BERT, the simplest method we contemplated was trimming off all tokens after a count of 512. This is the most common way found to deal with long texts. [76, 77] Furthermore, we investigated a process where text corpora bigger than 512 tokens get split into multiple chunks. Each chunk has the [CLS] token embedding added as its first token and, after processing the embeddings, gets summed up. At last, a normalization is calculated, as proposed by [78]. The last method investigated is based on the second method introduced, the only difference being that the number of chunks per text is tracked and the summed up embeddings are divided by it to calculate an average before normalizing. [79] Table 4.8 shows all parameters in an overview.

Embeddings were extracted prior to the individual training of each BERT model. Training was then conducted separately with the extracted embeddings on the model mentioned above. All combinations have been evaluated on a multitude of metrics.

We are aware that the setup lacks finesse in terms of the hyperparameters, however, we were under the perception that all techniques would perform equally badly when using the same hyperparameters, thus making them comparable.

Models	Token Handling	Balancing Methods
German Base-BERT	First 512 Tokens	Plain Data & Model
gBERT-Base	Summed up embedidngs	Focal Loss
German MedBERT	Summed up and averaged embeddings	Class Weighting
		Weighted Random Sampler
		Class Weighting + Weighted Random Sampler
		SMOTE BL
		SVM-SMOTE
		SVM-SMOTE + Focal Loss
		Adasyn
		Adasyn + Focal Loss
		SMOTE ENN
		SMOTETomek

Table 4.8: Bert Experiment Parameters; All parameters for the conducted BERT experiments are listed in this table to give an overview.

4.3.4 Text MLP Model

To construct the text-only prediction model, the experiments conducted on the BERT model were evaluated. We chose the best-performing combination of the parameters seen in table 4.8. The best-performing combination was evaluated to be gBert, weighted random sampler and class weighting and summed up embeddings. For this combination we first used the iteration one text embeddings and compared the results after the error analysis with the embeddings of iteration two.

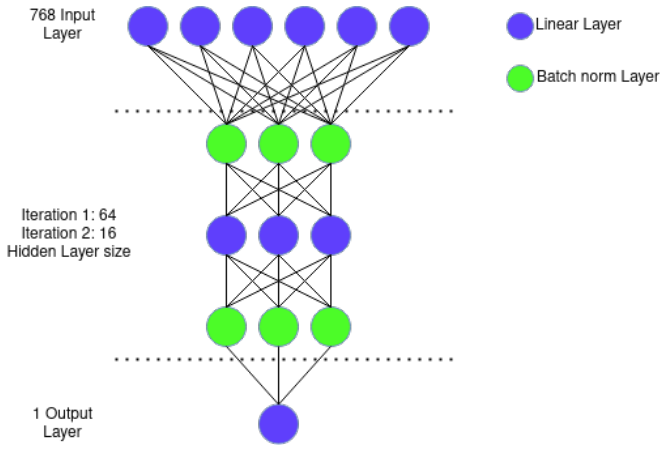


Figure 4.4: Text Model Architecture of iteration one and two. Blue nodes resemble linear layers, while green ones are batch normalization layers. In- and output layers for both iterations are the same, as well as the count of hidden layers. The only difference in the architectures can be seen in the size of the hidden layers.

Model Architecture. As seen in figure 4.4, the architecture consists of a 768 linear input layer, the size having been determined by the embedding size delivered from BERT. Furthermore, we used alternating linear and batch norm layers with different layer sizes for iteration one and two. The output layer was chosen to be one, since the model needed to make a binary decision as loss function binary cross entropy loss with class weights was used.

Hyperparameter selection. To get suitable hyperparameters, we conducted a hyperparameter search for both of the models. The selected hyperparameters can be seen in table 4.12. The column

grid search minimum states the lowest possible values that can result from the search, while grid search maximum states the highest. Grid search steps displays the amount the algorithm can add withe conducting the grid search. Iteration one and two hold the values that resulted from the grid search. Parameters excluded from the hyperparameter optimization were where the possibility of a decremental hidden layer architecture and the class weight. A decremental architecture, however, was tested with the given parameter, but performed worse. The class weight was calculated with the ratio of the classes found in the cohort. We deliberately chose a maximum of 64 hidden dimension size, since the embeddings of the last hidden layer of this model will be used as an input to the ensemble model (chapter 4.4) and we thus wanted to prevent the curse of dimensionality.

Hyperparameter	Grid Search Minimum	Grid Search Maximum	Grid Search Steps	Iteration 1	Iteration 2
Epochs	50	1000	20	185	145
Batch Size	8	96	8	32	80
Learning Rate	1e-6	1e-4	1e-1	1e-5	1e-4
Regularization Lambda	0.01	0.07	-	0.03	0.022
Layers	2	4	1	3	3
Hidden Dimension	16	64	8	64	16
Decremental	-	-	-	False	False
Class Weights	-	-	-	0.0102, 0.5525	0.0102, 0.5525

Table 4.9: Hyperparameter Optimization and resulting hyperparameters. The table shows, from left to right, first the selected ranges and steps for the grid search and, in the last two columns, the hyperparameter resulting from the grid search for iteration one and two.

4.3.5 BERT Pretraining

As gBERT is a general purpose German NLP model, its capabilities are not optimized for producing embeddings that are suitable for predicting disease. There are two possible approaches to improve the ability of a BERT model for a specific language domain like medicine. The first is pre-training, where the complete model gets trained from the ground up, including learning a vocabulary. This approach, however, is dependent on large quantities of text from that specific domain, strong processing units and time. BERT Models that underwent this procedure do exist but unfortunately only for the English language. [80] The another pretraining approach is, where the model learns a defined task and, in doing so, learns how to adapt its embeddings in a way that contributes to this task. This approach is not as time intensive and can be done on a smaller text corpus. The German Med-BERT we proposed in the text experiments section (section 4.5.2) is such a model, though it has not been pretrained for the classification of kidney graft loss but for general disease prediction.

Pretraining Task Definition To improve our prediction on the text corpus, we decided to also pretrain BERT on a task comparable to our downstream task. We had to diverge from the labels of the main task, since those were to sparse for daily labeling. Instead we pretrained the BERT model on predicting risk-factors of graft loss. Our labels were produced using the RIFLE [8] scheme. Since the RIFLE scheme defines stages that lead to a graft loss, it is possible for the BERT model to associate fine grained risk-factors with the texts. Table 4.10 shows the definition of the RIFLE criteria. We neglected the urine output, as advised by the medical professionals.

A requirement to calculate the RIFLE score is a baseline creatinine value need for each patient, to

Label	GFR (glomerular filtration rate) Criteria	Urine Output Criteria
RISK	Increased Baseline Creatinine x 1.5 or GFR decreased > 25%	Urine output < 0.5 ml/kg/h x 6 hours
INJURY	Increased Baseline Creatinine x 2.0 or GFR decreased > 50%	Urine output < 0.5 ml/kg/h x 12 hours
FAILURE	Increased Baseline Creatinine x 3.0 or GFR decreased > 75% Or Baseline Creatinine>4 mg/dl	Urine output < 0.3 ml/kg/h x 24 hours or anuria x 12 hours
LOSS	Persistent Acute Kidney Failure = Complete loss of kidney functions> 4 weeks	
ESRD	End Stage Renal Disease (ESRD) Complete loss of kidney function> 3 months	

Table 4.10: RIFLE Criteria for AKI prediction. The most left column shows the produced label for the pretrain task. While both of the other columns show the criteria for the risk factors. [8]

calculate which we use a formula define by Závada et al. [81]

$$Baseline_{Cr} = 0.74 - 0.2 * int(isFemale) + 0.08 * int(isBlack) + 0.003 * age$$

Lastly, we cleaned our texts using the iteration two pipeline and assigned the produced labels by date and patient id. The balance of the dataset can be observed in figure 4.5

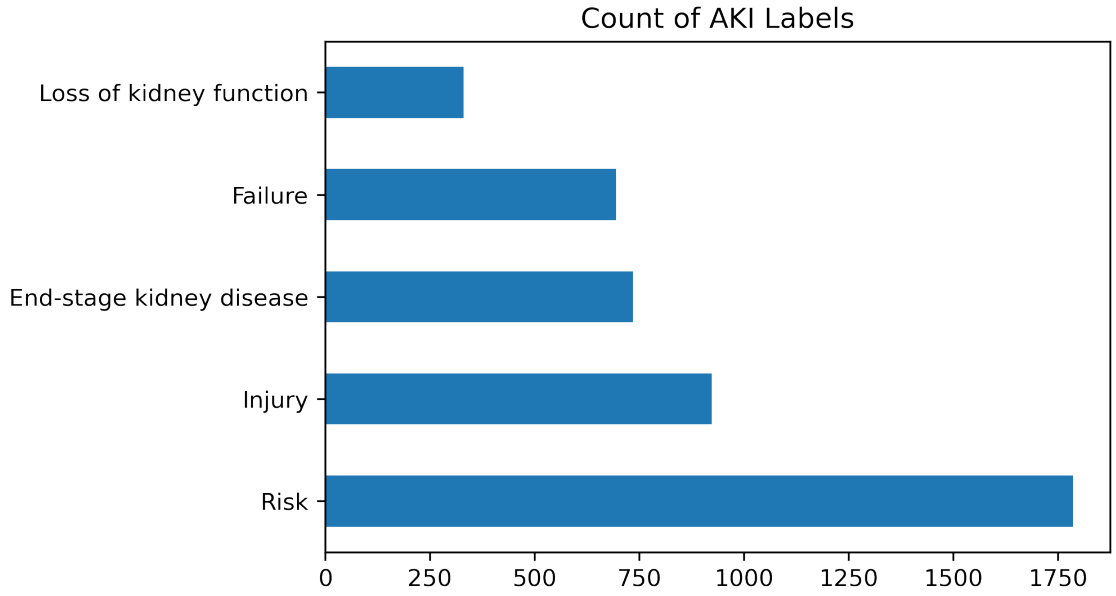


Figure 4.5: Balance of pretrain Dataset (logarithmic x-scale). The no risk label is still dominant in the dataset even when there are multiple labels.

Hyperparameter selection As with the previous model, we also conducted a hyperparameter search for the pretrain process. Table 4.11 shows the grid search and resulting parameters. The model was pretrained on 101.217 examples.

Hyperparameter	Grid Search Minimum	Grid Search Maximum	Grid Search Steps	Final Parameters
Epochs	2	6	1	5
Batch Size	16	64	8	64
Learning Rate	1e-7	1e-4	1e-1	6.0353398951111545e-06
Warm-up Steps	1	5	1	2

Table 4.11: Hyperparameter Optimization and resulting hyperparameters of BERT pretraining.

We finally trained the gBERT model and extracted the embeddings from it. The BERT model was trained using the same balancing techniques as the MLP text model.

4.4 Ensemble Learner

As the goal of this thesis is to prove that the addition of text will improve the prediction of graft failure after a kidney transplant, a model architecture was needed that combines the two vector representations for text and tabular data. Simply concatenating the vectors would have been an option, however, the length for only the text representation was 768 and adding the 299 tabular features to it resulted in a 1067 long input vector. The total number of examples available to us were 1263. Thus, hitting the curse of dimensionality was very likely, prompting us to leave out this design. Instead we went with an ensemble approach, which allows for a meaningful downsampling of the embedding size before concatenating.

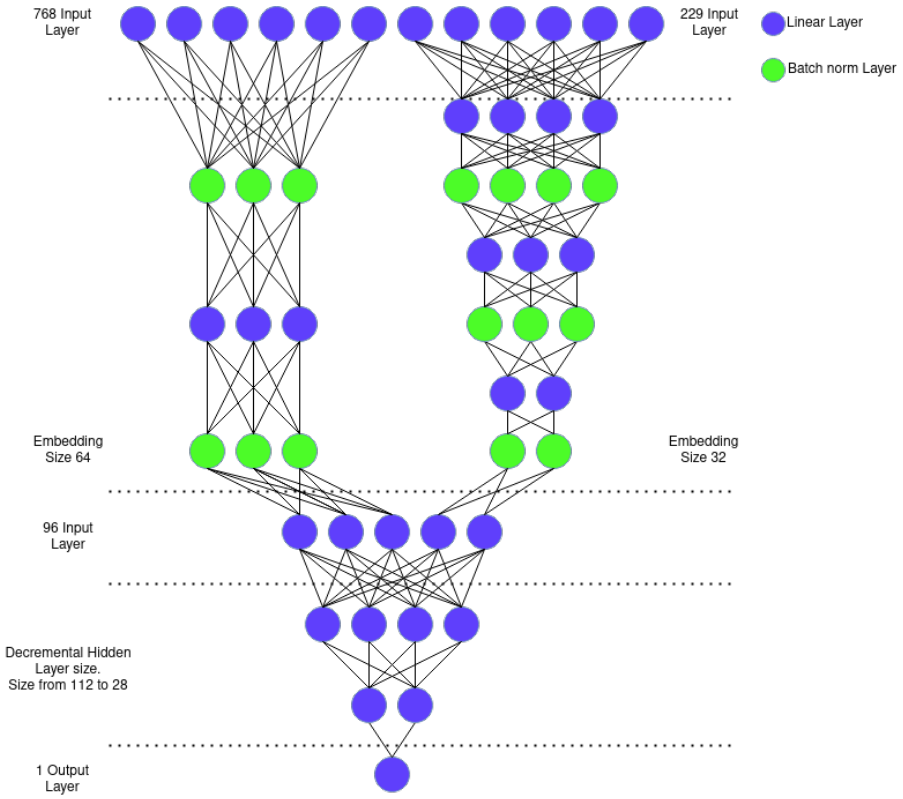


Figure 4.6: Ensemble Model Architecture. Beginning at the top of the figure the two previous models mentioned in the sections 4.2.2 and 4.3.4 receive the data they have also been trained on previously. The prediction head of the models has been removed and the resulting embeddings are concatenated and passed to the ensemble head with an input size of 96.

Model Architecture. We did not run experiments for this model. Instead we used the best-performing and overlapping techniques from the previous two models, which were class weighting and weighted random sampler plus class weighting. So two ensemble models had to be created with different layers and hyperparameters. All the parameters of the text model regarding the BERT model and feature aggregation were kept the same, as well as the transformations used on the tabular data. To construct the model, we pretrained the text and tabular model, removed the last layer and kept the layers from training. We concatenated the outputs of the models and used them as input for the head of the ensemble model. Figure 4.6 shows a generalized architecture for both of the ensemble models. As loss function we used binary cross entropy loss.

Hyperparameter Selection. We again conducted a hyperparameter search. This time we increased the ranges even further for epoch, batch size, layers and hidden dimensions to find suitable parameters. We also enabled the algorithm to choose between a decremental and homogeneous architecture. A further option could have been to unfreeze the pretrained model layers, however we decided to neglect this as a hyperparameter and always went with the freezing option. Table 4.12 shows the selected parameters for the search and their ranges, as well as the resulting parameters for the class weighting and class weighting + weighted random sampler model.

Hyperparameter	Grid Search Minimum	Grid Search Maximum	Grid Search Steps	CW	CW+WRS
Epochs	100	8000	200	6000	2500
Batch Size	64	1024	64	1024	64
Learning Rate	1e-8	1e-4	1e-1	1e-5	1e-5
Regularization Lambda	0.01	0.07	-	0.1	0.1
Layers	0	4	1	2	0
Hidden Dimension	16	128	8	112	48
Decremental	[True, False]	-	-	True	False
Freeze Layers	-	-	-	True	True
Class Weights	-	-	-	0.0102, 0.5525	0.0102, 0.5525

CW - Class Weighting WRS - Weighted Random Sampler

Table 4.12: Hyperparameter Optimization and resulting hyperparameters of the ensemble model. Interesting to note is the high number of epochs and the small number of hidden layers on the CW+WRS model.

4.5 Summary

This chapter covered the full implementation used to prove our hypothesis. It started by stating the hypothesis that medical text is complementary to medical tabular data and went on with defining the problem. We then described the Random Forest baseline model which we were competing against. Furthermore, a description of the feature selection and preparation, done by Haldar et al. [26] and adopted by us, was given. Striving away from the baseline approach, our first model, the tabular MLP, was introduced and explained in great detail, from hyperparameters to architecture, followed by an overview of our medical text data cleaning pipeline and the same information model related information for the MLP text model. A short excursion detailing the fine-tuning attempt can be found after that. The labeling procedure and the setup of the BERT model is explained in this part. Lastly, we presented the ensemble model which resulted from combining all the aforementioned factors.

5 Implementation

In this section we will cover how the in the previous chapter mentioned implementations have been done. We first will take a close look into the produced experiment and train environment analyse there hardware an software.

5.1 Environments

A digital environment is a dedicated space in which all the requirements to run a piece of software or an application are met. Those requirements include processing resources, such as RAM, GPU or CPU, additional software components, global variables and the structure the environment itself is made from (Docker Container, Conda Env). [82] Usually in a software project, there are the development and production environments but when working with machine learning, a third one is added, the training environment. While the development environment is used to program and test the application, the production environment grants the customer access to the application. Meanwhile, the training environment exists to train a machine learning model.

In this section we will cover the construction of our experiment and training environments.

5.1.1 Experiment Environment

In the experiment environment we conducted small scale experiments that would give us an outlook of which led us to choose the architecture. Furthermore, we also evaluated our models and data in this environment.

Processing Resources As the only processing unit provided was a CPU, working on this machine was rather slow. The CPU is not state-of-the-art and a GPU or TPU is completely missing. The operating system was a default Windows 10 installation and was not changeable. This led to some problems with the training environment, as it runs on Ubuntu. The provided 9GB of memory proved insufficient for some cases.

Software Packages We used Python 3.8 as our main interpreter and pip as a package manager. As further packages for data visualisation we used seaborn 0.11.0, pandas 1.1.3, mathpltlib 3.3.2 and numpy 1.15. Furthermore, Jupyter Notebook was used to prototype.

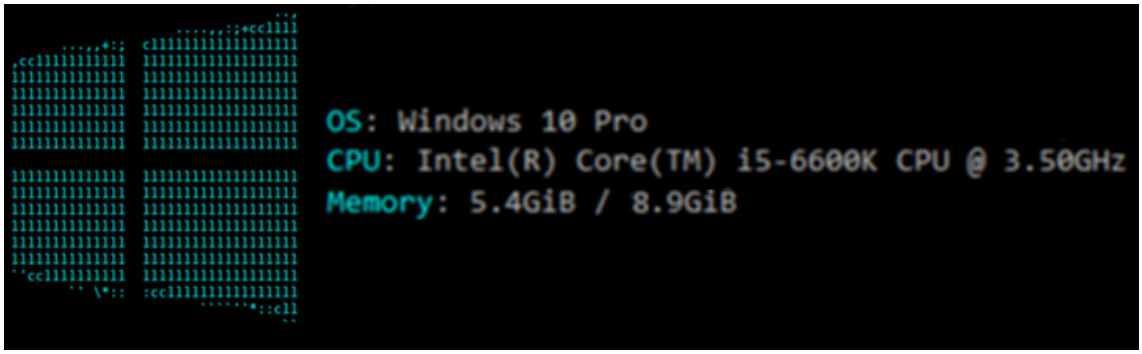


Figure 5.1: Used operating system and hardware components in our experimental environment.

5.1.2 Training Environment

The training environment given to us was much more capable and was solely used to do intense training setups.

Processing Resources As we were fine-tuning BERT on ca. 60.000 Text and were extracting about 6500 embeddings per run, potent resources were needed to shorten the processing time and allow for such large amounts of data to be held in the RAM. Figure 5.2 shows our available resources during the project for the training environment.

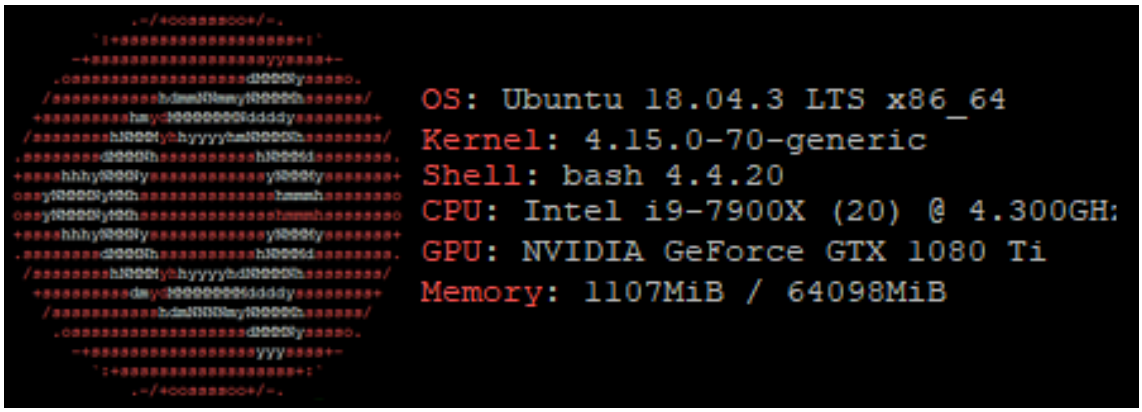


Figure 5.2: Used operating-system and hardware components in training environment.

The provided CPU is state-of-the-art and the RAM holds a reasonable amount of space. However, the most important component for a training environment is the GPU or TPU which calculates all transformations of the tensors during training. The one available to us, while not state-of-the-art or principal for ML tasks, was entirely sufficient for our use case. As an operating-system Ubuntu 18.04.3 was provided.

Software Packages Our main programming language used was Python 3.8. Python offers the option to import third party packages to harness their implementations. To build and train our models, we used PyTorch 1.7.1 and Huggingfaces Transformers 4.5.0. Processing the data was done by using pandas 1.1.3 and scikit-learn 0.23.2. For the hyperparameter optimization we used hyperopt 0.2.5 for the MLP models and optuna 2.7.0 + transformers hyperparameter search engine. Lastly, we used conda to create our packaging environment and the shell engine to execute our

scripts.

To port our versions onto this machine, we used git. A GPU-friendly Docker version was chosen, however, we ran into proxy-server problems that rendered us unable to pull the repository. Consequently, we defaulted to running our scripts directly on the OS. This had the unfortunate side effect that the script would stop running whenever the server connection was lost. To circumvent the problem, we used hohub to demonise the python scripts.

5.2 Flow of Application

The full flow of data through the application can be seen in illustration 5.3. Two endpoints are necessary to be contacted. Data expected is unstructured medical text and tabular data. The data then experiences the implemented full-on cleaning methods implemented and gets passed as an input to the two base models, which, in return, produce the embeddings for the ensemble model.

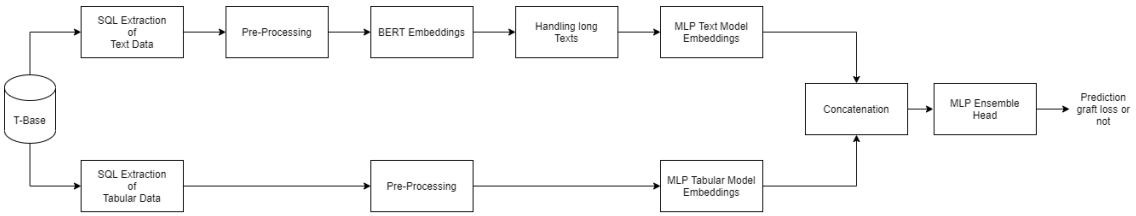


Figure 5.3: The full flow of data through the application. From the extraction of the data from T-Base to the actual prediction, the illustration shows, from left to right, every step in a macro view that is conducted to get the results.

5.3 Summary

In this chapter, we discussed the environments provided to us to work in. Additionally, we gave an overview of the full flow of data through the application. Machines not sharing the same operating system are sometimes difficult to get to work in tandem, as some requirements are different. Furthermore, the choice of python packages was laid out and their usage explained.

6 Evaluation

In this chapter, the results of created throughout all the experiments we conducted are covert, evaluated and interpreted. Furthermore, we conducted a qualitative and quantitative error analysis. The first section starts by covering the results produced in the machine learning experiments covered in the sections 4.2, 4.3, 4.4 and ends on comparing the them with the results of the baseline model provided by Haldar et al. [26], while the second section investigates the flaws and wrong predictions caused by data and models, i.e. error analysis.

6.1 Results

This section will cover the results collected throughout all of our experiments. The results will be explained and discussed, as well as compared and interpreted in each section. As we have two base models which are combined into a single ensemble model, it is important to not only discuss the end results of the ensemble. To fully understand the results, the metrics of the base models have to be investigated, as well. We will begin by examining the results produced by the tabular model, followed by the text model and, finally, the ensemble model. The last section will compare the results of all models with the baseline. [26]

How to Read We considered the F1-Score to be the most important metric for our project, as it is the harmonic mean of precision and recall and represents imbalanced datasets well. Further it is used frequently as a evaluation metric for classification tasks, thus makes our work comparable. Our design decisions for the final model were also based on the F1-Score. We decided to mark the decision making metric with a green color. Furthermore, the best results for a single metric is always outlined as bold type. Other important scores will be marked as blue, however, the reason for their importance will vary and be explained in text. Lastly, the worst-scoring model will be colored red. All scores displayed are macro averages.

Furthermore, the term homogeneous layer architecture refers to a feedforward layer architecture, where each hidden layer has the same number of nodes, while a decremental layer architecture refers to one where the number of nodes decreases with each following layer. This is illustrated in figure 4.4 and 4.2.

6.1.1 Tabular Model

As mentioned in section 4.2.1, we conducted experiments to find the best-suited balancing technique and model architecture. The results of this process are reproduced in table 6.1.

Experiments Colored green the table 6.1, a decremental MLP architecture supported by WRS scores the highest F1-Score of 0.493, closely followed by a model using plain data scoring with 0.492. The weakest results were produced by a combination of SMOTE and a non-decremental architecture only reaching an F1-Score of 0.248. It is noteworthy that the heterogeneous architecture using weighted random sampler scores much higher on recall and F2-Score, thus making it better at predicting failures. Even though the failure class is considered to be the more important one, we chose the F1-Score as our decision-making metric, hence we did not proceed with a homogeneous layer architecture.

	Metrics	Simple	CW	WRS	WRS+CW	FL	SMOTE
MLP (Homogeneous)	Recall	0.507	0.586	**0.654**	0.514	0.515	0.439
	Precision	0.500	**0.510**	**0.510**	**0.510**	0.500	0.390
	Accuracy	**0.858**	0.462	0.596	0.734	0.736	0.311
	F1-Score	**0.479**	0.336	0.402	0.442	0.443	**0.248**
	F2-Score	0.506	0.568	**0.620**	0.511	0.512	0.449
	AUC	0.507	0.586	**0.654**	0.514	0.515	0.439
	MCC	0.006	0.046	**0.084**	0.009	0.009	-0.036
MLP (Decremental)	Recall	0.493	0.464	**0.495**	0.442	0.480	0.440
	Precision	0.490	**0.500**	0.490	0.490	0.490	0.230
	Accuracy	0.968	0.773	**0.971**	0.868	0.942	0.863
	F1-Score	**0.492**	0.447	**0.493**	0.465	0.485	0.463
	F2-Score	0.493	0.47	**0.494**	0.451	0.482	0.449
	AUC	0.493	0.464	**0.495**	0.442	0.480	0.440
	MCC	-0.016	-0.024	**-0.014**	-0.049	-0.028	-0.050

FL - Focal loss CW - Class Weighting WRS - Weighted Random Sampler

Table 6.1: Numeric MLP Experiments. Best scoring architecture can be noted down as WRS + decremental layers, followed closely by a plain use of the data. Still interesting to note is that a homogeneous architecture is able to predict failure better.

Chosen Architecture Evaluating the experimental scores, we selected a decremental architecture combined with either a weighted random sampler or a pure data approach. After finding fitting hyperparameters for both of the architectures, the pure data approach significantly outperforms the WRS approach regardless of the metric, as seen in table 6.2. A thorough investigation of the comparison on the final models closely the pure data attempt, might outperform in every metric, however the differences are minimal. Except for recall, which shows a discrepancy of 6%. Metrics directly related to the recall, like F2-Score, also show a significantly higher result. A alteration from the baseline RF model to a MLP was needed to extracted the embeddings and be able to combine them with the text data.

Interpretation The experiments show clearly that using SMOTE is the weakest approach. We hypothesize that this is due to the fact that all experiments used the same number of epochs and the same learning rate for training. Over-sampling techniques fall short of being effective in this

	Metrics	Simple	Weighted Random Sampler
MLP (Decremental)	Recall	**0.61**	0.55
	Precision	**0.54**	0.52
	Accuracy	0.93	**0.94**
	F1-Score	**0.55**	0.53
	F2-Score	**0.6**	0.54
	AUC	**0.61**	0.55
	MCC	**0.13**	0.07

Table 6.2: Numeric Decremental MLP Final Model. The table shows the two models that were shown to be the best-performing ones in our experiments (table 6.1). An approach that uses the data without any manipulation comes out on top.

scenario since, to learn a higher amount of data, more epochs or a higher learning rate would be needed.

We further believe that a high recall could be achieved by using a homogeneous layer architecture, since the layers offer more budget for the rare classes to be learned and remembered, thus being more capable of predicting them.

As for the final models, the most prominent difference in their design lies in how the distribution of their batches is managed. This might have led to the network not paying as much attention to the minority classes, even though the weighted random sampler was set up to distribute the minor class more uniformly. But as the confusion matrix in figure 6.1 shows, the difference in prediction of failure is only one example apart.

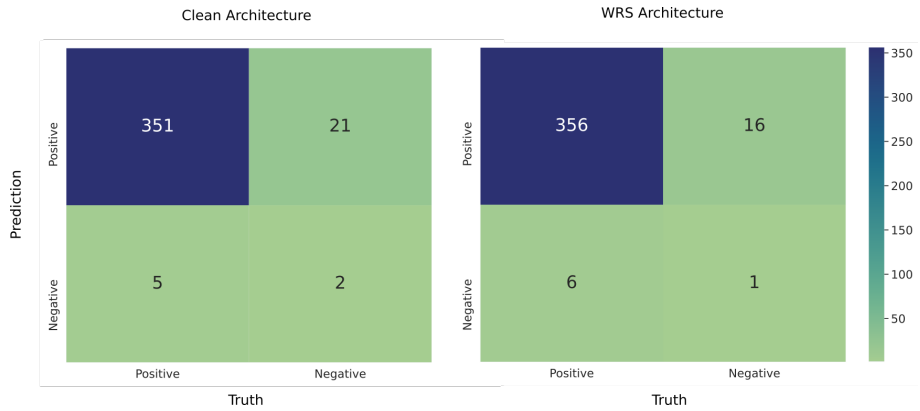


Figure 6.1: Confusion matrix of tabular models. Note the difference in false negatives between the models, leading to the high difference in recall.

6.1.2 Text Model

To find a suiting combination of BERT model, balancing technique and method to handle long texts we conducted a myriad of experiments visible in table 6.3. In this section the results will be explained and analysed.

Balancing Techniques and BERT Model The table shows that embeddings of the gBERT model

	Metrics	Simple	FL	CW	WRS	WRS+CW	SMOTE	SMOTE BL	SVM-SMOTE	SVM-SMOTE+FL	Adasyn	Adasyn+FL	SMOTE ENN	SMOTETomek
gBERT	Recall	0.50	0.587	0.70	0.50	**0.766**	0.635	0.535	0.475	0.487	0.619	0.622	0.598	0.635
	Precision	0.491	0.508	**0.602**	0.491	0.601	0.512	0.504	0.497	0.499	0.510	0.512	0.508	0.512
	Accuracy	0.982	0.739	0.960	0.982	0.953	0.697	0.776	0.794	0.406	0.665	0.259	0.348	0.697
	F1-Score	0.495	0.453	0.633	0.495	**0.642**	0.442	0.459	0.455	0.303	0.428	0.220	0.276	0.442
	F2-Score	0.498	0.569	0.677	0.498	**0.726**	0.606	0.529	0.479	0.490	0.594	0.597	0.577	0.606
	AUC	0.500	0.587	0.700	0.500	**0.766**	0.635	0.535	0.475	0.487	0.619	0.622	0.598	0.635
	MCC	0.000	0.053	0.285	0.000	**0.328**	0.079	0.023	-0.017	-0.007	0.068	0.077	0.056	0.079
MedBERT	Recall	0.500	0.584	0.554	0.500	**0.616**	0.576	0.575	0.588	0.404	0.400	0.473	0.357	0.576
	Precision	0.491	0.508	0.527	0.491	**0.538**	0.506	0.507	0.516	0.490	0.493	0.494	0.479	0.506
	Accuracy	0.982	0.734	0.950	0.982	0.934	0.580	0.715	0.879	0.243	0.509	0.103	0.150	0.580
	F1-Score	0.495	0.450	0.535	0.495	**0.552**	0.389	0.442	0.508	0.204	0.347	0.098	0.136	0.389
	F2-Score	0.498	0.567	0.548	0.498	**0.599**	0.560	0.560	0.572	0.419	0.415	0.477	0.376	0.560
	AUC	0.500	0.584	0.554	0.500	**0.616**	0.576	0.575	0.588	0.404	0.400	0.473	0.357	0.576
	MCC	0.000	0.051	0.077	0.000	**0.134**	0.041	0.045	0.075	-0.061	-0.054	-0.026	-0.109	0.041
Base-BERT	Recall	0.500	0.572	0.553	0.500	0.484	0.492	0.570	0.508	0.657	0.653	**0.663**	0.625	0.492
	Precision	0.491	0.506	0.525	0.491	0.498	0.499	0.506	0.501	0.511	0.511	0.514	0.51	0.499
	Accuracy	0.982	0.710	0.947	0.982	0.813	0.691	0.707	0.860	0.464	0.594	0.338	0.401	0.691
	F1-Score	0.495	0.440	**0.532**	0.495	0.462	0.425	0.439	0.480	0.341	0.401	0.272	0.306	0.425
	F2-Score	0.498	0.557	0.547	0.498	0.487	0.494	0.556	0.507	0.622	0.619	**0.626**	0.598	0.494
	AUC	0.500	0.572	0.553	0.500	0.484	0.492	0.570	0.508	0.657	0.653	**0.663**	0.625	0.492
	MCC	0.000	0.043	0.073	0.000	-0.011	-0.005	0.042	0.007	0.085	0.084	**0.094**	0.069	-0.005

FL - Focal loss CW - Class Weighting WRS - Weighted Random Sampler
ENN - Edited Nearest Neighbor BL - Boarderline

Table 6.3: BERT Models Experiments. The table shows all results of the conducted experiments. The used model can be noted on the left-handed side of the table, while the balancing techniques can be seen on top. Best results for a model and a metric are in bold. Furthermore, the decision making score is marked as green. All experiments have been run with the summing-up technique described in section 4.3.3.

and a combination of weighted random sampler and class weighting performs the best. The weakest embeddings are provided by Base-BERT, underperforming for most of the experiments. However, considering individual results, Base-BERT's embeddings seem to be better-suited for oversampling, as they outperform SVM-SMOTE+FL, Adasyn+FL and SMOTE ENN. For other oversampling techniques it is close to the best result. Albeit higher-scoring on those techniques, Adayn+FL is the weakest performing experiment overall. Further interesting to note is the fact that all three models perform exactly the same when no manipulation on the data occurs or WRS is used, see the column "Simple" and "WRS" on table 6.3. Simple means an absence of any balancing technique.

	Metrics	CW	WRS+CW
gBERT Summed	Recall	0.700	**0.766**
	Precision	**0.602**	0.601
	Accuracy	**0.960**	0.953
	F1-Score	0.633	**0.642**
	F2-Score	0.677	**0.726**
	AUC	0.700	**0.766**
	MCC	0.285	**0.328**
gBERT 512 Tokens	Recall	0.483	**0.693**
	Precision	0.490	**0.573**
	Accuracy	**0.947**	**0.947**
	F1-Score	0.486	**0.602**
	F2-Score	0.484	**0.665**
	AUC	0.483	**0.693**
	MCC	-0.026	**0.238**
gBERT Summed Average	Recall	0.554	**0.623**
	Precision	0.527	**0.552**
	Accuracy	**0.950**	0.947
	F1-Score	0.535	**0.570**
	F2-Score	0.548	**0.607**
	AUC	0.554	**0.623**
	MCC	0.077	**0.160**

CW - Class Weighting WRS - Weighted Random Sampler

Table 6.4: gBERT Method Comparison. Results of the comparison of the methods mentioned in section 4.3.3. Evaluating the table clearly leads to the conclusion that the best-suited token handling method is the method of summing-up.

Token Handling Methods Following our decisions on model and balancing technique, we also

conducted the experiments to find the right token aggregation technique. Table 6.4 shows the results of this setup. Our investigation shows that the best technique is summing-up the tokens. Leading with an F1-Score of 0.642 and beating the other methods by 0.04 and 0.72.

	Metrics	WRS+CW Iteration 1
MLP (Homogeneous)	Recall	**0.754**
	Precision	0.520
	Accuracy	0.654
	F1-Score	0.435
	F2-Score	0.692
	AUC	0.754
	MCC	0.143

CW - Class Weighting
WRS - Weighted Random Sampler

Table 6.5: Text MLP Final Model. Important to note is that even though iteration 2 performs significantly better, the results of the ensemble model result of use iteration 1.

Final Model Concluding the experiments, we decided to use WRS and CW for handling the bias in the dataset. The first iteration shows a high recall and F2-Score, predicting the failure classes, complementing the results of the tabular model well.

Interpretation The model that prevailed in BERT experiments is the gBERT model, which is not surprising, as gBERT has been trained on substantially more data then Base-BERT, thus making it better at generalizing. MedBERT is a fine-tuned version of Base-BERT, hence has seen the same potential, however, having been fine-tuned for the task of disease prediction gave it the edge over Base-BERT, while still being at a disadvantage against gBERT, probably due to the larger dataset.

As the data is hardly distinguishable, as figure 6.2 shows, oversampling generally is not the right tool to enhance the prediction. It may add more samples but it will not make individual samples easier to distinguish. The results show clearly that this is the case, as none of the oversampling techniques were able to beat the simple version of the data, except for the SVM-SMOTE. However it was expected that SMOTE BL and Tomek would perform well, since they clean up the dataset prior to oversampling. Additionally, the same problem mentioned in the previous chapter occurs for the oversampling techniques.

Looking at table 6.5, iteration one is able to predict the failure class almost completely but is also unable to distinguish success classes well from the failure class. Corresponding with physicians, this result is expected, as clinical text data is biased towards failure, since medical staff tends to only write about events impacting a disease negatively.

6.1.3 Ensemble Model

The ensemble model uses the most successful techniques of the two previous models. Using the pre-trained ensemblers, the ensemble model was able to outperform both of them. For this approach the text of iteration 1 has been used. The combination of WRS and CW significantly beat the only CW attempt in all metrics, seen in table 6.6, even producing a 6% higher F1-Score.

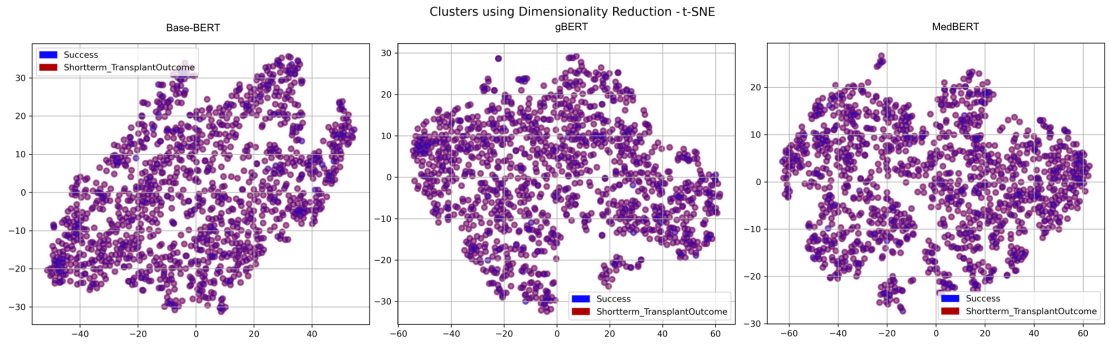


Figure 6.2: Plots showing the text embeddings for each model using dimensionality reduction technique t-sne. The classes of the data plotted are hard to differentiate regardless of the model that produced them, thus making them difficult to classify.

Metrics	CW	WRS+CW
Recall	0.66	**0.69**
Precision	0.53	**0.57**
Accuracy	0.89	**0.94**
F1-Score	0.53	**0.59**
F2-Score	0.63	**0.66**
AUC	0.66	**0.69**
MCC	0.14	**0.22**
CW - Class weighting	WRS - Weighted random sampler	

Table 6.6: Ensemble Learner Results. Showing the results of the two most successful methods of the previous models.

Interpretation The strong result of the ensemble model can explained with the compatibility of the two base models. As the text model is biased towards failure, the text model is able to predict failures the tabular model is not able to and vice versa, thus giving the ensemble head strong hints towards predicting the right class.

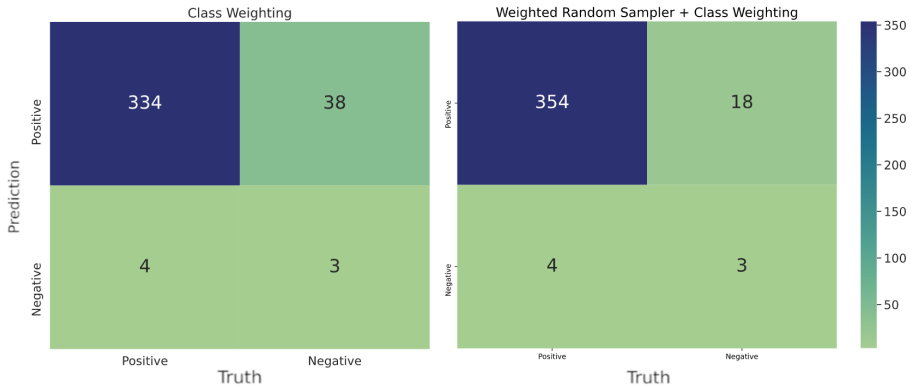


Figure 6.3: Confusion Matrix of Ensemble models. Both methods predict the failure class equally well, however success is predicted better by the WRS + CW approach.

6.1.4 Comparison of Results

Gathering all relevant results produced, table 6.7 shows that the ensemble model performs the strongest in terms of the F1-Score. The baseline is outperformed by 6% on the F1-Score and can record an increase of 16% of the MCC, hence handling the imbalance of the data much better. Of further note is that the tabular model already outperformed the baseline's F1-Score by 2% and that the text model is predicting failure the best.

Metrics	RF Baseline [26]	MLP Decremental Tabular Simple	MLP Homogeneous Text WRS+CW	Ensemble WRS+CW
Recall	0.55	0.61	**0.75**	0.69
Precision	0.52	0.54	0.52	**0.57**
Accuracy	**0.94**	0.93	0.65	**0.94**
F1-Score	0.53	0.55	0.43	**0.59**
F2-Score	0.54	0.6	**0.69**	0.66
AUC	0.55	0.61	**0.75**	0.69
MCC	0.06	0.13	0.14	**0.22**

CW - Class Weighting WRS - Weighted Random Sampler
MLP - Mulit-Layer perceptron

Table 6.7: All Models Comparison. This table shows all results of the models produced in this work, including the RF baseline, clearly showing that the ensemble model outperformed all other models regarding the F1-Score.

6.1.5 Iteration two Text and Pre-Trained

To further denoise the text data we developed our text cleaning pipeline further and pre-trained the BERT model used. The results of these efforts can be seen in table 6.8. In all previous mentioned work only text iteration one has been used, iteration two has not been part of the ensemble model and still needs to be investigated in tandem. However, a comparison of the results of the shallow cleaned iteration one text and heavily cleaned iteration two test is important as it gives insight into what effects the cleaning measures had on the prediction. Furthermore, comparing the pre-trained iteration two approach to the untuned results will contribute to our third hypothesis.

Final Models Iteration one, as in all other tables, still shows the highest recall of all models, which may be due to spurious correlations within the text. Although, the iteration two has a reduced recall but an increased F1-Score of 0.139. As the confusion matrix in figure 6.4 shows, iteration one is much more capable of identifying failures, while at the same time missclassifying successes as failures. pre-training the BERT model on graft loss risk definitions and applying a prediction on the embeddings increases the F1-Score even further. Also, interesting to note is the prediction of the negative classes by the iteration two standard and pre-trained model. As both have the same ratio and possibly the same patient miss-classified (figure 6.4).

Interpretation The bias towards the failure class found in text iteration one, as mention in section 6.1.2, seemed to be having a strong influence on the prediction. The cleaning pipeline of the iteration two text seems to have remedied that, as indicated by the lower recall of iteration two. We believe that this is because of non medical data that influenced the prediction, but got removed in iteration two. Furthermore, pre-training BERT on AKI risk factors and thus introducing him to medical concepts helped improve the prediction. Considering the recall of the iteration one text model and the precision of the tabular model the combination of the two seem to be complementary

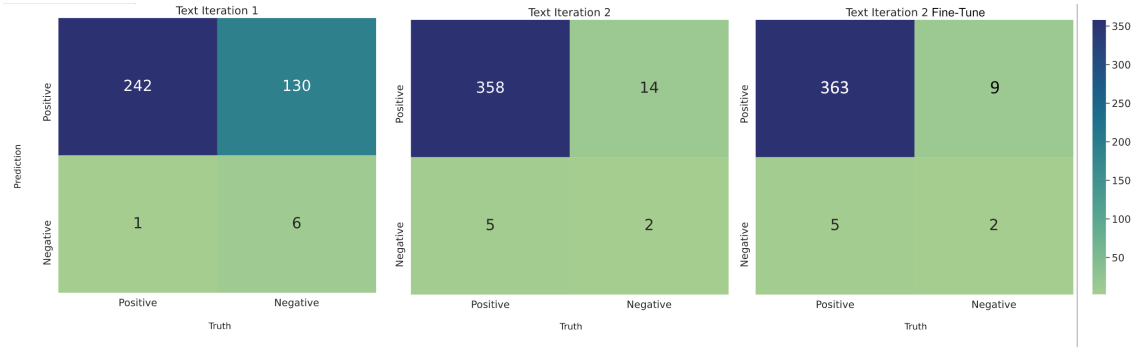


Figure 6.4: Confusion matrix of tabular models.Note especially the difference in false negatives between the models, leading to the high difference in recall.

	Metrics	Text Iteration 1	Text Iteration 2	Text Iteration 2 + Fine-Tune
MLP (Homogeneous)	Recall	**0.75**	0.62	0.63
	Precision	0.52	0.56	**0.58**
	Accuracy	0.65	0.95	**0.96**
	F1-Score	0.43	0.57	**0.60**
	F2-Score	**0.69**	0.61	0.62
	AUC	**0.75**	0.62	0.63
	MCC	0.14	0.17	**0.21**

Table 6.8: Evaluation of Text Model Results. All models used the same balancing technique WRS+CW. However, different text features have been used and pre-training was applied.

(table 6.7). However, considering the high drop of recall in the iteration two models implementing this model in the ensemble architecture, might lead to a weaker performance. Due to the iteration two model lacking the capability of predicting failure patients. So, a clear improvement in terms of ensembling might not be given. Only further experiments will show if text iteration two also performs better in tandem.

6.2 Error Analysis

This section will cover the investigation of the data and the performance of the model. For the qualitative error analysis of the model, we chose to conduct a data ablation test, which shows how well the models learn with more or less data. Following the qualitative error analysis of the text data by investigating features of the text that reduce the predictive capability of the model. To see the robustness of the text model, we also did an ablation of text to assess the robustness of the text model and to see if the model understands the text in regards to the labels. After that, tabular features are examined to reveal their flaws. Additionally, we conducted an experiment where we compared the most important features selected by the model with a selection of features chosen by medical doctors. Lastly, we examined the influence of the two base model on the predictions of the ensemble model.

6.2.1 Quantitative Error Analysis

To cover the quantitative error analysis, we conducted a quantitative data ablation test on all the relevant models. Evaluating the capability of a model to learn with more data is important in the context of an imbalanced dataset, as models tend to overcome imbalance of labels as the amount of data grows. [83] As figure 6.5, shows the tabular model using the simple architecture, the text model used WRS and CW and lastly the ensemble model using WRS were considered. To set up the test, we used 15%, 40% and 70% of the training dataset to train on and always tested on the same test set, choosing the F1-Score as the metric to improve upon. Furthermore, figure 6.6 shows that over-fitting was prevented.

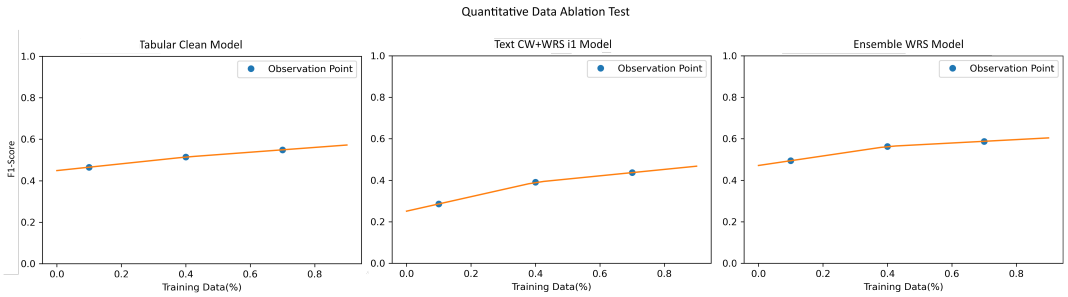


Figure 6.5: Models Data Ablation Test. The Figure shows how the models are able to improve with larger quantities of data. The Y-axis covers the resulting F1-Score, while the X-axis displays the corresponding percentile of data used.

According to figure 6.5, all considered models are able to learn more off larger amounts of data. Differences can be noted in the rise of the curves, as the tabular model improves at a linear rate, while the text and ensemble models learn the most within the first 40% of the dataset. After that the slope of the curve declines.

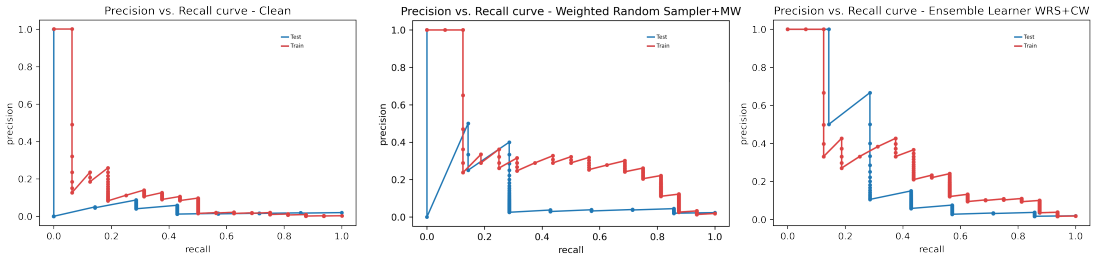


Figure 6.6: Precision and Recall curves of all models. From left to right tabular, text, ensemble. According to the curve all model did not over-fit

Understanding the Tabular model To better understand the decisions of the ensemble model, we looked at the individual predictions of the base models and compared those with the decisions the ensemble model made. The result of this investigation can be examined in table 6.9. We conducted this experiment on the test dataset, which is the same for all three models.

Noting the first row, when both the base models predict all examples correctly, the ensemble model also gets almost all of them right. The reason for this might be that some of the examples have contradictory tabular and text data. The next two rows show how much influence the base

Total Overlap	Tabular	Text	Ensemble R|W
230	Right: 230	Right: 230	224|6
123	Right: 123	Wrong 123	113|10
18	Wrong: 18	Right: 18	13|5
8	Wrong: 8	Wrong: 8	7|1

Table 6.9: Model Comparison Predictions. The tables shows for each row a selection of examples that are overlapping for the, in column two and three, described outcomes (right and wrong). The last column shows the prediction the ensemble model made on the same examples and the first columns shows the total number of examples considered.

model has on the decision of the ensemble model. When the tabular model predicts right and the text model wrong and vice versa, the majority of predictions on the ensemble model are also right. Giving us reason to believe that the influence the tabular model has on the prediction of the ensemble model is higher, despite the embedding size of the tabular model being half as big (figure 4.6). Whenever both base models predicted examples incorrectly, the ensemble got most of them right. This might be due to the fact that the individual model did not have enough information for the prediction individually, however combining their data complements the information.

6.2.2 Qualitative Error Analysis - Tabular Data

This section will cover the quality of the tabular model, as well as the data. First we will investigate the feature quality and, in the end, we will compare the top features for failure prediction chosen by the model and actual medical doctors.

Tabular Feature Analysis

As we investigated the pipeline for tabular features, many flaws became visible. Categorical features became numerical and categories that were supposed to be identical were duplicated due to spelling mistakes.

Imputation - the Death of Categorical Features As mentioned in section 4.1, an imputation was performed on the tabular data to fill in missing values. The imputation technique chosen by [26] Haldar imputes depending on the data available continuous values. The whole dataset was processed by the imputation without removing the categorical values and, as categorical values are discrete, continuous values were introduced to their feature range, effectively converting them to numerical values.

Shortcomings of categorical features Another technique that impaled the categorical features was the one-hot-encoding. Categories that hold only two states, like the sex of a patient, can be depicted as one feature. Applying one-hot-encoding to such features causes them to be split into two features with mirrored values and thus makes one of the features redundant.

Further we found one category of the feature "Spenderart", which points out if the donor was alive, related or dead at the time of transplantation, that had a spelling mistake and through this incident

generated another category, adding "hirntot" as a state next to "Hirntot".

Numerical to Categorical Lab values like the ProteinTUR (amount of protein in urine) have been aggregated over time periods by Haldar [26] (like mean of 3 months, mean of overall etc.), clearly supposed to produce a numerical value. The way most Lab values are stored in T-Base is by an ordinal encoding which goes by the encoding of triple minus to triple plus. The SQL-query creating the mean values, however, only adds a 1 to a counter regardless of the number of pluses found in the encoding, whereby negative values are completely neglected and positive values partially. Furthermore, to calculate the mean, the total count of values considered needs to be calculated, however, only entries that had a plus were added to the total number of values, thus making the ProteinTUR a categorical feature that only states if there was a positive entry or not.

Human vs. Machine

Understanding the prediction a model makes is an important part of error analysis, as this kind of investigation can lead to further improvements when, for example, features gain unexpected importance, which could indicate that the feature may have been prepared in a bad fashion. Figure 6.7 shows the features most important to the tabular model for predicting the failure class. To understand if the features selected by the model make sense, we gave a board[1] of all features available to the model to five physicians and let them pick their top 10 features. To make the picks of the medical doctors more comprehensible, we used the ranking of the selected features as scores and added them up, which resulted in figure 6.8. A full overview of all picks can be seen in the appendix.

Model Picks To evaluate the features of the model, we asked one medical doctor to investigate them. The patient age (second strongest feature) was found to be a strong feature for the prediction, which was also confirmed by the physician. Additionally, further the patient's sex might be a good indicator, as men tend to be riskier and thus have a higher chance of losing kidney. Moreover, the Banff and primary kidney function are very good indicators for the prediction of graft loss, as primary kidney function means that the kidney is already working during the transplantation and the Banff is an indicator for the quality of the kidney. [84] Furthermore, it was not clear to the physician why the model was fixated on the ProteinTUR, instead of creatinine, since creatinine is used as a biomarker for the diagnosis of kidney-related problems. [85] The importance of creatinine in the opinion of medical doctors becomes clear when looking at figure 6.8. The height of a patient also might play a role, but only when the height of the donor diverges strongly, meaning the kidney might be too small or big for the recipient. The diagnosis of a hypertonie is, in the opinion of the medical doctor, highly ambiguous, as it can occur when a future graft loss will happen but it can also be related to any other sickness.

Human vs. Machine Comparing figure 6.7 and 6.8, unfortunately none of the features overlap in ranking. However, some features can be found in both rankings, like the Banff, the primary kidney function and the protein output (Daily Protein) which correlates with the ProteinTUR favored by the model. It is interesting to note that the physicians value the age of the donor more than the age

[1] `https://trello.com/b/mHwx8Xxo/template`

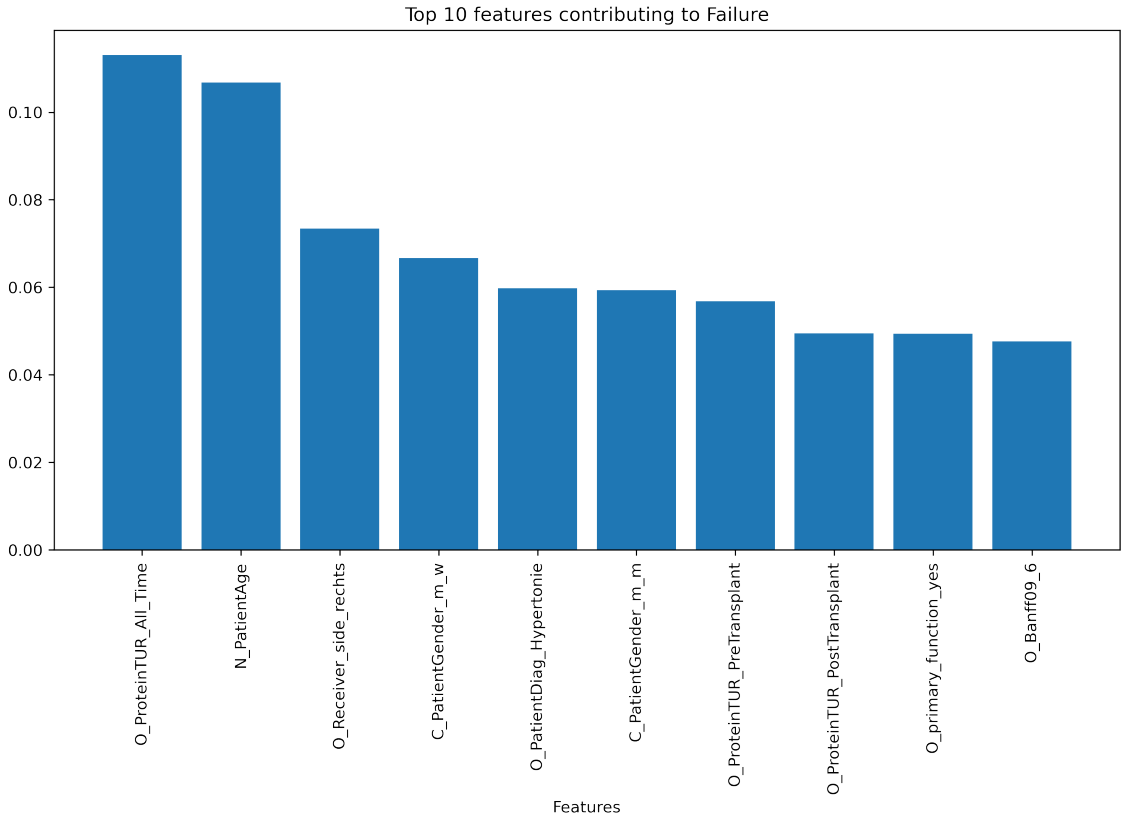


Figure 6.7: Top 10 tabular features ranked by importance for the prediction of failure. The features chosen by the model are mostly protein or demographic related.

of the receiver, while the model thinks otherwise. The most important indicator, in the opinion of physicians, are the creatinine values of a patient. This indicator can not be found at all in the top 10 of the model.

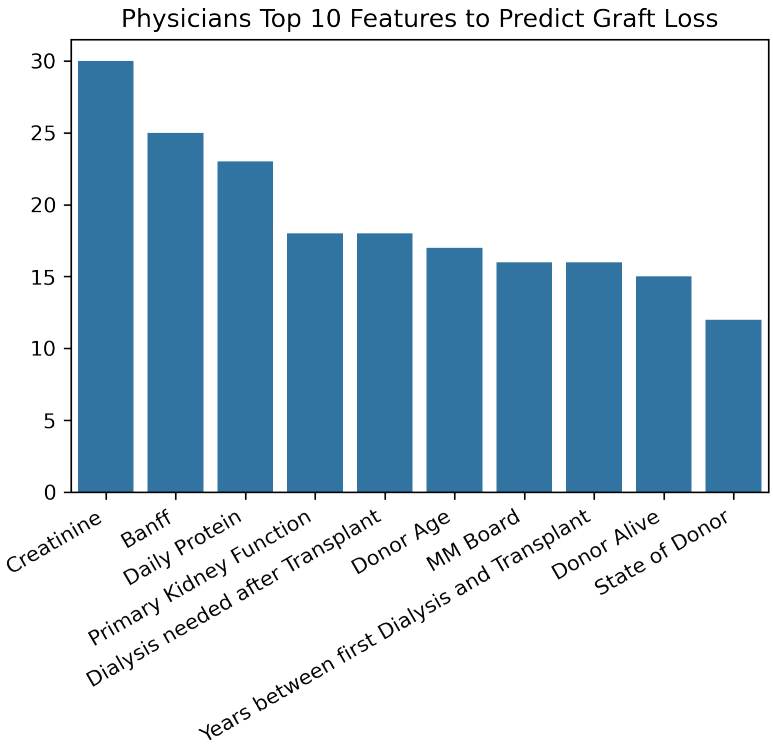


Figure 6.8: Top 10 Features for the Prediction of Graft Loss of a Collective of five Physicians.

Feature Ablation Test

To uncover flaws in the tabular model, we investigated a single missclassified patient profile using the top 10 features and compared it to that of a similar patient whose transplant results had been predicted correctly, seen in table 6.10. Both patients had undergone a graft loss but the patient mentioned in the first row was classified as a success. Features that are differ between the two patients are their age, the primary kidney function and the Banff, so these features will be investigated.

	ProteinTUR All Time	Age	Receiver side rechts	Gender W	Hypertonie	Gender M	ProteinTUR PreTrans	ProteinTUR PostTrans	primary function	Banff9 6
Wrong	1.0	39.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	1.0
Right	1.0	18.0	0.0	1.0	0.0	0.0	0.0	1.0	1.0	0.0

Table 6.10: Comparison of two simular patient examples.

Examining the age features in the training and test set (figure 6.10) shows that the model never saw a patient of the failure class being in the age range between 30 and 45, yet exactly that is the exact age range depicted in the test set. The missclassified patient falls into that age range. In general, the age of failure patients more often tends to be between 60 and 70 years.

Furthermore, a correlation between females experiencing a graft loss and the primary kidney function exists in our dataset, as seen in figure 6.10. Out of 14 female patients in the whole dataset, 9

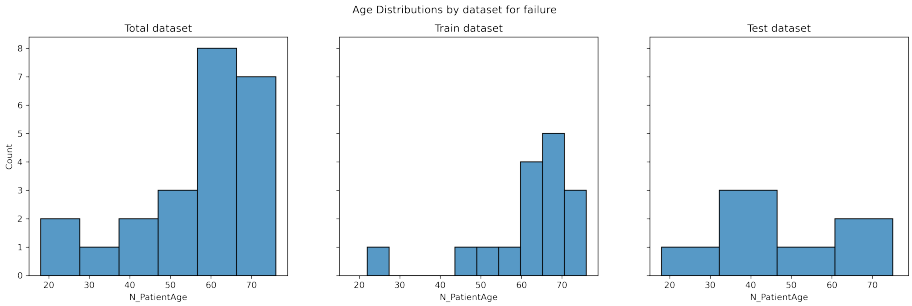


Figure 6.9: Distribution of age by dataset. Interesting to note is the lack of the age span from 30 to 45 in the train set.

have no primary kidney function, while only 4 do have it, as a female receiving a kidney transplant with primary functionality and losing it later is an uncommon combination, thus this example was missclassified.

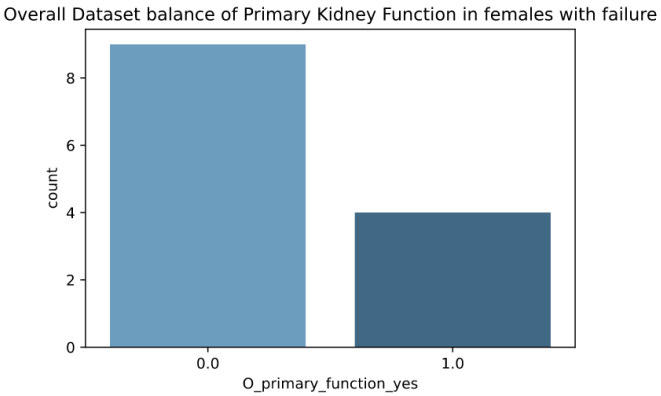


Figure 6.10: Distribution of primary kidney function in female patients experiencing graft loss. A Bias towards not having a primary function can be noted.

To prove that the mentioned features are the cause of the missclassification, we conducted an ablation study, changed the features by hand and observed the outcome in form of the sigmoid values. Originaly, the sigmoid value was at 0.43. Changing the primary kidney function to 1 changed the outcome to 0.45 and additionally setting the Banff to 0 changed the value to 0.54, thus flipping the prediction.

6.2.3 Qualitative Error Analysis - Text Data

This section will explain flaws we found related to the text corpus and text model. To conduct the error analysis, we investigated the weaknesses of the text corpus by evaluating medical abbreviations, numbers found and unknown words produced by BERT. Furthermore, we selected a single patient who underwent two surgeries and evaluated the impact the text copora of the transplants have on the prediction of the model.

Text Feature Analysis

NLP models struggle with certain parts of text, like numbers. [86] Furthermore, using slang terms or abbreviations unseen by the model will diminish its prediction capabilities. BERT especially, if not properly pre-trained, is not aware of characters it has never seen and, thus, does not deliver value in these cases, or even loses some.

Category Aggregation As standardising text will improve the quality of the data, since patterns can be recognized more easily by the model, we added the category of each Untersuchungs text at the start of it. However there were 105 unique categories in the dataset, some of which were abbreviations of others or spelling mistakes. By using an algorithm to aggregate the categories, we were able to condense the total number to 82.

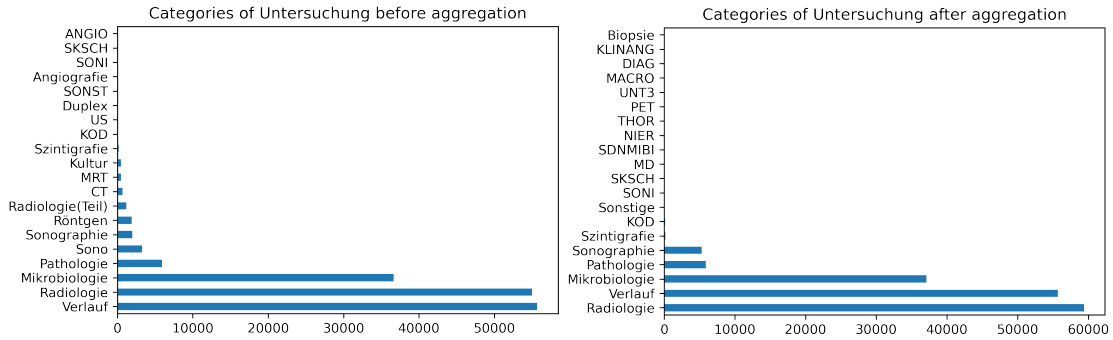


Figure 6.11: Top 20 Untersuchung Categories before and after Aggregation.

As figure 6.11 shows, categories like Sono or Radiologie(Teil) coulde be aggregated into Sonographie and Radiologie.

Abbreviations The medical domain has its own corpus of abbreviations, an NLP model not pre-trained on such a corpus will have difficulties interpreting the text. We found 1423 unique abbreviations in a list of 2514 available ones and, in total, 3.949.758 abbreviations were replaced.

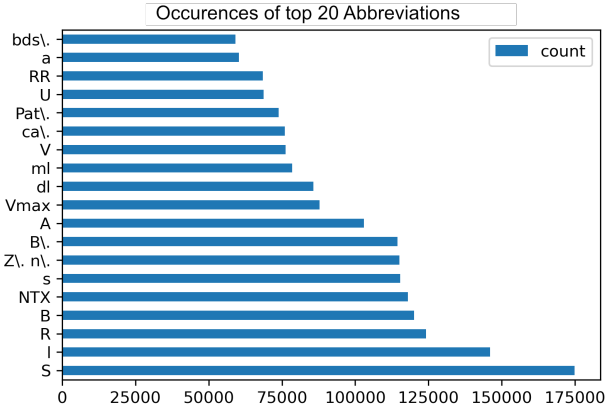


Figure 6.12: Top 20 Abbreviations found in Untersuchung.

Figure 6.12 shows the top 20 abbreviations found in the corpus. Most of the places are occupied by metrics like liter (l) or blood pressure (RR). All abbreviations can be found in the appendix.

Furthermore, a special type of abbreviation, the TNM-Code, was be replaced algorithmically. [72] For example pT1pN0G2R0 translates to:

> *"Tumorausdehnung minimal, festgestellt durch chirurgischen Eingriff. Kein Befall der Lymphknoten, druch chirurgischen Eingriff festgestellt. Tumorgewebe lässt sich eingeschränkt differnzieren. Nach Therapie kein Tumor mehr vorhanden."*

Only 79 cases were replaced.

Numeric Tokens While BERT is able to handle numbers, they pose a weakness to the predictive capabilities of the embeddings. [86] Investigating the dataset, we found 10.305.730 number-only-tokens, which amounts to 7.78% of all tokens.

Unknown Words Words unknown to BERT result in the [UNK] token. The most common reason for BERT not recognising a word is a single character. In our corpus we were were able to locate 62007 unknown words in total originating from 1993 unique words. The total percentage of unknown tokens is at 0.11%. When investigating the responsible characters, we were able to observe that most of them were accented letters, so we replaced them with their unaccented counterparts.In the case of some characters, it became apparent that they were following a certain pattern and could thus be replaced with known letters. Furthermore, mathematical symbols were spelled in full.

Lastly, in some minor instances, notes written completely or partially in English were found. Those might negatively impact the prediction, since a German language model was used.

Feature Ablation Test

To further investigate the robustness of our text model, we considered a patient with two transplants, of which the first had been a failure and the second a success. We deliberately chose this construct, as text from the first transplant should contribute towards a failure prediction, whereas text closer to the second transplant should contribute to a prediction of success. Hence, by carefully neglecting texts, we are able to tell how well the model interprets the text. As an indicator we used the sigmoid values produced by the last layer of the model.

The patient's first transplant took place in 1990, the second one was conducted in 2007. Using all available text concerning this patient, the result of the sigmoid value was at 0.68 and therefore predicting a failure. Using only text data from one year before until one year after the second surgery, the second surgery until one year after the sigmoid value decreased to 0.61. Lastly, when using only data from 2007 to 2008, the sigmoid value decreased even further to 0.57. The prediction of the model is still incorrect, however, the approach shows that the model can interpret the given text right.

6.3 Discussion

Many results have been produced during this thesis, table 6.3 alone features an overwhelming amount of results. As the results are so manifold, a discussion is needed to give an understanding of all results and their interrelation.

As the evaluation of the given hypothesis will also take place in this section, they will be repeated once. Our hypothesis are:

1. That medical text holds information complementary to the tabular data of the patient and will therefore, by combining the two data types, improve the performance of the model.
2. An MLP model using a balancing technique will outperform a RF model on the same imbalanced dataset.
3. Further pre-training BERT with AKI risk-factors that will improve graft loss prediction on clinical notes.

Tabular Model Closely examining table 6.7, the F1-Score of the tabular model is at 55% outperforming the RF model by 2% and thus confirming our second hypothesis partially. Partially, since the best chosen architecture for the tabular model, resulting from the experiments, was the simple architecture not using any balancing technique. Surprisingly, while the dataset used is highly imbalanced the simple architecture should theoretically not achieve the best results, as it does not handle the imbalance.

Text Model The final text-iteration-one model performs weaker on the precision, however, its recall is through the board the strongest. The capability to predict failures might be due to a bias in the text data, as medical doctors only tend to write down findings that contribute towards a disease. The high recall indicates that the text model is able to predict patients, that the tabular model is incapable of predicting, thus basing our first hypothesis.

Ensemble The ensemble model is able to outperform the baseline with a rise of the F1-Score by 6%, as well as the individual text and tabular models. (table 6.7) This shows that tabular and text data found in the medical database are complementary, thus proving our first hypothesis completely. The scores of the two individual models reflect our first hypothesis, since the recall for the text model is high, while on the other side the precision of the tabular model is also high. The ensemble model itself does not reach an as high recall as the iteration-one-text model, but has the second highest of all models, trading off some of its recall to profit off the predictions from the tabular model.

Fine-Tuning Text Furthermore, we managed to pretrain the gBERT model on AKI risk-factors, thus making the embeddings aware of the concept. Letting the model learn the risk-factors increases the F1-Score by 3%, outperforming the not pretrained iteration-two-text model and thereby proving our third hypothesis.

Quantitative Error Analysis The quantitative error analysis shows that all models are able to learn from more data, making them suitable for a data product. Furthermore, table 6.9 shows the

influence of the individual models towards the predictions of the ensemble model. A stronger influence by the tabular model can be noticed. The fact that the ensemble model can predict examples that both base models missclassify individually, adds to our hypothesis, proving that both data sources contain complementary data.

Qualitative Error Analysis The investigations of the qualitative error analysis show that most features of the tabular model are manipulated in a way that might lower the quality of the data. Noting this, the tabular model might hold a much bigger potential. Moreover, the top 10 feature picks of the model resemble this, as the most important features picked by medical doctors are not even in the top 10 (figure 6.7 and 6.8). The features ablation test reveals weaknesses of the dataset, as the distribution of age is lacking certain ranges in the training dataset. This is due to the fact that the dataset is highly imbalanced towards the success class and has a low number of samples in general.

6.4 Summary

In this chapter, we examined the results of all models and thus were able to prove our hypothesis. Further a deep dive in to the flaws of the project was conducted. The results show clearly that an improvement of the F1-Score is possible by combining medical text and tabular data, thus proving our first hypothesis. Furthermore, our second hypothesis is proven by the fact that our MLP tabular model is outperforming the baseline RF baseline model by 2%. In the end, our third hypothesis is proven through pretraining the gBERT model on AKI risk-factors and increasing the F1-Score by 3%.

7 Conclusion

This thesis took a deep dive into the medical dataset of kidney transplant patients available in the T-Base database of the Charité Berlin, providing an attempt for cleaning and preparing the text data and a model to combine it with medical tabular data. By constricting the ensemble model we were able to provide a deep insight into the individual capabilities of the two data types. Furthermore, we investigated the effect of letting a NLP model learn risk factors to better predict a disease.

Results Our approach of combing text and tabular data using an ensemble model scores a 59% of macro average F1-Score and thereby outperforms the Random Forest baseline by 6%, thus proving our first hypothesis of medical text data complementing tabular data of a patient. Our first hypothesis is further strengthened by the fact that the two base models (MLP tabular 55% and text 45% F1-Score) were unable to outperform the combined ensemble model. However, we went a step further and pretrained gBERT on AKI risk-factors, to increase the quality of the embeddings. The success of this approach can be witnessed in the 60% F1-Score produced by a MLP text model, proving our third hypothesis of learning risk-factors increasing the predictive capabilities of a BERT model. Lastly, our second hypothesis got proven by the fact that our MLP tabular model outperforms the random forest baseline by 2% of F1-Score.

Data Achievements Moreover, we revealed problems with the provided tabular data processing pipeline, where features were processed in a way that introduced a lot of noise to the data. The top-most important features for the tabular model support this finding, as the value deemed most vital by medical doctors for a graft loss prediction - creatinine - cannot be found in the list of most important features for the tabular model. Investigating a single example of a misclassified patient revealed that the small size of the dataset has its limits when splitting, as the distribution of the age feature is missing an equal amount of ages and failure patients in some age ranges. Examining the provided text corpus, we were able to highlight that the text requires intensive cleaning to provide a suitable representation. To achieve this representation, we took several steps, most notably expanding abbreviations, cleaning words unknown to BERT and removing text that was not contributing any valuable information.

Limitations Each project has its limitations mostly to the factor of time. To find the right balancing technique a multitude of methods have been evaluated, to keep the expedition of methods within scope we used uniform hyperparameter for all the methods. Some of the methods might suffer from this choice, thus not delivering accurate results.

Concluding the error analysis we found flaws that add a reasonable amount of noise to the tabular data. Furthermore, we found unwanted characters in T-Base, which might be part of a bug we

were not able to investigate.

We further observed a drop of recall from text-iteration-one to two model, but were unable to investigate the cause of this drop, due to time limitations. Moreover, we were unable to combine the iteration-two-pretrained-text model with our ensemble model, so that we are unable to state if the text-iteration-two model will improve the performance of the ensemble model. This serves as a point of entry for future research.

Résumé Given the flaws found in the tabular model and the rudimentary approach of abbreviation expansion not taking the context of the abbreviation into account and further not having implemented the fine-tuned text model on the ensemble model, we believe that this model still holds a lot of untapped potential. In addition, the dataset will grow over time, as new patients will, unfortunately, be included in the database and offer a better distribution of classes and features. Even though we were able to prove our hypothesis and increase the prediction, there is still room for

7.1 Future outlook

Future improvements can be made in many areas but three sectors come to mind first and foremost: data engineering tasks, improvement of data quality and the continued search for a better model architecture. The tasks we deem the easiest to implement or the most essential to make further progress on the project can be found under short-term improvement. The mid-range improvements should be conducted after the conclusion of the short-term improvements. However, those are tasks which we think will have the most impact and need the least amount of time. Long-term improvements are tasks that could not be easily managed during the course of one thesis. In the backlog, a handful of interesting references can be found, which could be tried out, but we believe that they will lead to equal results.

Short-term improvement An easily implemented improvement would be to upgrade the ensemble model with the fine-tuned text model. This will increase the F1 score by approximately 2-3%. An investigation of the oversampling techniques with suitable hyperparameters could also bring further insight which might help create a better model. The infrastructure to achieve this is already implemented in our code-base. A rather long but absolutely necessary step before proceeding with anything related to the tabular data is the abolishment of all the bugs found in the data preparation pipeline.

Furthermore, to modernise the fine-tuning approach, a different risk definition was proposed to us by a medical doctor during the project: the Akain definition for diagnosing AKIs. [87] Additionally, neglecting the time factor for the classifications of an AKI by the RIFLE definition was recommend to us. The produced labels should also be used as an input feature for the tabular model.

An investigation of the sudden drop of recall from text iteration one to two might reveal further insight into the bias of the text data and should thus be conducted. Another investigation should be conducted on the five failure patients who were misclassified by the iteration two text models. Lastly, we took patients' text data that appeared prior to a transplant. Our investigation of the

text ablation study showed that, if a patient had had two transplants, the text of the first transplant would contribute to the wrong class. Thus, a time limit needs to be defined to limit how much pre-transplant text should be included for a prediction.

Mid-Range improvement As tabular and text data are combined in this thesis, another dimension of data can be added to the prediction - time. Implementing this parameter can be achieved by using an LSTM or T-LSTM. [88]

Furthermore, our technique of splitting long texts is quite rudimentary, as it just splits the text every 512 tokens. A better approach would be to use spaCy and split the text in a way that keeps the number of tokens in a batch as high as possible, while also respecting its grammatical structure.

The ensemble approach is quite successful and can be applied to other structures of the project. As each Untersuchungs text belongs to a category, an NLP model for each of the 4-5 largest categories could be trained which generates new text embeddings combining all of those models.

As we were conducting this project, a new German BERT model was released. GottBERT, as it is called, looks promising and should also be evaluated. [89]

Long-term improvement The most important goal for the medical sector in the next few years should be to build a common data platform for all hospitals in Germany. Moreover, UIs should be generated that try to standardise the input of the data for tabular data. For text data, it is necessary to at least have a recommendation engine that makes suggestions so that the collected text becomes more uniform. Lastly, a German BERT, or whichever will be the language model of the future, will need to be created which is fully capable of understanding and working with German medical domain text.

Backlog

- Dice loss and dice coefficient [90]
- MCC as loss function [91]
- Constrastive loss n-shot model [92]
- FLANNEL Ensemble Model [70]
- Ensemble Tree Layer [93]
- Other Risk Factors for Fine-Tuning [94]

Bibliography

[1] M. Aguiar, A. Dias, and M. Field, "Feedforward networks: Adaptation, feedback, and synchrony," *Journal of Nonlinear Science*, vol. 29, pp. 1129–1164, 2019.

[2] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *ArXiv*, vol. abs/1706.03762, 2017.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *NAACL-HLT*, 2019.

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

[5] B. Chan, "Open sourcing german bert."

[6] B. Chan, S. Schweter, and T. Möller, "German's next language model," *ArXiv*, vol. abs/2010.10906, 2020.

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

[8] S. W. Hussain, A. Qadeer, K. Munawar, M. S. S. Qureshi, M. Y. Khan, A. Abdullah, S. Bano, and Z. S. Shad, "Determining the incidence of acute kidney injury using the rifle criteria in the medical intensive care unit in a tertiary care hospital setting in pakistan," *Cureus*, vol. 11, 2019.

[9] R. Dobbs, W. R. Halgrimson, I. Madueke, H. T. Vigneswaran, J. O. Wilson, and S. Crivellaro, "Single-port robot-assisted laparoscopic radical prostatectomy: initial experience and technique with the da vinci® sp platform," *BJU International*, vol. 124, 2019.

[10] Y. Hao, L. Wang, J. Wen-bo, W. Wu, S. Ai, S. Lu, S. Zhao, and K. Dai, "3d printing hip prostheses offer accurate reconstruction, stable fixation, and functional recovery for revision total hip arthroplasty with complex acetabular bone defect," *Engineering*, vol. 6, pp. 1285–1290, 2020.

[11] V. Lestari, R. Rahim, P. T. Nguyen, W. Hashim, and A. Maseleno, "Smart health records," *International Journal of Engineering*, vol. 8, pp. 986–989, 2019.

[12] M. Sun, J. Baron, A. Dighe, P. Szolovits, R. Wunderink, T. Isakova, and Y. Luo, "Early prediction of acute kidney injury in critical care setting using clinical notes and structured multivariate physiological measurements," *Studies in health technology and informatics*, vol. 264, pp. 368–372, 2019.

[13] M. Ijaz, M. Attique, and Y. Son, "Data-driven cervical cancer prediction model with outlier detection and over-sampling methods," *Sensors (Basel, Switzerland)*, vol. 20, 2020.

[14] R. Mehta, J. Kellum, S. Shah, B. Molitoris, C. Ronco, D. Warnock, and A. Levin, "Acute kidney injury network: Report of an initiative to improve outcomes in acute kidney injury," *Critical care (London, England)*, vol. 11, p. R31, 02 2007.

[15] J. Gameiro, T. Branco, and J. A. Lopes, "Artificial intelligence in acute kidney injury risk prediction," *Journal of Clinical Medicine*, vol. 9, 2020.

[16] H. C. Rayner, M. Thomas, and D. Milford, *Understanding Kidney Diseases* -. Berlin, Heidelberg: Springer, 2015.

[17] O. OECD, "Health care resources," 2020.

[18] D. G. O. U. und BVOU, "Die knappe ressource arzt," *Orthopädie und Unfallchirurgie*, vol. 9, pp. 3–3, Jun 2019.

[19] A. Kramer, M. Pippias, M. Noordzij, V. S. Stel, N. Afentakis, P. M. Ambühl, A. M. Andrusev, E. A. Fuster, F. E. Arribas Monzón, A. Åsberg, M. Barbullushi, M. Bonthuis, F. J. Caskey, P. Castro de la Nuez, H. Cernevskis, J.-M. des Grottes, L. Garneata, E. Golan, M. H. Hemmelder, K. Ioannou, F. Jarraya, M. Kolesnyk, K. Komissarov, M. Lassalle, F. Macario, B. Mahillo-Duran, A. L. Martín de Francisco, R. Palsson, Pechter, H. Resic, B. Rutkowski, C. Santiuste de Pablos, N. Seyahi, S. Simic Ogrizovic, M. F. Slon Roblero, V. Spustova, O. Stojceva-Taneva, J. Traynor, Z. A. Massy, and K. J. Jager, "The European Renal Association – European Dialysis and Transplant Association (ERA-EDTA) Registry Annual Report 2015: a summary," *Clinical Kidney Journal*, vol. 11, pp. 108–122, 01 2018.

[20] I. Helanterä, T. Isola, T. Lehtonen, F. Åberg, M. Lempinen, and H. Isoniemi, "Association of clinical factors with the costs of kidney transplantation in the current era," *Annals of transplantation*, vol. 24, pp. 393–400, July 2019.

[21] A. Hart, J. M. Smith, M. A. Skeans, S. K. Gustafson, A. R. Wilk, S. Castro, A. Robinson, J. L. Wainright, J. J. Snyder, B. L. Kasiske, and A. K. Israni, "Optn/srtr 2017 annual data report: Kidney," *American Journal of Transplantation*, vol. 19, no. S2, pp. 19–123, 2019.

[22] B. Jawdeh and A. Govil, "Acute kidney injury in transplant setting: Differential diagnosis and impact on health and health care," *Advances in Chronic Kidney Disease*, vol. 24, pp. 228–232, 07 2017.

[23] R. Weiss, M. Meersch, H.-J. Pavenstädt, and A. Zarbock, "Acute Kidney Injury," *Dtsch Arztebl International*, vol. 116, no. 49, pp. 833–842, 2019.

[24] J. Fox, "Safer care for acutely ill patients.," *British journal of nursing*, vol. 16 15, p. 913, 2007.

[25] S. Pozzoli, M. Simonini, and P. Manunta, "Predicting acute kidney injury: current status and future challenges," *Journal of Nephrology*, vol. 31, pp. 209–223, Apr 2018.

[26] N. Haldar, "Artificial neural network for the prediction of short-term graft failure of transplanted kidneys," Master's thesis, Technische Universität Berlin, 2020.

[27] Y. Li, L. Yao, C. Mao, A. Srivastava, X. Jiang, and Y. Luo, "Early prediction of acute kidney injury in critical care setting using clinical notes," *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pp. 683–686, 2018.

[28] T. Mikolov, K. Chen, G. S. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *ICLR*, 2013.

[29] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.

[30] J. Pennington, R. Socher, and C. Manning, "GloVe: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Doha, Qatar), pp. 1532–1543, Association for Computational Linguistics, Oct. 2014.

[31] A. Luque, A. Carrasco, A. Martín, and A. D. L. Heras, "The impact of class imbalance in classification performance metrics based on the binary confusion matrix," *Pattern Recognit.*, vol. 91, pp. 216–231, 2019.

[32] D. Powers, "What the f-measure doesn't measure: Features, flaws, fallacies and fixes," *ArXiv*, vol. abs/1503.06410, 2015.

[33] A. B. Yedidia, "Against the f-score," 2016.

[34] C. J. V. Rijsbergen, *Information Retrieval*. USA: Butterworth-Heinemann, 2nd ed., 1979.

[35] H. A. Güvenir and M. Kurtcephe, "Ranking instances by maximizing the area under roc curve," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, pp. 2356–2366, 2013.

[36] J. Huang and C. Ling, "Using auc and accuracy in evaluating learning algorithms," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, pp. 299–310, 2005.

[37] T. Fawcett, "An introduction to roc analysis," *Pattern Recognit. Lett.*, vol. 27, pp. 861–874, 2006.

[38] Jin Huang and C. X. Ling, "Using auc and accuracy in evaluating learning algorithms," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 3, pp. 299–310, 2005.

[39] M. Schonlau and R. Y. Zou, "The random forest algorithm for statistical learning," *The Stata Journal*, vol. 20, pp. 29 – 3, 2020.

[40] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2004.

[41] G. Piccinini, "The first computational theory of mind and brain: A close look at mcculloch and pitts's "logical calculus of ideas immanent in nervous activity"," *Synthese*, vol. 141, 08 2004.

[42] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[43] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, "Activation functions: Comparison of trends in practice and research for deep learning," *ArXiv*, vol. abs/1811.03378, 2018.

[44] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2016.

[45] C. Käding, E. Rodner, A. Freytag, and J. Denzler, "Fine-tuning deep neural networks in continuous learning scenarios," in *ACCV Workshops*, 2016.

[46] Q. Chen, Z. Zhuo, and W. Wang, "Bert for joint intent classification and slot filling," *ArXiv*, vol. abs/1902.10909, 2019.

[47] T.-Y. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, pp. 318–327, 2020.

[48] O. Ozdemir and E. B. Sonmez, "Weighted cross-entropy for unbalanced data with application on covid x-ray images," 2020.

[49] P. Efraimidis, "Weighted random sampling over data streams," in *Algorithms, Probability, Networks, and Games*, 2015.

[50] N. V. Chawla, K. Bowyer, L. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority oversampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, 2002.

[51] H. Han, W. Wang, and B. Mao, "Borderline-smote: A new over-sampling method in imbalanced data sets learning," in *ICIC*, 2005.

[52] R. Akbani, S. Kwek, and N. Japkowicz, "Applying support vector machines to imbalanced datasets," in *ECML*, 2004.

[53] K. Veropoulos, I. Campbell, and N. Cristianini, "Controlling the sensitivity of support vector machines," 1999.

[54] G. E. A. P. A. Batista, R. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *SIGKDD Explor.*, vol. 6, pp. 20–29, 2004.

[55] R. Ghorbani and R. Ghousi, "Comparing different resampling methods in predicting students' performance using machine learning techniques," *IEEE Access*, vol. 8, pp. 67899–67911, 2020.

[56] A. Elhassan and Al-Mohanna, "Classification of imbalance data using tomek link (t-link) combined with random under-sampling (rus) as a data reduction method," 2017.

[57] H. He, Y. Bai, E. A. Garcia, and S. Li, "Adasyn: Adaptive synthetic sampling approach for imbalanced learning," *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pp. 1322–1328, 2008.

[58] A. E. W. Johnson, T. J. Pollard, L. Shen, L. Lehman, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. Celi, and R. Mark, "Mimic-iii, a freely accessible critical care database," *Scientific Data*, vol. 3, 2016.

[59] S. Barbieri, J. Kemp, O. Perez-Concha, S. Kotwal, M. Gallagher, A. Ritchie, and L. Jorm, "Benchmarking deep learning architectures for predicting readmission to the icu and describing patients-at-risk," *Scientific Reports*, vol. 10, 2020.

[60] Y. Lin, Y. Zhou, F. Faghri, M. Shaw, and R. Campbell, "Analysis and prediction of unplanned intensive care unit readmission using recurrent neural networks with long short-term memory," *PLoS ONE*, vol. 14, 2019.

[61] T. Zebin and T. Chaussalet, "Design and implementation of a deep recurrent model for prediction of readmission in urgent care using electronic health records," *2019 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, pp. 1–5, 2019.

[62] S. Darabi, M. Kachuee, S. Fazeli, and M. Sarrafzadeh, "Taper: Time-aware patient ehr representation," *IEEE Journal of Biomedical and Health Informatics*, vol. 24, pp. 3268–3275, 2020.

[63] S. Maheshwari, R. K. Gupta, P. Gupta, and A. Shukla, "Deep-learning approaches for health care: Patients in intensive care," 2020.

[64] K. Solez and L. Racusen, "The banff classification revisited.," *Kidney international*, vol. 83 2, pp. 201–6, 2013.

[65] K. Potdar, T. S. Pardawala, and C. D. Pai, "A comparative study of categorical variable encoding techniques for neural network classifiers," *International Journal of Computer Applications*, vol. 175, pp. 7–9, 2017.

[66] W. Chan, C. Saharia, G. E. Hinton, M. Norouzi, and N. Jaitly, "Imputer: Sequence modelling via imputation and dynamic programming," in *ICML*, 2020.

[67] I. T. Jolliffe, "Principal component analysis," in *International Encyclopedia of Statistical Science*, 2011.

[68] A. Koutsoukas, K. Monaghan, X. Li, and J. Huan, "Deep-learning: investigating deep neural networks hyper-parameters and comparison of performance to shallow methods for modeling bioactivity data," *Journal of Cheminformatics*, vol. 9, 2017.

[69] A. Agarwal, A. S. Chivukula, M. H. Bhuyan, T. Jan, B. Narayan, and M. Prasad, "Identification and classification of cyberbullying posts: A recurrent neural network approach using under-sampling and class weighting," in *ICONIP*, 2020.

[70] Z. Qiao, A. Bae, L. Glass, C. Xiao, and J. Sun, "Flannel (focal loss based neural network ensemble) for covid-19 detection," *Journal of the American Medical Informatics Association : JAMIA*, vol. 28, pp. 444 – 452, 2020.

[71] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *ArXiv*, vol. abs/1502.03167, 2015.

[72] F. Massicano, A. Sasso, H. Tomaz, M. Oleynik, C. Nobrega, and D. Patrão, "An ontology for tnm clinical stage inference," in *ONTOBRAS*, 2015.

[73] E. Shinohara, E. Aramaki, T. Imai, Y. Miura, M. Tonoike, T. Ohkuma, H. Masuichi, and K. Ohe, "An easily implemented method for abbreviation expansion for the medical domain in japanese text. a preliminary study.," *Methods of information in medicine*, vol. 52 1, pp. 51–61, 2013.

[74] P. J. O. Suárez, B. Sagot, and L. Romary, "Asynchronous pipeline for processing huge corpora on medium to low resource infrastructures," 2019.

[75] J. Tiedemann, "Parallel data, tools and interfaces in OPUS," in *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, (Istanbul, Turkey), pp. 2214–2218, European Language Resources Association (ELRA), May 2012.

[76] A. Adhikari, A. Ram, R. Tang, and J. Lin, "Docbert: Bert for document classification," *ArXiv*, vol. abs/1904.08398, 2019.

[77] C. Sun, X. Qiu, Y. Xu, and X. Huang, "How to fine-tune bert for text classification?," *ArXiv*, vol. abs/1905.05583, 2019.

[78] M. Kula, "Metadata embeddings for user and item cold-start recommendations," *ArXiv*, vol. abs/1507.08439, 2015.

[79] T. Kenter, A. Borisov, and M. Rijke, "Siamese cbow: Optimizing word embeddings for sentence representations," *ArXiv*, vol. abs/1606.04640, 2016.

[80] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, and J. Kang, "Biobert: a pre-trained biomedical language representation model for biomedical text mining," *Bioinformatics*, vol. 36, pp. 1234 – 1240, 2020.

[81] J. Závada, E. Hoste, R. Cartin-Ceba, P. Calzavacca, O. Gajic, G. Clermont, R. Bellomo, and J. Kellum, "A comparison of three methods to estimate baseline creatinine for rifle classification.," *Nephrology, dialysis, transplantation : official publication of the European Dialysis and Transplant Association - European Renal Association*, vol. 25 12, pp. 3911–8, 2010.

[82] J. Humble, C. Read, and D. North, "The deployment production line," *AGILE 2006 (AGILE'06)*, pp. 6 pp.–118, 2006.

[83] B. A. Juba and H. Le, "Precision-recall versus accuracy and the role of large data sets," in *AAAI*, 2019.

[84] A. Loupy, M. Haas, C. Roufosse, M. Naesens, B. Adam, M. Afrouzian, E. Akalin, N. Alachkar, S. Bagnasco, J. Becker, L. Cornell, M. C. C. van Groningen, A. Demetris, D. Dragun, J. D. van Huyen, A. Farris, A. Fogo, I. Gibson, D. Glotz, J. Gueguen, Kikić, N. Kozakowski, E. Kraus, C. Lefaucheur, H. Liapis, R. Mannon, R. Montgomery, B. Nankivell, V. Nickeleit, P. Nickerson, M. Rabant, L. Racusen, P. Randhawa, B. Robin, I. Rosales, R. Sapir-Pichhadze, C. Schinstock, D. Serón, H. K. Singh, R. N. Smith, M. Stegall, A. Zeevi, K. Solez, R. Colvin, and M. Mengel, "The banff 2019 kidney meeting report (i): Updates on and clarification of criteria for t cell– and antibody-mediated rejection," *American Journal of Transplantation*, vol. 20, pp. 2318 – 2331, 2020.

[85] M. Merchant, M. Brier, M. Slaughter, J. B. Klein, and K. Mcleish, "Biomarker enhanced risk prediction for development of aki after cardiac surgery," *BMC Nephrology*, vol. 19, 2018.

[86] E. Wallace, Y. Wang, S. Li, S. Singh, and M. Gardner, "Do nlp models know numbers? probing numeracy in embeddings," *ArXiv*, vol. abs/1909.07940, 2019.

[87] C.-Y. Lin and Y. Chen, "Acute kidney injury classification: Akin and rifle criteria in critical patients.," *World journal of critical care medicine*, vol. 1 2, pp. 40–5, 2012.

[88] I. M. Baytas, C. Xiao, X. Zhang, F. Wang, A. K. Jain, and J. Zhou, "Patient subtyping via time-aware lstm networks," *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017.

[89] R. Scheible, F. Thomczyk, P. Tippmann, V. Jaravine, and M. Boeker, "Gottbert: a pure german language model," *ArXiv*, vol. abs/2012.02110, 2020.

[90] X. Li, X. Sun, Y. Meng, J. Liang, F. Wu, and J. Li, "Dice loss for data-imbalanced nlp tasks," in *ACL*, 2020.

[91] K. Abhishek and G. Hamarneh, "Matthews correlation coefficient loss for deep convolutional networks: Application to skin lesion segmentation," *ArXiv*, vol. abs/2010.13454, 2020.

[92] M. Shorfuzzaman and M. Hossain, "Metacovid: A siamese neural network framework with contrastive loss for n-shot diagnosis of covid-19 patients," *Pattern Recognition*, vol. 113, pp. 107700 – 107700, 2020.

[93] H. Hazimeh, N. Ponomareva, P. Mol, Z. Tan, and R. Mazumder, "The tree ensemble layer: Differentiability meets conditional computation," *ArXiv*, vol. abs/2002.07772, 2020.

[94] I. Ali and P. Kalra, "A validation study of the 4-variable and 8-variable kidney failure risk equation in transplant recipients in the united kingdom," *BMC Nephrology*, vol. 22, 2021.

Appendix

Regular Expressions Variables

Street names	Telephone numbers
Zeile	Telefon
Weg	Tel
Strasse	Tel\
Straße	Tel\.
str\	Tel\.-Nr\.
str	Telefonnummer
Chaussee	Fax
Ring	Telefax
Damm	
Feld	
Platz	
Allee	

Physicians choice of features to predict graft Loss

Physician 1	Physician 2	Physician 3
Receiver Age	State of Donor	Primary Kidney Function
Years between first Dialysis and Transplant	Dialysis needed after Transplant	Dialysis needed after Transplant
Creatinine	Daily Protein	Donor Alive
Protein Concentration	Primary Kidney Function	Years between first Dialysis and Transplant
Banff	MM Board	Donor Degree of Kinship
Diagnoses Heath-Insufficiency	Banff	Age Receiver
Receiver Medication	Creatinine	MM Board
Type of Dialysis	Diagnosis Sepsis	Ischemia Cold of Kidney
State of Donor	CRP	Receiver Weight
Primary Kidney Function	Receiver Medication	Donor Age
Physician 4	**Physician 5**	
Banff	Donor Age	
Creatinine	Creatinine	
ProteinTUR	Daily Protein	
Daily Protein	Donor Alive	
Receiver Blood pressure systole	Donor Age	
Receiver Blood pressure diastolic	MM Board	
Receiver Medication	Banff	
Diagnosis Sepsis	CRP	
Diagnosis Hypovolemia	Diagnosis Myocardial Infarction	
MM Board	Diagnosis Sepsis	