Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Távközlési és Médiainformatikai Tanszék

Zsófia Császár

DATA MINING IN HEALTHCARE

Clinical decision support based on blood test data of heart transplant patients

SUPERVISORS

Dr. Alija Pašić and Péter Revisnyei

BUDAPEST, 2023

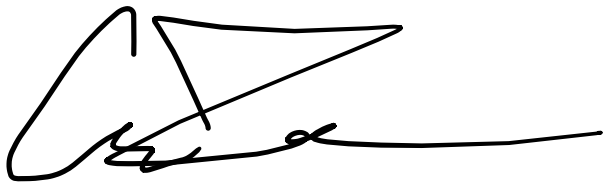
Table of contents

Hallgatói nyilatkozat

Alulírott **Császár Zsófia**, szigorló hallgató kijelentem, hogy ezt a szakdolgozatot meg nem engedett segítség nélkül, saját magam készítettem, csak a megadott forrásokat (szakirodalom, eszközök stb.) használtam fel. Minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Hozzájárulok, hogy a jelen munkám alapadatait (szerző(k), cím, angol és magyar nyelvű tartalmi kivonat, készítés éve, konzulens(ek) neve) a BME VIK nyilvánosan hozzáférhető elektronikus formában, a munka teljes szövegét pedig az egyetem belső hálózatán keresztül (vagy hitelesített felhasználók számára) közzétegye. Kijelentem, hogy a benyújtott munka és annak elektronikus verziója megegyezik. Dékáni engedéllyel titkosított diplomatervek esetén a dolgozat szövege csak 3 év eltelte után válik hozzáférhetővé.

Kelt: Budapest, 2023. 06. 09.



.....

Császár Zsófia

Kivonat

A modern technológiának köszönhetően, manapság már az egészségügyben is lehetséges hatalmas mennyiségű adatot generálni és gyűjteni. Mivel a gépi tanulási módszerek hatékonynak bizonyultak a komplex adatokban rejlő minták és kapcsolatok felismerésében, fontos szerepet játszhatnak a klinikai döntés-támogatásban is. A szívtranszplantációs műtét utáni túlélés esélyeit számos tényező befolyásolhatja, így fontos és hasznos lehet a betegek különböző adatai alapján az állapotukra vonatkozó előrejelzéseket tenni. A projekthez rendelkezésre álló orvosi adatszett a betegek különböző időpontokban vett artériás vérgáz paraméterinek méréseit tartalmazta, mind a szívtranszplantáció előtt és után. A mért paraméterek idősoros adatait először is meg kellett tisztítani és aggregálni, hogy különböző adatbányászati technikákkal elemezni lehessen. A főbb megállapítások szerint a paraméterek között kimutathatók lineáris és nem lineáris összefüggések, illetve a vérgáz mérések változásában is szignifikáns különbségek mutatkoznak a különböző túlélési idejű betegek között. Lineáris regresszió, döntési fa alapú, idősoros és neurális háló alapú modellek készültek különböző problémákra, hogy kiderüljön milyen típusú modelleket és kérdéseket érdemes tovább vizsgálni. A fő következtetés az, hogy a paraméterek előre jelezhetők egymásból többváltozós többrétegű perceptron (MLP) modellekkel, 0.0015-0.0025 közötti átlagos RMSE (root mean square error) és 1% alatti átlagos MAPE (mean absolute percentage error) mellett. A feature importance vizsgálat alapján, egyik paramétert sem szabad kizárni az előrejelzésből. Továbbá, a vérgáz paraméterek időben nagyobb távra is előre jelezhetők hasonlóan alacsony átlagos hibákkal. Teljesen ismeretlen betegek esetében, más betegek adatai alapján ugyan kisebb pontossággal, viszont anélkül lehet előrejelzéseket tenni, hogy a prediktált értékek tévesen esnének a normál referencia-intervallumba. A létrehozott előre jelző modellek csökkenthetik az ABG-vizsgálatok gyakoriságát, ezáltal a költségeket is. Az orvosok tájékozódhatnának a várható tendenciákról, valamint észlelhetnék az esetlegesen bekövetkező életveszélyes állapotokat is.

Abstract

Thanks to modern technology, it is possible to generate and collect large amounts of data in healthcare too. As machine learning methods proved to be efficient in recognizing patterns and connections in complex data, they can play an important role in supporting the clinical decision making. Many factors are influencing the postoperative survival of heart transplant patients, so basing predictions on different kinds of data about the patients' condition can be important and helpful. The clinical dataset given for this project contains measurements of arterial blood gas parameters taken at different times from patients before and after the heart transplantation. The time-series data was first cleaned and aggregated, then it was analyzed with different data mining techniques. According to the key findings, linear and non-linear relationships can be shown among the parameters, and significant differences in change of blood gas measurements between patients with different survival length. Linear regression, tree-based, time-series, and neural network baseline models were created for different problems to see, which type of models and questions are worth to examine further. The main conclusion is that parameters can be predicted from each other using multivariate multilayer perceptron (MLP) models with average RMSEs (root mean square error) between 0.0015-0.0025 and average MAPEs (mean absolute percentage error) under 1%. Based on the feature importance examination, none of the parameters should be excluded from prediction. In addition, it is possible to predict blood gas parameters further in time with similarly low average errors. Blood gas parameters can be predicted for completely unknown patients based only on other patients as well, although with less accuracy, but without making predictions that would falsely considered to be in the normal reference-interval. The resulting predictive models could reduce the frequency of ABG testing and this way its costs as well. Clinicians could gain insights to expected trends and detect possible life-threatening conditions too.

1 Introduction

As nowadays large amounts of data are being generated and collected in healthcare too, data mining opportunities continuously arise in the industry. Machine learning methods are widely used in healthcare, because they proved to be effective in understanding patterns and finding correlations from massive and complex data. The insights offered by the application of data mining can play an important role in supporting clinical decision making.

The first example of human heart transplantation happened more than 50 years ago, and nowadays heart transplantation is considered as the gold-standard treatment for patients who have end-stage heart failure. Although there has been significant progress made regarding this life-saving operation, the success of the transplantation is still severely restricted due to numerous factors [1]. Furthermore, till 2017 there was no generally accepted risk-prediction model for prognosis after heart transplantation [2], and after researching more recent studies still none was found. Because of all the factors influencing the posttransplant survival, it can be an important task to make predictions based on different type of data about the patients' condition, or even survival.

This report begins with a brief overview of the challenges in heart transplantation and arterial blood gas parameter prediction. After getting to know the biological background of the project, the goals of data mining along with its process are discussed. Since data mining uses many statistical and machine learning techniques, these are also described, putting the emphasis on regression and neural network models. After understanding the technologies related to the project, some related studies about similar tasks or technologies are introduced. The description of the performed work starts with the introduction of the dataset and continues with the cleaning and preparation of data. Then the main discoveries of the Exploratory Data Analysis and the application of the algorithms are presented. After explaining the process of modelling and the created different baseline models, the performance of some models is evaluated and compared. Based on this, the best models or more important questions are further investigated with a new evaluation approach, different training, hyper-parameter optimization and feature importance examination. Finally, the results are examined to draw conclusions and

identify opportunities for further improvement. The report ends with summarizing the main results, the additional conclusions, and possible further directions of the project.

2 Heart transplantation and blood gas analysis

2.1 Challenges in human heart transplantation

More than 50 years have passed since Dr. Christiaan Barnard performed the first human-to-human heart transplantation in South Africa. The initial optimism around heart transplantation (HTx) quickly disappeared, when it turned out that the survival usually only lasted for days or weeks. Fortunately, during the next two decades the survival period has significantly improved, thanks to applying more carefulness in donor and recipient selection, better donor heart management and the use of cyclosporine as the main agent for immunosuppression [1].

By 2014, the one-year survival after HTx was around 90%, which is a great achievement compared to 30% in the 1970s. However, the long-term outcomes have not changed much, and there are many serious challenges in the field. One reason of new challenges the transplant clinicians are facing is the changing demographics of heart recipients. A greater part of patients in their sixties and seventies are being transplanted, who have higher risks of infection and cardiac allograft vasculopathy. The advances in heart surgery also led to younger patients to survive growing up with congenital heart disease and develop heart failure later in their life. These patients usually have higher risks of arterial bleeding and mortality [3]. Other challenges of HTx include the harmful effects of immunosuppression, which aims at preventing or treating the rejection while at the same time minimizing the risk of infection or cancer. In fact, the success of HTx has been closely related to the discovery of effective immunosuppressive treatments [1]. There are still many unanswered questions regarding immunosuppression, not to mention chronic rejection, antibody-mediated rejection or malignancy [3].

2.2 Arterial blood gas test

The measurements of blood gas along with other monitoring techniques provide information to the clinician is crucial in assessing patients, therapeutic decision making and prognostication [4].

Arterial blood gas (ABG) tests are blood tests performed by using blood from the artery. An ABG test is used to assess gas exchange in patients with respiratory disorders, to acquire patients' acid-base status, and it is one of the most commonly performed tests

in intensive care units (ICUs). Furthermore, it has numerous applications in other medicine related areas as well. The ABG test reports the pH of the blood, the partial pressure of carbon dioxide and oxygen, the bicarbonate level and many analysers also include concentrations of lactate, haemoglobin, several electrolytes, oxyhaemoglobin, carboxyhaemoglobin, and methaemoglobin [5]. The ABG test is not only expensive but also stressful for the subject to carry out, thus the frequency of testing should be reduced by relying on previous results.

2.3 Blood gas parameter prediction

Prediction of future values for blood gas parameters would lead to better planning regarding treatment. Besides, having information about expected trends, the clinicians might be able to prevent life-threatening changes in values as well. Unfortunately, the prediction of blood gas parameters is usually a very difficult and complex task.

The complexity can arise from the sudden changes in measured values, especially regarding new-borns [4]. Another issue is that every patient has their own personal dynamics of biochemical processes in the arterial blood, which can be changing during a healing process [5].

However, there is great need for precise and rapid predictions in the area. The limited resources of ICUs need efficient management, especially when external stressors, like a pandemic increases patient numbers [6]. Laboratory testing occurs frequently for patients in intensive care, and part of the tests are only run by default without reflecting changes about the critical status of ICU patients. Using blood test excessively also increases resource utilization, contributes to blood loss, can lead to incorrect diagnosis [7]. ABG tests are globally standardized in ICUs and obtained relatively frequently as well, thus ABG test parameters can be used to develop predictive tools on. Machine learning can be used for the prediction making and this way also in optimizing the allocation of resources. In addition, machine learning methods which inherently integrate a large amount of data, can also play an important role in supporting clinical decision making [6].

3 Data mining

Nowadays huge amounts of data are collected from almost every aspect of our lives daily. The medical and health industry is not an exception either, it can generate enormous amounts of data as well, for example from medical records. The need to gain valuable information from this vast amount of data led to the birth of data mining [8].

3.1 Process of data mining

The definition of data mining is discovering interesting and useful patterns and relationships in data. The goal of data mining can vary, for example it can be used to generate insightful and understandable reports to end users [10]. There are two types of goals in general: in verification, the system is used to verify the user's hypothesis, while in discovery, the system is used to find new patterns. Discovery can be further divided into two categories, prediction, and description. Prediction means finding patterns in order to predict future behaviour of certain entities, while description means finding patterns for presenting them to users in a form, they can understand [9].

In terms of the CRISP-DM (CRoss Industry Standard Process for Data Mining) project, a process model was defined providing a framework for data mining projects, so the projects would not depend highly on a particular person or team, as before. The CRISP-DM process model can be used in any industry and with any technology to make the data mining project less expensive, more reliable and faster as well. The CRISP-DM reference model for data mining consists of six phases. The process begins with defining a data mining problem and designing a preliminary project plan. After that, initial data is collected, data quality problems and first insights are identified. Understanding initial data is also necessary for business understanding, so the first to phases are strongly connected. The third phase is about creating the final dataset for the model from raw data. Among other tasks, data preparation includes attribute selection, creating new attributes and cleaning the data. There is a strong link with the next phase because data problems or need for constructing new data can also be identified during modelling. In the modelling phase, different techniques are applied and parameterized. After building one or more seemingly optimal model, they are evaluated and the steps of constructing them are reviewed. The purpose of the evaluation phase is to make sure every important business issue has been considered and to decide on the use of the data mining results. In

the final phase, depending on the requirements, for example the results can be presented in a report to the end user, or a repeatable data mining process might be implemented [11].

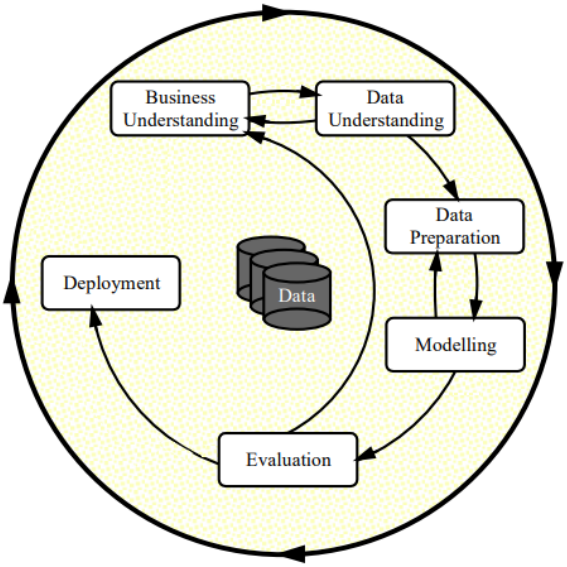


Figure 1: Phases of CRISP-DM Model for Data Mining [10]

3.2 Statistics

3.2.1 Statistics in data mining

Data mining integrates many techniques from statistics. According to Han et al., statistics studies the collection, analysis, interpretation or explanation, and presentation of data. Statistical models can either be the outcome of a data mining task or a data mining task can be built on them as well [8].

Basic statistical descriptions, such as measures of central tendency (mean, median and mode) and dispersion measures (range, quartiles, interquartile range, variance, and standard deviation), can summarize and give an overall picture of data [8]. Visualization tools like histograms, box plots or scatter plots are also useful for understanding the structure of data. Another widely used statistical analysis technique is cluster analysis, which aims at creating internally homogeneous and externally heterogenous clusters by organizing information about variables. How changes in one variable result in changes in another, can be measured with correlation analysis. The correlation coefficient can be useful to understand the predictive abilities of an independent variable. Furthermore, with regression analysis relationships between a dependent variable and one or more independent variable can be estimated. Some other

popular techniques include discriminant analysis, factor analysis and other types of regression analysis, for example logistic regression [12]. However, applying statistical methods on large data sets is often challenging, as many methods have high computational complexity and cost, so algorithms must be designed and tuned carefully [8].

3.2.2 Time series analysis

An important area of statistics includes methods for analyzing and modelling time series. A time series consists of observations, that are made sequentially in time. Examples to time series exist in numerous fields for example in economics, physical sciences, or engineering. Time series analysis can have different purposes, like obtaining descriptive measures, explaining properties of one time series based on another, statistical quality control or predicting future values.

The traditional time-series analysis methods focus on the decomposition of the variation in the series. According to Chatfield, the variation can be decomposed into four different kinds of components. First, **seasonal effect** is a periodically reoccurring variation, that is easily understandable. Besides seasonal effects, there can be **other cyclic changes** that are present at a fixed period, like the daily variation in temperature. Another component is the **trend**, which is a long-term change in the level of the mean. The definition of "long-term" here must depend on the number of observations. After removing cyclic variations and trend form the time-series, a series of **other irregular fluctuations** remain. Some of these irregular variations might be explained with probability models, like moving average or autoregressive models [13].

The Auto Regression (AR) model calculates the regression of past time series and present or future values in the series, while the Moving Average (MA) model calculates the errors of past time series instead of the regression. Combinations of AR and MA models also exist, where the effect of previous time series and errors are also taken into account for forecasting the future values [14]. Forecasting of the time-series can be univariate, which means the forecasts of a variable are based on its past observations, while in multivariate forecasting the variable depends (at least partly) on values of one or more other series. To apply any variations of ARMA models, the time-series needs to be stationarity. Intuitively, a time-series is considered to be stationary if there is no

systematic change in the mean or variance, and no strictly periodic variations are present [13].

Overall, statistical methods are applied in data mining for various reasons. Statistics not only helps to understand the data, but also to discover patterns and understand the underlying reasons affecting them. In addition, statistics plays a major role in developing and evaluating models, so using it in data mining is basically inevitable.

3.3 Applied machine learning algorithms

3.3.1 Machine learning

"Machine learning investigates how computers can learn (or improve their performance) based on data" [8]. Furthermore, the purpose of machine learning is to automate time-consuming human activities in the knowledge engineering process with techniques, that can identify regularities in training data [12]. Nowadays a huge variety of applications take advantage of machine learning: web page ranking, collaborative filtering, automatic translation and face recognition, to name a few. Just like the range of applications, the range of machine learning problems is wide as well [15].

Han et al. collected some classic problems in machine learning, that are strongly connected to data mining [8]:

- **Supervised learning** has two main categories, **classification** and **regression.** In order to supervise the learning of the model, labeled examples are used for training the classification model and continuous numerical values for the regression model.
- **Unsupervised learning**, also known as **clustering,** is typically used to discover classes in the data. The learning is unsupervised because the training data is not labeled.
- **Semi-supervised learning** uses labeled and unlabeled examples as well to train the model.
- **Active learning** aims at optimizing the model quality by letting users participate in the learning process to gain knowledge from them.

In the terms of this project, the machine learning problem was a supervised learning problem. The time-series data was used to predict future values and understand

the relationships between blood gas parameters. Different kinds of machine learning models were tried including regression models, tree-based models, time-series models, and neural networks as well.

3.3.2 Linear regression models

Regression models are suitable for approximation and the simplest model is based on linear regression. In linear regression the data is fitted on a straight line. The response variable (y) can be described as a linear function of a predictor variable (x), with an equation $y = wx + b$, where w means the slope of the line and b the y-intercept. These regression coefficients can be determined with the least squares method, that minimizes the error between the real line separating the data and the estimated line.

An extension of linear regression is multiple linear regression, where y can be modelled as a linear function of more than one predictor variables [8]. In cases of multiple-regression models where the independent variables are highly correlated, using ridge regression is advised in order to reduce the effects of correlation and stabilize the regression coefficients [16].

In this project, different kinds of regression models were tried. The *LinearRegression* model fits a linear model by minimizing the residual sum of squares between observed and predicted targets. The *BayesianRidge* model is based on Bayesian Ridge Regression, which is a type of Bayesian regression. Bayesian regression creates linear regression by using probability distributors instead of point estimates with the response variable assumed to come from a probability distribution. *BayesianRidge* model iteratively maximizes the marginal log-likelihood for the data points [18].

3.3.3 Tree-based regression models

Besides classification problems, decision trees can be used for regression tasks as well. A decision tree has a flowchart-like tree structure with internal nodes that contain tests on an attribute, branches that shows outcomes of the test, and leaf nodes containing class labels [8].

The regression tree algorithm works iteratively by splitting the dataset and averaging the original target values to create predictions on both sides of the split. Then the chosen metric is calculated from the original and predicted outputs. Having *n* values for predictor and output variables, when *n-1* metrics are calculated, a choice is made to

split the dataset where the error metric is the lowest. With the selected split, the other data points go to one of the nodes and the process is repeatedly done on both sides creating a tree-like structure [19]. The *ExtraTreesRegressor*, which is used in the project, operates by fitting numerous randomized decision trees on different sub-samples of the dataset and averages the predictions to be more accurate and avoid over-fitting [18]

3.3.4 Neural networks

The creation of neural networks comes from the idea, that the human brain computes in a completely different way than a digital computer does. The human brain can organize its neurons, the structural constituents of the brain, to perform computations like perception or pattern recognition.

Basically, an artificial neural network (ANN) is designed to model the way the brain performs a task. The neuron of a neural network is an information-processing unit that is the basis of the neural network. On Figure 2, the model of a neuron and its main elements are presented.

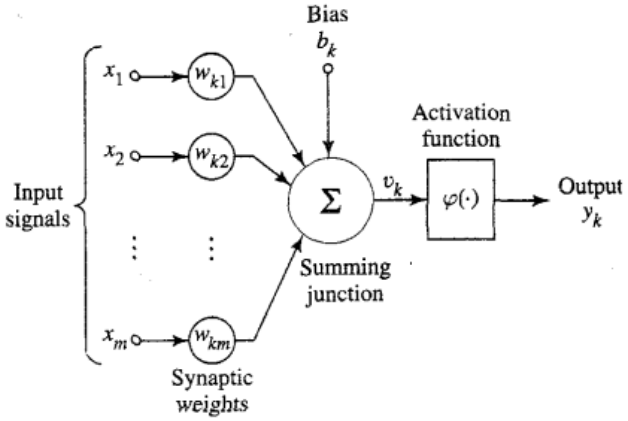


Figure 2: Nonlinear model of a neuron [20]

The set of synapses (connecting links) are characterized by their weights, with that input signals of the synapses are multiplied by. A summing junction is responsible for summing the weighted input signals and the activation function limits the output signal's amplitude range to a finite value. Three basic types of activation functions are identified: threshold, piecewise-linear and sigmoid functions [20]. Sigmoid function, which has an s-shaped graph was the most used activation function, however by now another function was discovered that usually performs better and is easier to train, the Rectified Linear Units (*ReLU*). *ReLU* is a piecewise-linear function which outputs the input itself if it is positive, and zero if it is not. It has become the default activation

function for many kinds of neural networks due to its fast training and good results [22]. The model on the figure also includes external bias, that increases the net input of the activation function if it is positive and decreases in case it is negative.

The perceptron, consisting of a single neuron with synaptic weights and bias that can be adjusted, is the simplest type of a neural network. This perceptron with a single neuron is limited to classification tasks between only two classes. However, neurons can be organized in layers as well. The simplest layered neural network consists of an input layer and an output layer only. This single-layer network is considered to be feedforward, as the projection only happens in one direction, from the input layer to the output layer. Feedforward neural networks can have one or more hidden layers as well, where the nodes are called hidden neurons [20]. The model of a multilayer feedforward neural network on Figure 3 shows the three different layers. The input layer transfers data received from the network to the connected neurons in the hidden layer. The data is processed in the hidden layer and then transferred to the output layer, which provides an output based on the analysis of the received data [22].

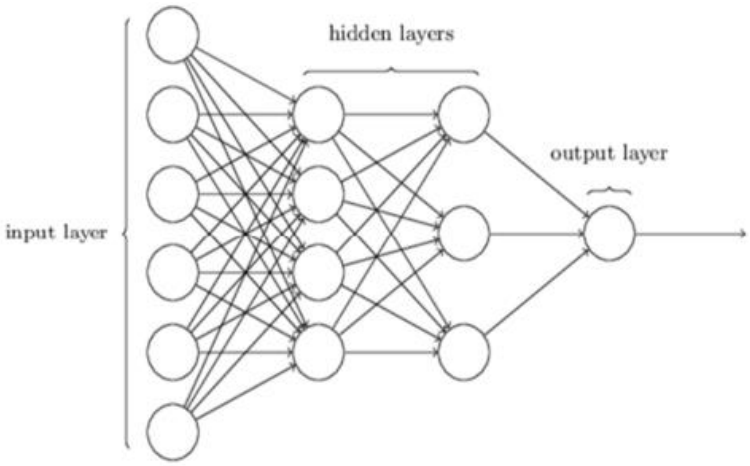


Figure 3: Model of a multilayer neural network [22]

Types of neural networks that are not feedforward but have feedback loops are called recurrent neural networks. A feedback loop in a single layer neural network works in a way that each neuron feeds back its output signal to the inputs of all other neurons. It can also have self-feedback loops where the output of a neuron is also fed back to its own input. Feedback loops can have a great effect on the network's learning abilities and performance as well.

In this project, multilayer perceptrons (MLP) are used for modelling. In this type of neural network each neuron includes a nonlinear activation function, the network has one or more hidden layers and also high degrees of connectivity. MLPs has been successful in solving various kinds of difficult problems due to training them with the well-known back-propagation algorithm. This algorithm consists of a forward and a backward pass. In the forward pass phase, the network processes the input by activating the neurons and produces an output value while the synaptic weights stay fixed. During the backward pass, the synaptic weights are adjusted based on the calculated error that is propagated back through, layer by layer [20]. This process is repeated for each input data in the given training dataset. One round of passing the entire dataset is called an epoch in the *Keras* library [21] used in this project.

Using neural networks has several advantages, some of the main benefits Haykin [20] defined are the following:

- **Nonlinearity:** It is especially important when the generator of the input signal is also nonlinear, for example a speech signal.
- **Input-output mapping:** It is created by the network to be able to perform supervised learning and learn from the given examples.
- **Adaptivity:** Neural networks are able to adapt their synaptic weights according to changes in the environment, so they can be retrained easily when a minor change happens.
- **Evidential response:** Neural networks can provide information about the made decision in pattern classification problems, so this way the unsure patterns can be rejected to improve performance.
- **VLSI implementability:** Because of its massive parallelism, a neural network can compute certain tasks fast. This way it is appropriate for implementing very-large-scale-integrated (VLSI) technology, which can capture complex behaviors.

However**,** neural networks have some limitations as well, for example, they require training, and a large neural network needs a lot of processing time [22]. Another problem with neural networks is that they are an example of the black-box approach, where the model is selected in a mechanistic way and there is little understanding about the

underlying mechanism. Because of this, ‘black boxes’ and also neural networks might not always give sufficient results [13].

3.4 Training and optimization

The dataset was split into train and test set with different approaches in the project. One approach was to split in time and use for example first 60% of data points in time for training, while remaining 40% for testing. Another approach was to split patients and use some percentage of them for training, then test the model on unseen patients.

Other than simple train-test splitting, the K-fold cross validation method was used for training in the project. The main idea of the cross-validation is the hold out method, meaning the available set of N examples is divided to K subsets (K>1) and the model is trained on all subsets except for one. This remaining subset is used to measure the validation error on, and the process is repeated K times, every time using a different subset for validation [20].

Building an optimal machine learning model can be a complex and time-consuming process. A key component of this process is to design an ideal model architecture by optimal hyperparameter configuration. There are two types of parameters in machine learning models: model parameters, which can be initialized and updated during the learning process and hyperparameters, which cannot be estimated from the learning process. Hyperparameters must be set before training because they are used to configure the model, or to specify the algorithm for minimizing the loss function. There are different types, hyperparameters can be categorical, discrete, or continuous.

However, manual tuning might be ineffective in some cases, for example if the model evaluation is time consuming or there is a large number of hyperparameters [23]. Fortunately, automated hyperparameter optimization can reduce the required human effort in machine learning, improve the algorithms’ performance and is also more reproducible than manual search [24]. The process consists of four components: a regressor or a classifier with its objective function, a search space, a search or optimization method used for finding hyperparameter combinations, and an evaluation function for comparing the performance of different configurations. The main goal for hyperparameter optimization (HPO) is to enable users to apply machine learning models effectively by automating the hyperparameter tuning process [23].

Grid search was used here, which is a basic HPO method that performs an exhaustive search on the hyperparameters given by the user, so the user must have preliminary knowledge of these. This method is widely used because of its mathematical simplicity, and it can run in parallel because results of one trial are independent from other trial results. However, the consumption of computational resources grows exponentially when more hyperparameters need to be tuned simultaneously [25].

3.5 Evaluation metrics

When predicting continuous variables, a measure is needed which can tell how close the predictions are from the actual values. Mean Square Error (MSE) can be used for this purpose, which is calculated with the formula,

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

where Y is the actual value and $\hat{Y}$ is the predicted value [19] . In the project the Root Mean Square Error (RMSE) is used for evaluating the prediction accuracy, which is calculated by taking the root of MSE, as in the formula:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2}$$

In the baseline model evaluation only RMSE is used, but later, when not only accuracy but the correctness of the predictions is examined as well, the Mean Absolute Percentage Error (MAPE) is considered to give better insights. MAPE is calculated as the following formula, where m is the number of predicted samples, and the values are the original measurements [4]:

$$Error = \frac{\sum_{j=1}^{m}\frac{abs(true\_value_j - predicted\_value_j)}{true\_value_j}}{m} * 100$$

The MAPE shows the mean of relative absolute differences in a predicted sample, which gives an estimate of the range in which on average the model predicts above or below the actual measurement value.

3.6 Technology

For the data mining project, *Python* [26] programming language was used because of the rich set of libraries it offers. *Jupyter Notebook* [27] provided and interactive environment to extract insights of the data.

Pandas [28] , *NumPy* [29] and *SciPy* [30] libraries were used for data manipulation, conversion, and calculations, while *Matplotlib* [31] and *Seaborn* [32] for visualization. *PyCaret* [33] was used for automated machine learning which helped in model selection. The final machine learning models were created with *Scikit-learn* [34], *Keras* [21], *pmdarima* [35] and *pyclustertend* [36].

4 Related work

Machine learning methods have been widely applied in clinical diagnosis and prognosis prediction, as they proved to be advantageous in finding inherent correlations and understanding patterns of massive and complex data. As nowadays HTx is still considered as the gold-standard treatment for patients with end-stage heart failure, and the decision about transplant candidacy and donor organ allocation is also influenced by the post-transplant survival. Therefore, the prediction of the recipient's survival is a very important issue. However, there is no risk-prediction model for assessing prognosis after HTx, that is accepted generally and has high-accuracy [26].

Zhou et al. [26] made an attempt to develop a 1-year survival prediction model of HTx, that can help in clinical decision-making as well as in optimization of organ allocation strategies. Their best performing model was a Random Forest, and they have found the albumin, the age and left atrium diameter as the most important variables affecting 1-year mortality of HTx. They also reached the conclusion, that machine learning methods are most resist to overfitting, compared to traditional regression analysis. Medved et al. [2] found, that a deep learning based risk prediction model has greater accuracy for the prediction of HTx outcomes, than a traditional logistic-regression based model.

Besides the survival, other conditions can be examined regarding HTx. For example, Mohacsi et al. [38] investigated lactic acidosis following HTx by performing ABG analysis, however, they only used statistical methods. They found no correlation between lactic acidosis and blood gas analysis during the examined extracorporeal perfusion period. Braith et al. [39] also used statistical analysis to examine ABG parameters in order to draw conclusions about the development of cardiodynamic hyperpnea in heart transplant recipients.

Others also examined ABG parameters with using machine learning approaches, for different purposes. Wajs et al. [5] focused on optimizing the forecast of ABG parameters. The used Multilayer Artificial Neural Networks on time-series data of extremely premature infants. They found that it is very difficult to build a proper model based on the historical data due to the patients' changing personal dynamics and biochemical processes. Thus, they used a model working in real time loop, meaning it

was retrained in every time step, using data only from a certain interval. In another study, Wajs et al. [4] examined ABG parameters in new-borns again, and reached the conclusion, that it is possible to successfully predict ABG value by predicting single points iteratively, instead of predicting an entire time series immediately. The ANN they used, predicted only result in every step and reached an average error below 1%. Wernly et al. [6] researched, how mortality in septic patients can be predicted based on ABG parameters. They used a type of Deep Neural Networks, using long short-term memory (LSTM) to learn dependencies between ABG parameters. According to their results, LSTM-based models can help ICU physicians by predicting mortality with high accuracy.

5 Blood test data mining

In this section, the available dataset consisting of the arterial blood gas measurements of heart transplant patient is first cleaned and prepared for analysis. Then the pre-processed dataset is explored to understand the behavior of parameters and differences among patients. After that, models are created for different problems and their performance is evaluated. In the end, the results of blood test data mining are summarized.

5.1 Data pre-processing

5.1.1 About the dataset

The dataset used for the project contains the blood test results of patients who went through heart transplantation. The meaning of the attributes in the dataset are explained in Table 1.

Attribute	Description
pt_id	Unique patient ID
event	1 = patient died, 0 = patient survived
event_gr	Length of time the patient survived after surgery. Values: >5é+, >2é+, <90n+, <1é+, <7n+, >10é+, >1é+, <30n+, surv
min	Time of measurement in minutes before or after surgery
pH	Acidity of the blood (%)
PO2	Partial pressure of oxygen (mmHg)
PCO2	Partial pressure of carbon dioxide (mmHg)
Hct	Hematocrit - a measurement of the volume percentage of red blood cells (%)
Na+	Natrium concentration (mmol/l)
Cl-	Chloride concentration (mmol/l)
tHb	Total hemoglobin concentration (g/dL)
Glu	Glucose concentration (mmol/l)
Lac	Lactate concentration (mmol/l)
cHCO3-	Hydrogen carbonate concentration (mmol/l)
BE	Base excess - amount of HCO3- (mEq/L)

Table 1: Description of attributes in the dataset

The index of the created data frame is the *pt_id* column, which identifies 5057 different patients. The data is considered to be time series data, as the measurements were made sequentially in time, with the *min* attribute marking the time points. Besides this initial dataset, the reference value ranges of the different blood gas parameters were also collected.

5.1.2 Data cleaning and preparation

The data preparation phase of the project includes activities that contribute to creating the final dataset for modelling from the initial data, such as transforming and cleaning the data. Cleaning data is necessary to handle missing data, empty values or incomplete data [8].

As the table contained several empty values, the first step of cleaning was dropping the rows where all blood gas parameters values were missing. This way the initial 15628 rows in the dataset decreased to 9445. Out of the remaining 1535 unique patients, many patients had only few measurements (rows), as presented on Figure 4. Data of patients with measurements at only one or few times can not be used as time series data, so the data was filtered to those who have at least 10 measurements, leaving 293 patients.

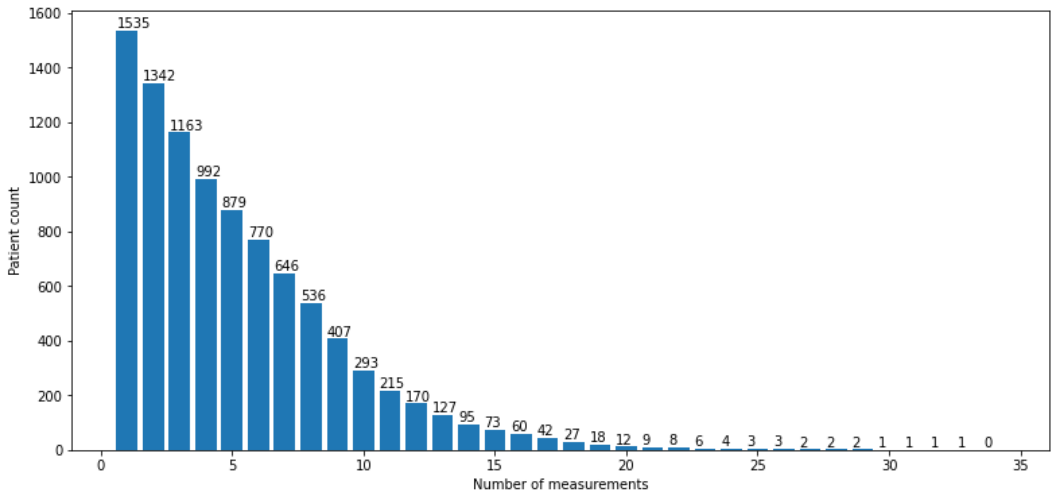


Figure 4: Count of patients regarding the number of measurements

Patients had the measurements at different times, so a common time interval needed to be defined for further analysis. The average value of the earliest measurement for patients was around 6 minutes before surgery, while the average of the latest measurement was around 573 minutes after surgery. According to this, the data was further filtered to those who had measurements between -10 and 600 minutes, leaving a final number of 94 patients in the dataset.

For modelling purposes, the data has been transformed to have values for every patient and in every minute for a certain time interval (after the exploratory data analysis). There were some cases where more measurements have been recorded in few minutes, so the data was first transformed to contain the averages of measurements that were recorded

less than 5 minutes after each other. After that, the interpolation was done by using *Akima1DInterpolator* [40] from the *SciPy* package [30]. With the *Akima* interpolation, a curve can be created that passes through the given points smoothly. The slope of the curve is determined locally at each point, using the next neighboring points to determine coefficients for the interpolation polynomial [41]. As the *Cl-* parameter had missing values at many time points and only one measurement for several patients, it could not be interpolated and were not used for modelling. From the created slope 300 data points were sampled equally, meaning one sampled data point (time step) has a length of around 2 minutes. The final dataset for baseline modelling contained 94 patients' interpolated data of 10 blood gas parameter for a 300 time step long interval.

5.2 Exploratory data analysis

Exploratory data analysis (EDA) was defined by Behrens as "a well-established statistical tradition that provides conceptual and computational tools for discovering patterns to foster hypothesis development and refinement" [42].

To begin with, the survival of patients was examined. As presented on Figure 5, the number of patients that died after the transplantation is less than half of patients who survived.

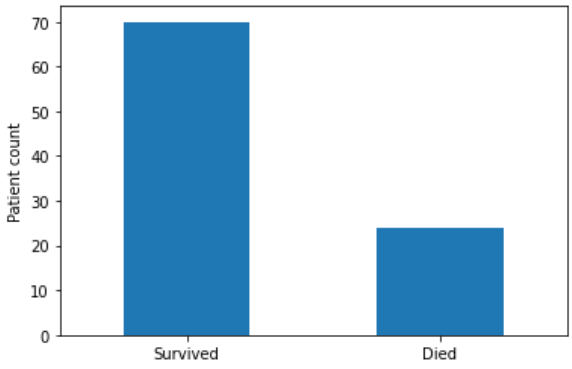


Figure 5: Number of surviving and dead patients

The change in time of blood gas parameters were compared between the two groups with statistical significance test. The compared samples were created for each blood gas parameter by rounding the minutes to nearest tens and taking the mean values of these rounded times. Non-parametric tests were used as the created samples did not follow a normal distribution. According to the Mann-Whitney test's results presented on Table 2, at a 0.05 significance level the null hypothesis can be rejected at all parameters except

Glu and *Cl-*, where the higher p-values suggest that the samples were drawn from the same distributions.

Param	Stat	p	Result
BE	3686.0	2.539238e-10	Different distribution (reject H0)
cHCO3-	3681.5	2.887069e-10	Different distribution (reject H0)
Hct	3317.5	2.610381e-06	Different distribution (reject H0)
Lac	1310.5	3.353643e-05	Different distribution (reject H0)
tHb	3126.5	1.147768e-04	Different distribution (reject H0)
PCO2	3045.0	4.696700e-04	Different distribution (reject H0)
pH	3011.5	8.089519e-04	Different distribution (reject H0)
Na+	1775.5	3.597902e-02	Different distribution (reject H0)
PO2	1794.0	4.384609e-02	Different distribution (reject H0)
Glu	1941.5	1.722447e-01	Same distribution (fail to reject H0)
Cl-	2283.5	8.861083e-01	Same distribution (fail to reject H0)

Table 2: Mann-Whitney tests results on comparing parameter changes between survived and dead patients

Figure 6 shows the change of mean values over time in the two groups for the parameters with a significant (*BE*) and non-significant (*Cl-*) Mann-Whitney result.

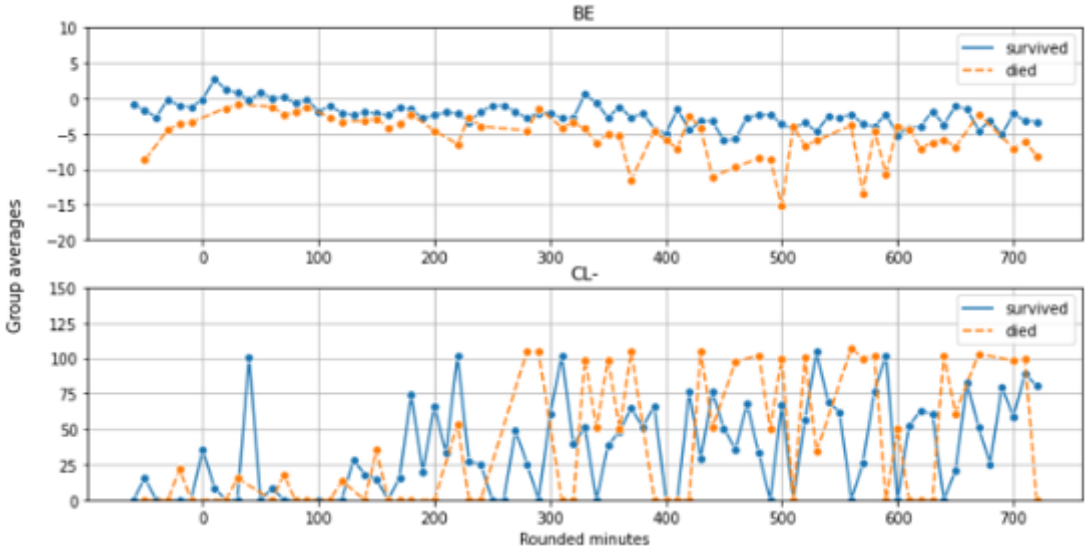


Figure 6: Change of BE and CL- in groups of survived and dead patients (y-axis: group averages at rounded minutes)

The measurements in the two groups were also compared by taking the mean values of all parameters at different points in time as well. According to the Mann-Whitney test's results, with a 0.05 significance level there is not enough evidence to reject the null hypothesis at any point of time. On Figure 7, the differences between mean values of the different parameters are plotted for each examined time point. The plotted values were

calculated by taking the average of survived and dead patient groups for each parameter at the rounded time points and calculating their difference. Even though this plot shows big differences at some minutes, p-values of the statistical test suggest that the samples does not differ significantly at any time.

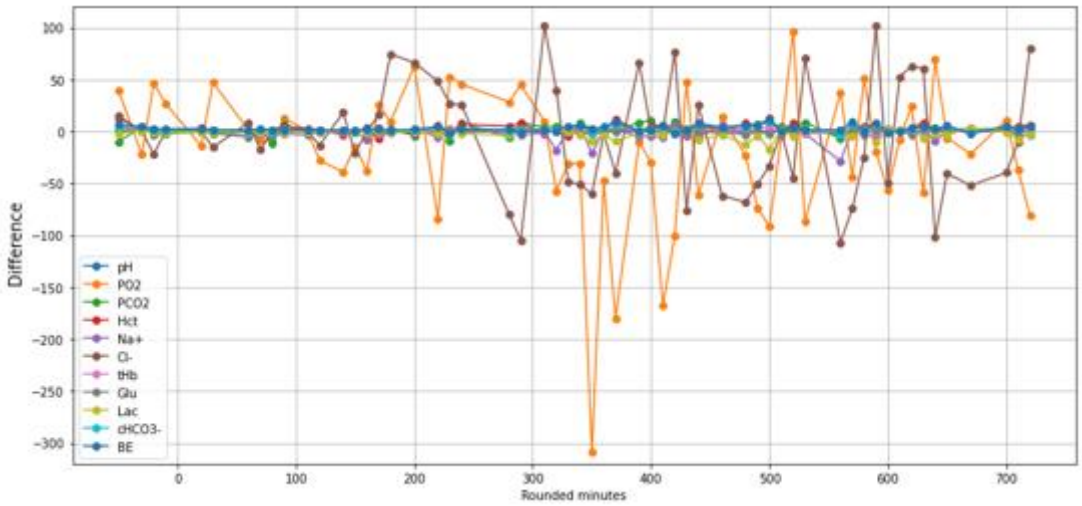


Figure 7: Differences of mean values between survived and dead patient groups (y-axis: difference of group averages at rounded minutes)

The group of patients who passed away was further examined, regarding the length of time patients survived after the transplantation. Figure 8 shows the distribution of these patients among the different time categories.

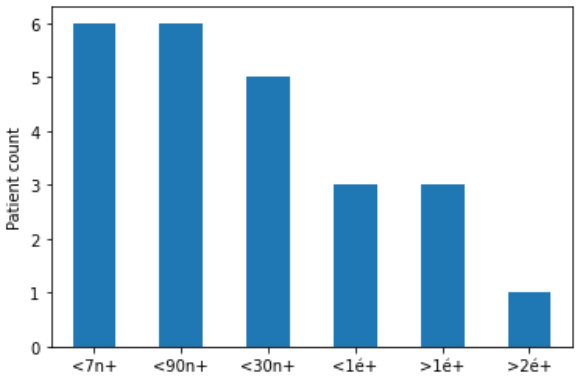


Figure 8: Distribution of dead patients regarding the time of survival after surgery

To check whether the change of blood gas parameter values differ significantly between the different groups, again a non-parametric test, the Kruskal-Wallis test was conducted. Like before, the six compared sample was created by rounding the minutes to nearest tens and using the mean values of parameters at the rounded minutes. The results show that except for the *Glu* and *Na+* parameters, the null hypothesis can be rejected at all other

parameters meaning the samples created from group averages differ significantly. The lowest p-values appear at the *tHb*, *Hct* and *pH* parameters.

The blood gas parameters measurements were further explored in terms of reference intervals, since values outside reference intervals can be dangerous. Investigating further the differences between the patients who survived and those who passed away, the percentage of patients with values out of reference interval for each parameter was compared over time. According to the Mann-Whitney significance test's result, the null hypothesis can be rejected and the samples differ significantly for 6 parameters. The changes in the number of patients outside the reference intervals over time for the parameter with the lowest (*BE*) and highest (*PCO2)* p-value are plotted on Figure 9.

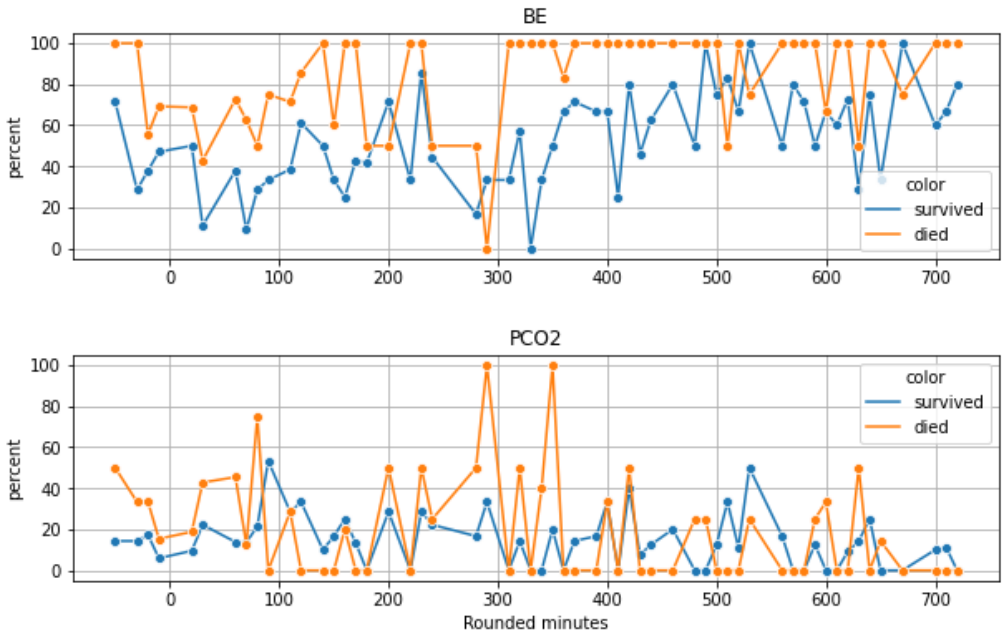


Figure 9: Percentage of patients outside reference intervals of BE and PCO2 over time

For gaining information about the connection between the different blood gas parameters, Spearman correlation analysis was conducted with the *spearmanr* function from *SciPy* package. The correlation of each parameter combination was checked individually for every patient. The results were filtered to only include significantly correlating combinations using a 0.05 significance level. Then the positively and negatively correlating parameter pairs were separated and ranked by the Spearman correlation coefficient and p-value for each patient. Finally, the individual results were summarized to check how often a pair of parameters has the strongest positive or negative correlation

among patients. The results on Table 3 show how many times a pair is in the strongest 3 combinations for all patients regarding positive or negative correlation. Looking at the positive correlations, the combination of *cHCO3-* and *BE* was one in the three pairs having the strongest correlation for 77 patients. The pair of *Hct* and *tHb* was among the three strongest correlating pairs for almost the same number of patients as well. For the negative correlation, *pH* and *PCO2* was one of the three strongest correlating pairs for far more patients than any other combination. Based on this table, these pairs mentioned have the strongest positive/negative correlation in general.

To discover if clusters exist in the data, Principal Component Analysis (PCA) was applied on the interpolated values. PCA is an old technique used for reducing dimensionality in a dataset that consists of many correlated variables. The main idea of PCA is to achieve this reduction by transforming to an uncorrelated and ordered set of variables, the principal components (PC), while keeping the highest possible amount from

Positive correlation	Num of patients
cHCO3- & BE	77
Hct & tHb	76
PCO2 & cHCO3-	26
Glu & Lac	11
Na+ & Glu	5

Negative correlation	Num of patients
pH & PCO2	53
Hct & Lac	17
tHb & BE	13
Hct & BE	12
tHb & Lac	12

Table 3: Results of correlation analysis among parameters

the variation of the dataset [43]. The Hopkins test was used for evaluation, which can be helpful in deciding whether the data follows a uniform distribution, or it has clustering tendencies. The Hopkins score from *pyclustertend* package [36] close to 0 indicates that the data is not uniformly distributed and might have existing clusters, but a higher score around 0.3 means the data does not have clustering tendencies [36].

PCA was applied on data from different time intervals, but the lowest Hopkins score was achieved by using an interval for each parameter in which the standard deviation was highest. The two PCs used together covered 55% of explained variance, with the *BE* having the highest loading score in the first and *PCO2* in the second

component. The result is plotted on Figure 10, from which clustering tendencies can be seen, however no clear clusters can be defined.

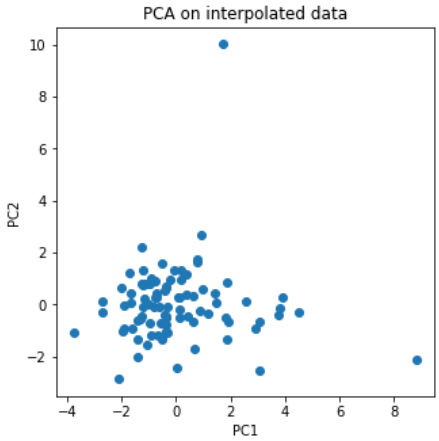


Figure 10: Result of PCA on interpolated data

Furthermore, Functional Principal Component Analysis (FPCA) was tried on each parameter individually as well, implemented with the *fdasrsf* package. FPCA is useful when keeping the patterns in the time-series data is more important than keeping the absolute variance, as it determines the corresponding functions for underlying patterns [44]. The results (part of them plotted on Figure 11) indicate that none of the parameters can be used for defining clear clusters.

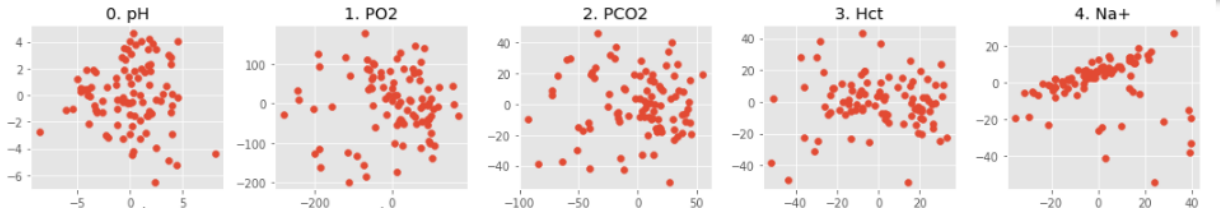


Figure 11: Part of the results from FPCA

The interpolated data about the blood gas parameters was further examined, using the Predictive Power Score (PPS). The PPS is an alternative correlation metric, that can detect non-linear and asymmetric relationship between features, even for not numerical ones. For example, it can be applied for feature selection as PPS shows which features can be predicted by others, so the ones that do not add new information can be eliminated [45]. The PPS matrix of the parameters is visualized on Figure 12, where target features are on the y and predictors are on the x axis.

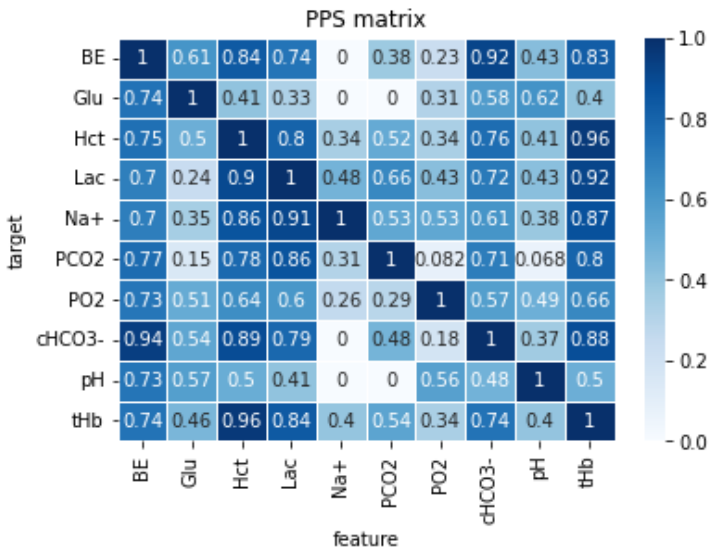


Figure 12: PPS matrix of parameters

Darkness of the cells represents stronger predictive power, and relatively dark cells are present for every target. This means, for each parameter there are some good predictors among the others. Considering the results of all the different tests, linear and non-linear relationships can be shown among the blood gas parameters.

5.3 Baseline models

To explore the relationships between blood gas parameters and patterns in the time-series data, many different questions were investigated. Parameter values of the patients were predicted using their past values, other parameter values and other patients' measurements as well. For the first approach, regression models were tried for prediction. After that, two other directions were explored, time series models and neural networks. All models were tested on one patient first, because of the limitation of time and resources, and to see if the algorithms are suitable to be applied on all patients. The baseline models are summarized in this table:

Model name	Model type	Predictors	Target
Regression 1	ExtraTreesRegressor with K-fold	50-time step lagged values of own other parameters	one parameter
Regression 2	ExtraTreesRegressor/ BayesianRidge / LinearRegression with K-fold	50-time step lagged values of all parameters from other patients	one parameter
Time-series	AutoARIMA	values of own same parameter	one parameter
Univariate	Neural network	1-time step lagged values of own same parameter	one parameter
Multivariate 1	Neural network	values of own other parameters	one parameter at same time step
Multivariate 2	Neural network	values of same parameter from other patients	one parameter at same time step
Paralell	Neural network	1-time step lagged values of own all parameters	all parameters

Table 4: Summary of baseline models applied for each patient

5.3.1 Linear regression and tree-based models

The goal of regression was to predict values of one blood gas parameter for patients individually. For one *Regression1* model, the predictor variables were the patients' own measurements of other parameters. The target variable was a value measured 50 time steps later, than the values of the predictor variables. With the help of the *PyCaret* library, several types of regression models were applied and evaluated on the one randomly selected patients' data, using each parameter as target variable. According to the results, the *ExtraTreesRegressor* from *scikitlearn* package model had the best performance sorting by the *R-squared* metric. It was applied on every patient's data using 5-fold cross validation. The performance is evaluated by the root mean squared error (RMSE) of all trials. The prediction made in the last fold and the corresponding test values are plotted for some parameters of this patient on Figure 13, with the time steps in test interval on the x-axis.

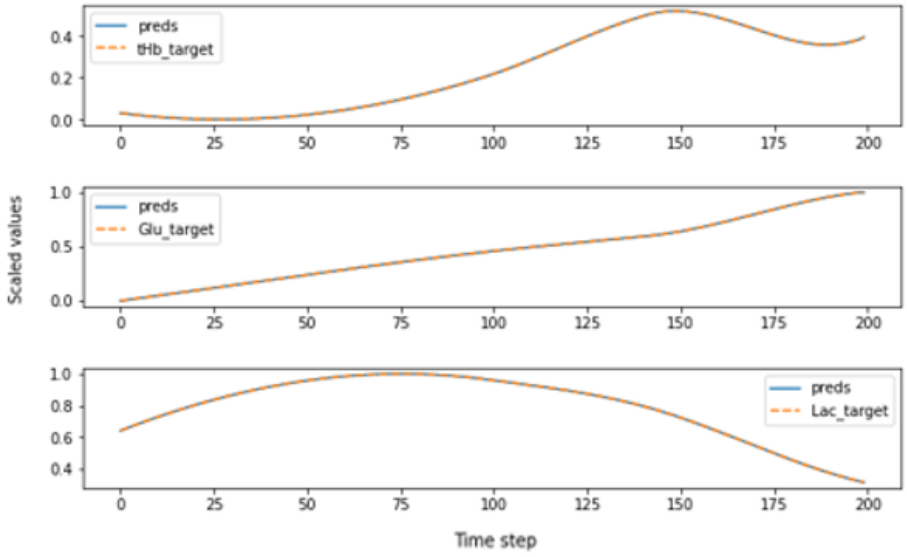


Figure 13: Comparison of predicted and test values for regression model (y-axis: normalized predicted/test values of parameters)

For the second approach in *Regression2* model, the predictor variables were the measurements of all parameters from all other patients. The target variable was a 50 time steps late measurement and the best performing regression models were defined as before too. The best model was *ExtraTreesRegressor* for the *Glu* parameter, the *BayesianRidge* for *Hct* and *Na+* parameters, and *LinearRegression* for all other parameters. The corresponding models were applied using K-fold cross validation on each patient's data.

In both cases, the average RMSE of the K-fold validation was collected for train and test results for each patient, using each parameter as target. Further analysis of these results is discussed in *Section 5.4*.

5.3.2 Time series models

Performance of time-series models were tested on a problem, where the goal was to predict values of one parameter based on the patients' own past values of that parameter. Using the *PyCaret* library [33], many kinds of time-series models were compared against each other, by being applied on one randomly selected patient's *PO2* data.

According to the results in *pycaret*, the *AutoARIMA* model from *pmdarima* package had the best performance. This type of model automatically defines the most optimal parameters for an ARIMA model by conducting differencing tests [46]. The ARIMA model is a variation of ARMA model, which contains the letter *I* for 'integrated' because it uses differencing to make the series stationary and then fits it to the differenced data to finally integrate it to provide a model [13]. For training, 200 time steps were used, and the remaining 100 for testing. Comparison of predicted and test values in the test time interval (100 time steps) for some parameters of this patient is plotted on Figure 14.

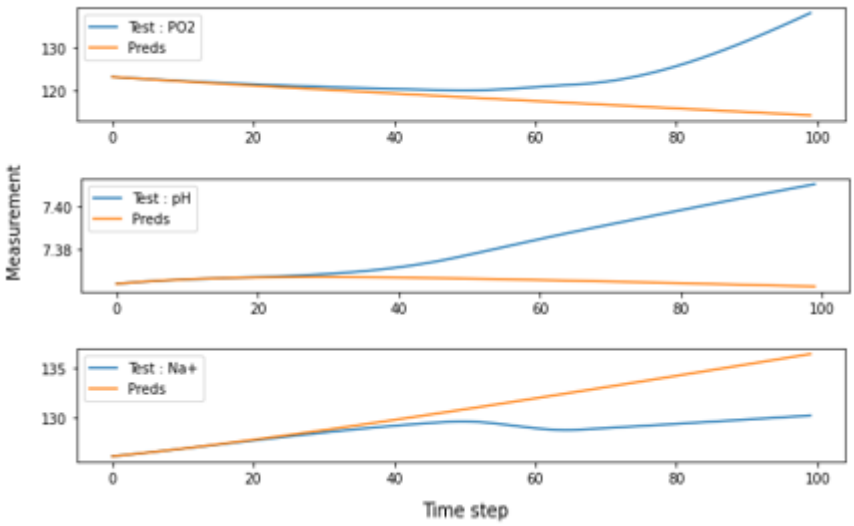


Figure 14: Comparison of predicted and test values for *AutoArima* model

From the plots and MSE values it is clear that the predictions do not follow well the actual values. The reason for this might be the non-stationarity of the series. As an infinite number of non-stationary structures can exist, Chatfield [13] also emphasized that the ARIMA model is only capable of describing certain types of non-stationarity series. He

also stated that relying on automatic ARIMA modelling is complicated and requires considerable experience.

To learn about stationarity in all time-series, the data for all patients' each parameter was tested using the Augmented Dickey-Fuller test [47]. The number of stationary series each patient has (out of the 10 different parameters) was summarized on Figure 15.

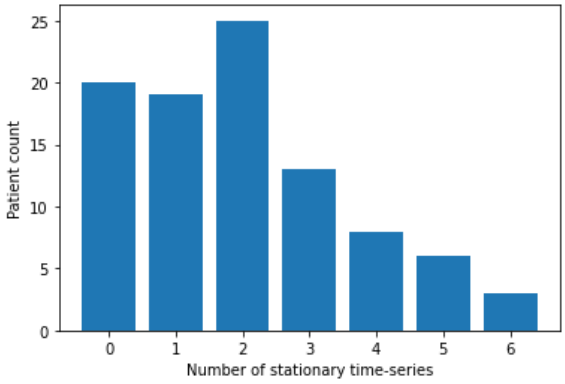


Figure 15: Patient count in terms of number of stationary series

According to the plot, more than half of the patients have less than 3 stationary time-series. Furthermore, in multivariate time-series modelling problems, the presence of non-stationarity makes the modelling complicated and even achieving stationarity does not always lead to satisfying results [13]. Because multivariate modelling is necessary for learning about relationships between the different parameters, another direction was considered for further modelling.

5.3.3 Neural network models

The next step was exploring neural network models, as they do not have any criteria about the time series data, like stationarity. For creating MLP models, the *Sequential* model and *Dense* layer type was used from *Keras* library [21]. The *Sequential* model can deal with simple and layer-based problems, taking one input and giving one output. The *Dense* is a type of layer where all connections are very deep, meaning the neurons get their input from all other neurons in the previous layer of the network.

5.3.3.1 Univariate MLP

The first *Univariate* MLP model was tested on the same problem as the time-series models, aiming at the prediction of values for one parameter based on patients' own measurements of the same parameter. The prediction was calculated for each patient

and each parameter separately. The target variable was the time-series of one parameter and the predictor variable was a 1 time step lagged time-series of the same measurements.

The input was scaled to values between 0 and 1 with *MinMaxScaler* [48] estimator which scales input values with the following transformation:

$$X_{scaled} = \frac{X - X_{min}}{X_{max}} * (1 - 0) + 0$$

60% of the scaled data was used for training the model, the rest for testing. Two Dense layers were used in the model, one with *relu* activation function, and the other layer with *linear* activation. The model trained for maximum 200 epochs with stopping early if there was no improvement for 30 following epochs. For adjusting the weights and optimizing the mean square error as a loss function, the Adaptive Moment Estimation (*Adam)* optimizer was used. The model was trained 5 times and the RMSEs of the separate runs were averaged to give a final metric. Figure 16 shows how the loss of the training (*loss*) and test set (*val_loss*) is changing by epochs during the last 3 run times for a randomly selected patient's *PO2* data.

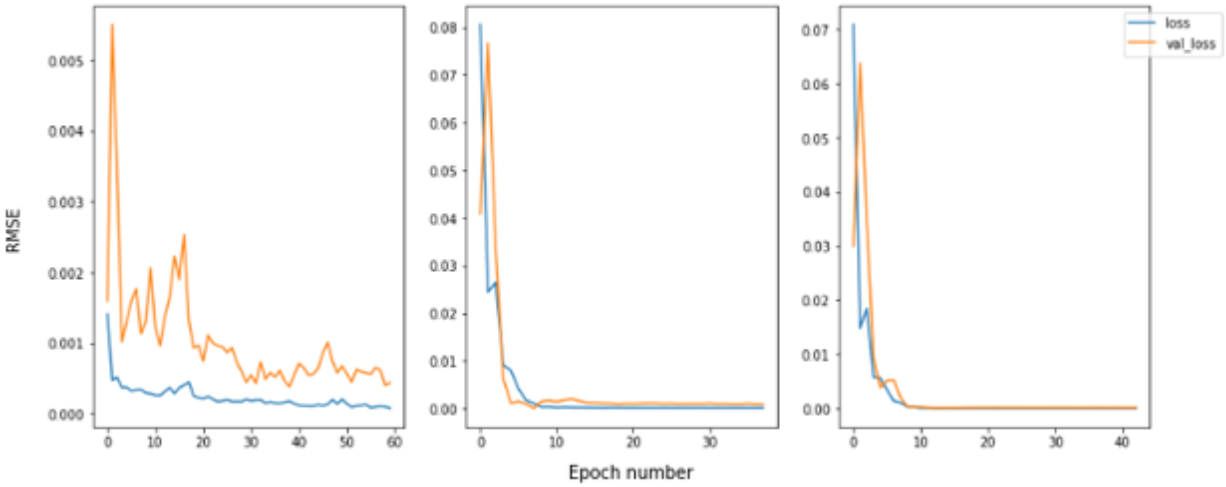


Figure 16: Training and validation loss of MLP model with lagged data

As the values of different parameters were scaled between 0-1 before, the RMSEs range on the same scale and can be compared. The average RMSE of the blood gas parameters from averaging the results of each patient are shown on Figure 17 below.

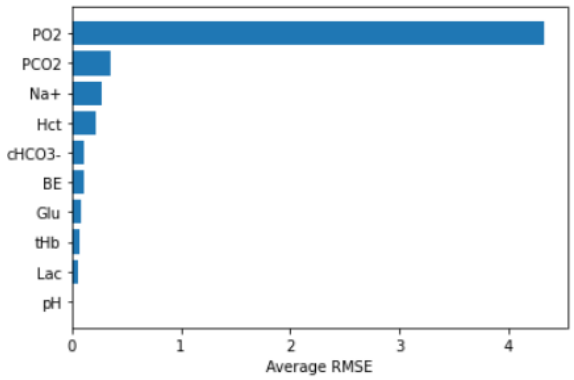


Figure 17: Average RMSE for parameters using all patients' results

From the plot it is clear that the *PO2* parameter had much higher errors, compared to the other parameters. The average RMSEs from this plot are on Table 5, from which *pH* parameter has the lowest average error.

param	Average RMSE
pH	0.003737
Lac	0.062365
tHb	0.071261
Glu	0.089170
BE	0.107871
cHCO3-	0.115080
Hct	0.217398
Na+	0.276093
PCO2	0.349818
PO2	4.316950

Table 5: Average RMSEs for Univariate MLP

5.3.3.2 Multivariate MLP

The task for *Multivariate1* model was the same as for *Regression1* model, to predict values of one parameter, based on the patients' own measurements of other parameters. The MLP model had the same layers and optimizer as the previous one, it was trained for the maximum number of 2000 epochs, and 70% of the scaled input data was used for training. These settings apply for the following models as well. The predictor variables for one target value were the values of other parameters at the same time step.

The *Multivariate2* model was applied in a way, where the values of one parameter were predicted based on measurements of the same parameter from all other patients. On Figure 18, the distributions of the RMSEs are compared for the two variations of this model.

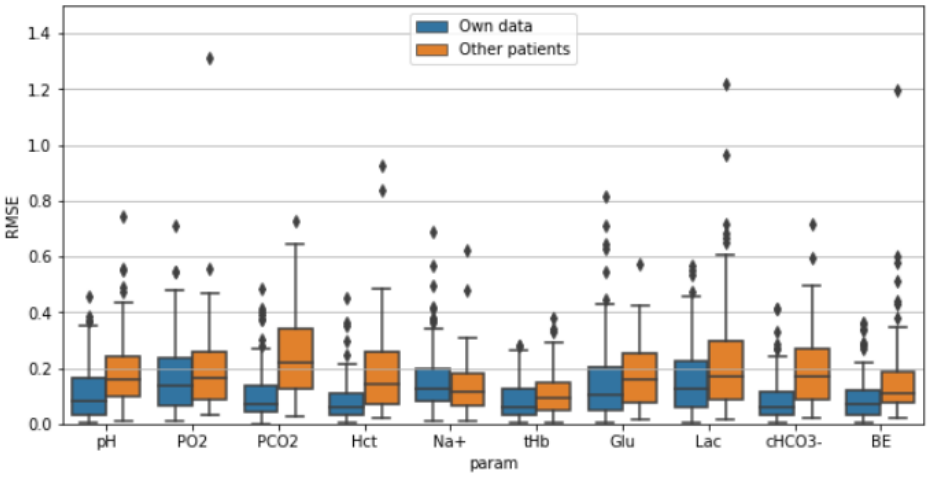


Figure 18: RMSE distributions for two variations of multivariate MLP models

From the plot the conclusion is that using a patients' own other parameters for prediction is more accurate in most cases, which is not a surprise. However, the RMSE values of the model where other patients' data is used for prediction are not much greater and predicting the change of a blood gas parameter accurately without any information of the patient might be more interesting.

5.3.3.3 Multivariate MLP with parallel series

Using a different kind of model at the *Paralell* model, each blood gas parameter could be predicted in parallel, based on the patients' own measurements. A target variable (vector) in this case consists of a value for each parameter at a certain time step and the predictor variables were values of each parameter from the previous time step. The plots on Figure 19 show the comparison of predicted and test values in the test time interval (120 time steps) for the same patient's data which was used for the *AutoARIMA* model in section *5.3.2*.

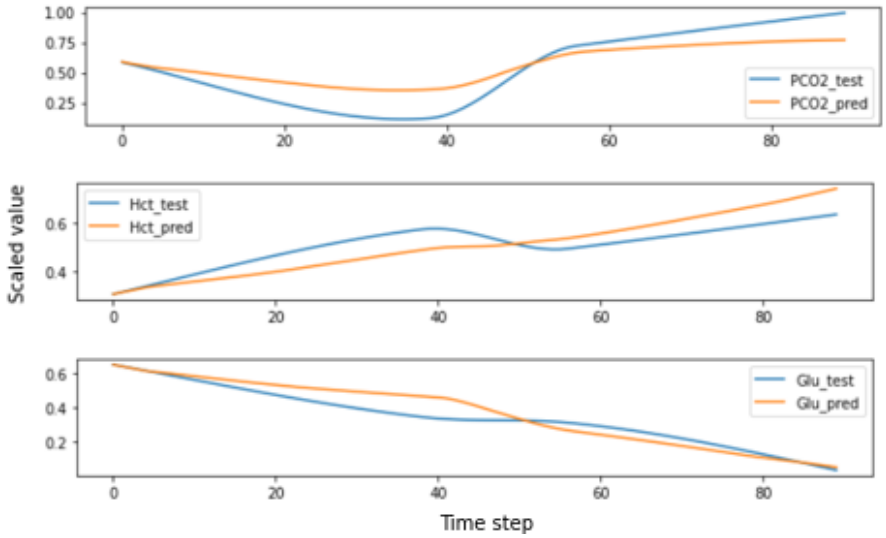


Figure 19: Comparison of predicted and test values for multivariate MLP model

The predictions in most cases seem to be close to the test values, surely following them better than the ones on Figure 14 made by the time-series model. The RMSE values was collected for this model as well for each patient.

5.4 Baseline model evaluation

5.4.1 Score comparison

In this section, the results are summarized and compared for the models, that were applied on the time-series for every blood gas parameter of each patient. The target of these models was always a time-series for one or all (parallel MLP) blood gas parameters of one patient. These 6 different models were applied for each patient, and the RMSEs were summarized. Averaging the RMSEs for the 10 blood gas parameters on Table 6 the *Regression2* model is the only one having an average RMSE above 1.

	Average RMSE
Regression2	9.157914e+07
Univariate	5.609743e-01
Multivariate2	1.858194e-01
Regression1	1.479383e-01
Multivariate1	1.235471e-01
ParalelI	7.575317e-02

Table 6: Average RMSEs based on all parameters for compared models

The poor performance of the Regression2 model can also be seen by plotting the distribution of test RMSEs among patients on Figure 20.

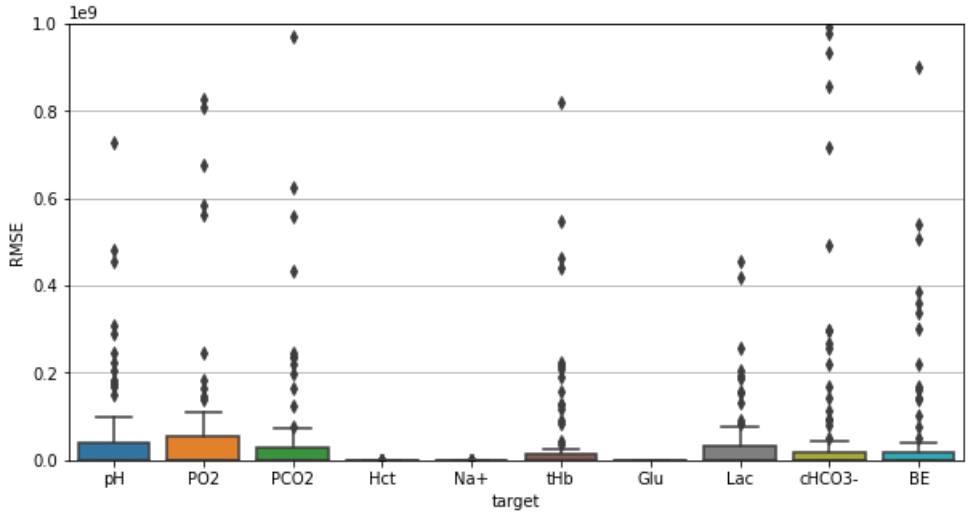


Figure 20: RMSE distributions for *Regression2* model

The extreme errors are not surprising, as here the target parameter was predicted only by using measurements from other patients. As every patient has their own personal dynamics of biochemical processes in the arterial blood, it is a difficult problem. There are many outlier errors with huge differences in this regression model, meaning the model could not find right connections for predicting between different patients. As there are non-linear relationships in the data, a linear regression model may not be able to capture patterns.

On the other hand, the *Multivariate2* model also predict based on other patients, but only from the same parameter as the target, and this way had not much higher RMSEs, than the models where patients' own data was used.

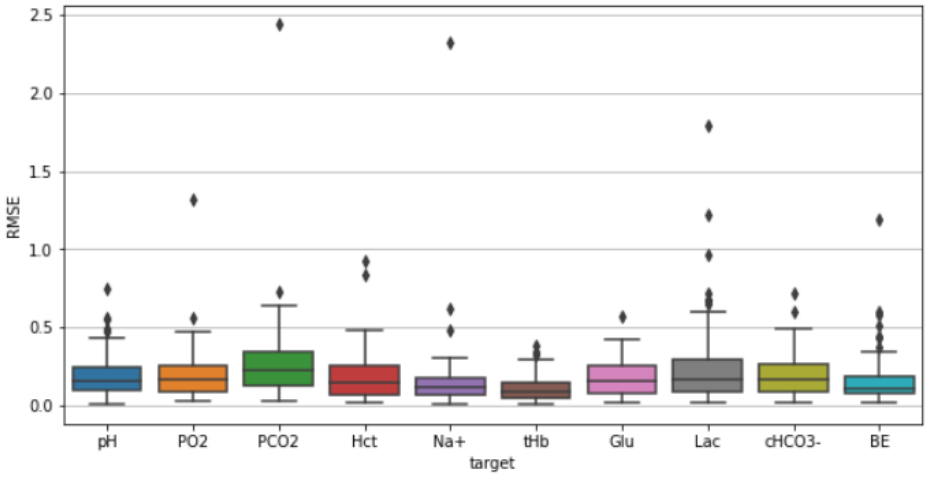


Figure 21: RMSE distribution for *Multivariate2* model

By looking at the best performing models for the different blood gas parameters individually, the results on Table 7 show the same as the previous table, that either the *Parallel* or the *Univariate* model is the best.

	Best model
BE	Parallel
Glu	Parallel
Hct	Parallel
Lac	Univariate
Na+	Parallel
PCO2	Parallel
PO2	Parallel
cHCO3-	Parallel
pH	Univariate
tHb	Univariate

Table 7: Best performing model for each blood gas parameter

The distributions of RMSEs for these two models range on a lot smaller scale, then for the *Regression2* model. There are still outlier values, but the RMSEs are overall quite small for each blood gas parameter.

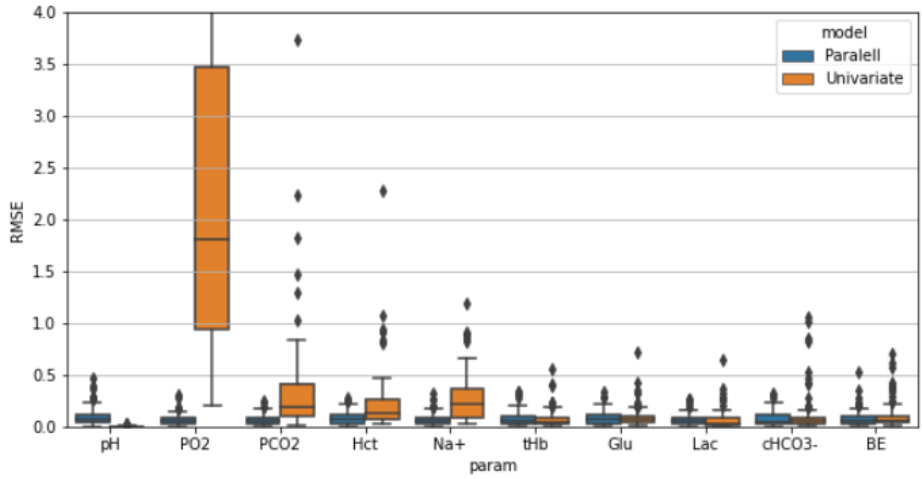


Figure 22: RMSE distributions for *Parallel* and *Univariate* models

The reason for the accurate predictions might be that the patients' own parameters were used as predictors and based on the correlation analysis in *Section 5.2* some blood gas parameters have strong and significant correlation. Furthermore, the PPS matrix also showed that there are some good predictors for each parameters considering all patients' measurements as well.

In this section the input data, the relationships among blood gas parameters and many different questions were examined. Several algorithms were tried to build baseline models, that can give somewhat accurate predictions for the change of blood gas parameters. All these served as a starting point, to see which directions are worth to be explored further for interesting and significant results.

6 Augmented models

Based on the baseline results, the main direction for this section is about improving and understanding the *Multivariate1* model, which can predict blood gas parameters based on each other. Although it was not among the best models when looking at the parameters individually, it had better average RMSE than the *Univariate1* model and it had a more complex task, as it only had other parameters as predictors. After optimization the *Multivariate1* model will be further tested to gain a more stable and reliable estimate of the performance. The feature importance values will be examined as well to understand the influence of different blood gas parameters on each other and to see if there were any parameters which would not need to be measured at all.

Another direction is the development of the current most accurate model, the *Parallel* model, which predicts all parameters in parallel. The goal here is to examine the possibilities for predicting further in time, not only for the next time step. In addition, the performance of the *Multivariate2* model is tested and examined more deeply to understand how well it is capable of learning patterns among the patients' personal dynamics. It is also important to find out if there are any blood gas parameters where the model can give acceptable estimates for completely unknown patients.

Furthermore, during the project additional data became available. The decision was to expand the current dataset with new patients' data and apply the knowledge gained in the previous sections on a bigger dataset in this section. This additional data contained measurements in a wider time period, even for years in some cases. Unfortunately, the time of the heart transplantation surgery was unknown, so not all the new data could be used. Patients who had measurements in the initial dataset too could be extracted as the new dataset contained those measurements as well. After matching new measurements to patients in the initial dataset, 48 additional patients' data became suitable for analysis, having 142 patients altogether.

6.1 Examining reference intervals

To gain a better understanding and more reliable evaluation of the models' performance, the MAPE evaluation metric was introduced. This metric was chosen based on the study written by Wajs et al. [4], where the writers also tried to develop a predictive

algorithm for arterial blood gas measurements and trained it with historical samples. In this study, the writers also used ANN for prediction and predicted the different blood gas parameters in parallel.

The MAPE is calculated for the original values, not the normalized ones, so questions about the predicting correctly regarding reference intervals can also be answered. For example, on Figure 23 the original *PCO2* measurements are plotted for a random patient along with the values calculated by adding (red line) or subtracting (blue line) the error percentage. The range bounded by the red and blue line is the possible prediction range. The lower and upper limit on the plot are the boundaries of the reference interval in which the patient's PCO2 measurements can be considered normal. From the plot it can be spotted that in some cases the model would predict that the value is out of reference interval when it is actually not, or it would predict that the value is in the reference interval, when it is not considered as normal. False normal (FN) predictions are obviously worse than False abnormal (FA), as for some blood gas parameters values going outside reference intervals can lead to undesirable conditions.

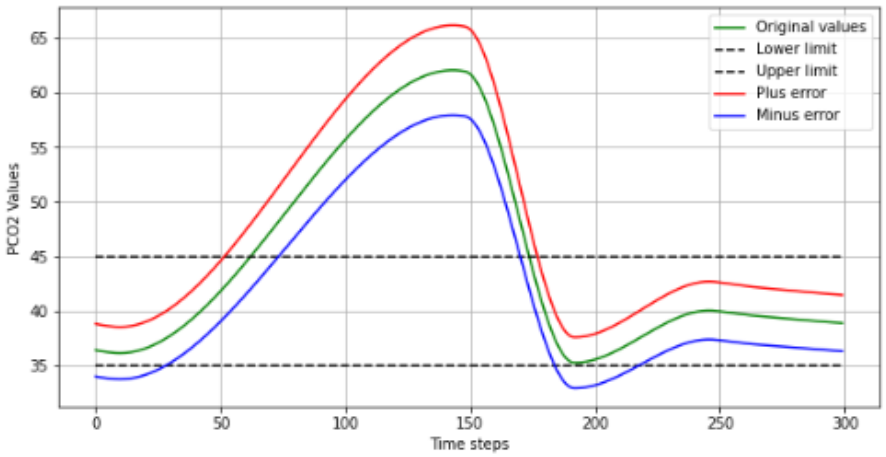


Figure 23: Change of original values and values with error by reference intervals

For example, a low level of *PCO2* indicates that the patient is not oxygenating properly, while a high *PCO2* indicates underventilation. At a *PO2* below 60 (mmHg) the patient needs supplemental oxygen, while below 26 (mmHg) the patient is at risk of death [5]. To see what danger the final calculated errors mean at different blood gas parameters, each model's performance is evaluated by checking the number of time steps where it would give correct, FA or FN predictions regarding reference intervals. These numbers will be referred as metrics of the prediction range's correctness and are used in the evaluation of the *Multivariate1* and *Multivariate2* models.

6.2 Predicting parameters from each other

6.2.1 Hyperparameter optimization

In this project the goal of HPO was to find the hyperparameter combination for *Multivariate1* model that can improve the evaluation scores, so the model can learn the relationships of blood gas parameters better and give more accurate predictions based on each other.

6.2.1.1 Tuned hyperparameters and evaluation

As the ANNs used in the project are simple MLPs with an input an output layer, the hyper-parameters related to the construction of the model were not changed. The loss function, activation function and the optimizer were already chosen as well. The focus was on tuning hyper-parameters related to the optimization and training process.

The learning rate was the first to be optimized, as it is considered to be one of the most important hyper-parameters. The learning rate defines the step size at which the weights are updated during training. A large step size makes the training process faster as the model moves quickly towards the minimum point of the loss function, but there is a risk of overshooting that point and oscillating around it, without ever converging. A small learning rate can converge smoothly, but it can take a long time to reach the minimum. The goal is to find a learning rate with which the model can steadily improve and find the best weights to minimize the loss function in a reasonable time [23]. By examining the *Multivariate1* model's loss by epoch during training with default learning rate of 0.001, the large fluctuations suggest instability in the learning progress.

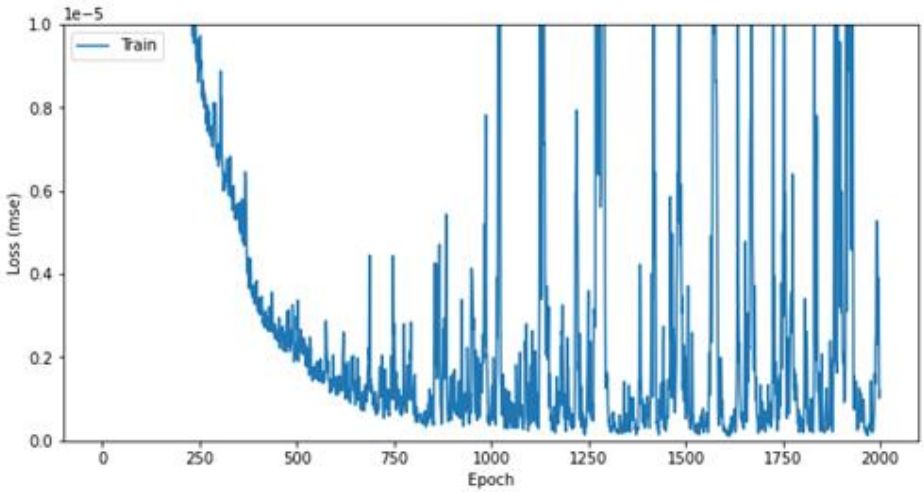


Figure 25: Training loss by epochs with original learning rate

Certain parameters need to be defined for the sklearn library's GridSearchCV function, such as *param_grid*, which is a dictionary with the names of parameters as keys and lists of parameters to try as values. The *scoring* parameter also needs to be passed to the grid search function if the estimator does not provide a score function. This parameter determines the evaluation strategy for the performance of the cross-validated model on the test set. For the first grid search, possible values (0.01, 0.001, 0.0001, 0.00001, 0.000001) for the learning rate were passed to the GridSearchCV funciton. Based on the results, out of these values the best learning rate is 0.0001. The change of the model's training loss by epochs with the optimized learning rate is clearly way smoother than before. The RMSE in this case decreased from 0.00083 to 0.00051 and the MAPE from 0.089 to 0.058.

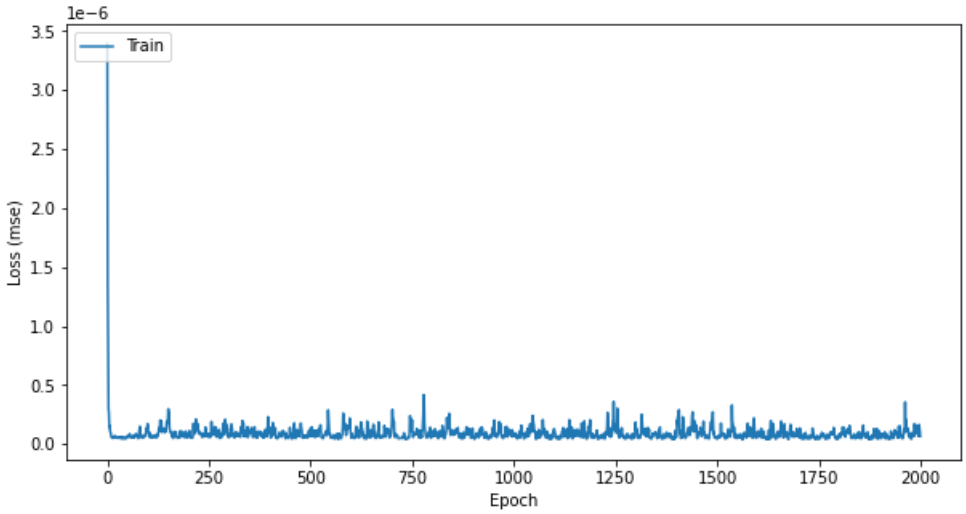


Figure 24: Training loss by epochs with optimal learning rate

After optimizing the learning rate, different weight initializers were passed to the grid search function. These initializers set the initial random weights of Keras layers [49]. Out of the eight initializers (*uniform, lecun_uniform, normal, zero, glorot_normal, glorot_uniform, he_normal, he_uniform*) the three with the best evaluation score were *glorot_uniform, normal* and *lecun_uniform.* These initializers were passed to the final grid search along with possible mini-batch and epochs sizes. The mini-batch size defines the number of processed samples before weight update, while epoch number defines the number of times the entire training dataset is passed [23]. The best hyper-parameter combination based on the last grid search results is has *normal* as weight initializer, 32 as batch size and epoch number remained 2000.

To compare the overall performance of the original and optimized *Multivariate1* models, it was tested with the tuned hyper-parameters for all patients and each blood gas parameter. This time, a 5-fold validation was used for testing the model, not just a simple 70-30% split of the dataset, as before. For each patients' every parameter, an average test RMSE and MAPE were calculated from the 5 folds, along with standard deviation of the errors. Figure 26 compares the averaged RMSEs for the original and hyper-parameter optimized models on the upper bar plot, and the averaged MAPEs below. By only looking at these plots, the conclusion could be that the hyper-parameter optimized model performs better, as it has lower errors on average.

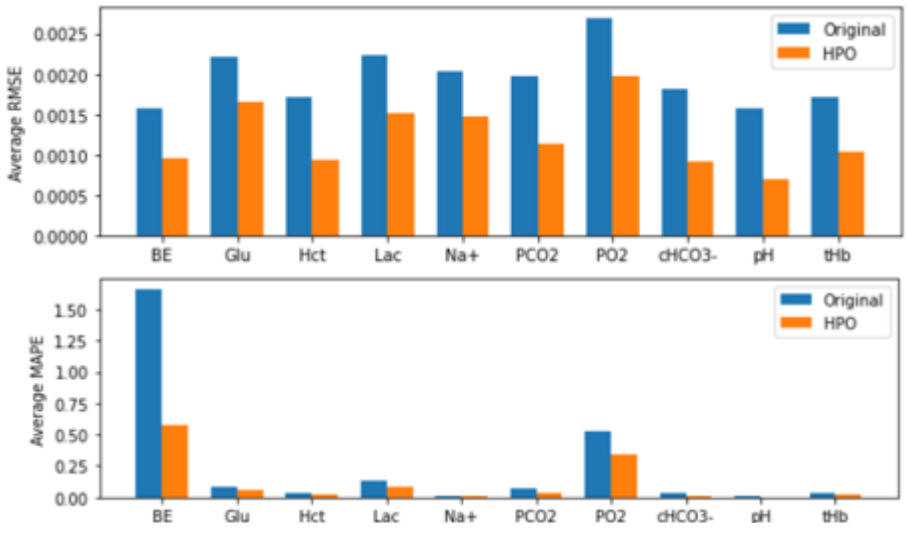


Figure 26: Comparison of average error metrics for original and optimized model

However, when looking at the CVs of the error metrics as well on Figure 27, it looks like the optimized model has errors varying on a wider range for almost every parameter.

It means that even though the optimized model has lower average errors among all patients, its performance is less stable with more outstanding errors.

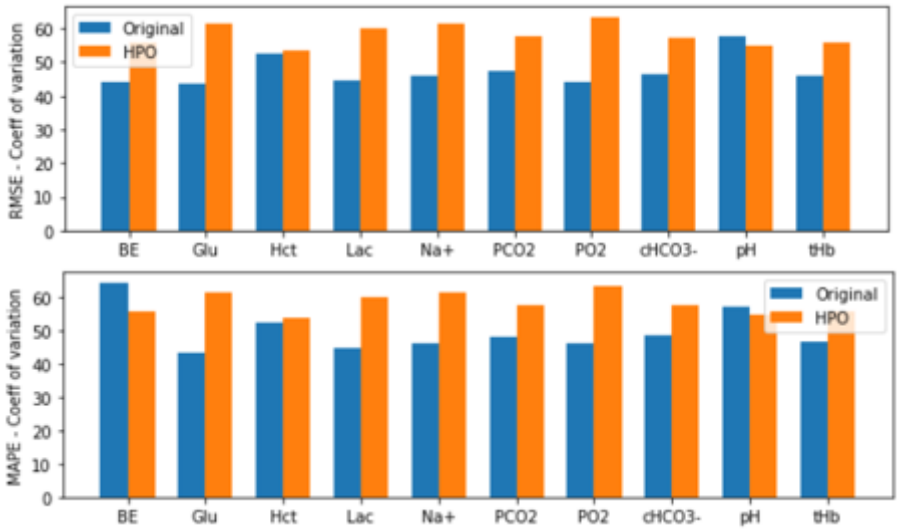


Figure 27: Comparison of error CVs for original and optimized models

Finally, the correctness of the possible prediction range was compared for the models on Figure 28. As the prediction range is calculated with an averaged MAPE, it is not a surprise that the optimized model has larger ranges for correct predictions. Even though the optimized model would make less FA predictions on average, there are some parameters where the model would make more FN predictions.

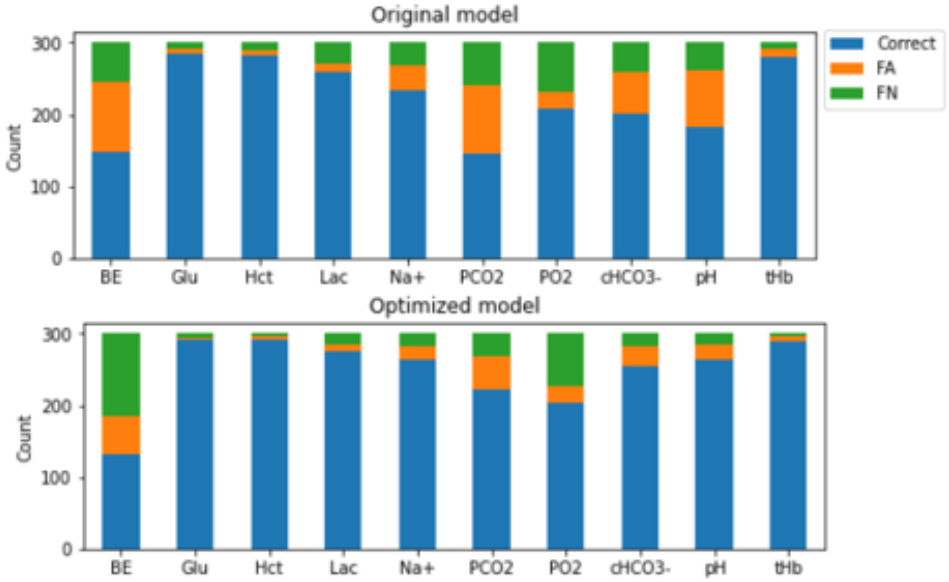


Figure 28: Comparison of prediction range correctness for original and optimized model

Taking all this into consideration, the model with the original parameters might be better, because stability is very important for this problem. It must be noted that the

hyper-parameters were optimized on one patient's data because of time and resource limitations, which can explain why the errors have bigger variances. The chosen hyper-parameters might not be the optimal for some other patients with very different personal dynamics. It requires further research and different approach to find hyper-parameters that are optimal in general if such parameters even exist.

6.2.2 Feature importance examination

To understand deeper the relationships between blood gas parameters and examine their effects on each other, the feature importance of *Multivariate1* model was examined.

For this purpose, the SHAP method (SHapley Additive exPlanations) was used, which is based on cooperative game theory and is used to enhance transparency and interpretability of models. The SHAP values can help in explaining how different features affect the model's output. The absolute SHAP value of a feature shows how much that feature affected the prediction, while the sign of the SHAP value indicates the directionality [50].

On Figure 29 the SHAP values are plotted with a beeswarm plot for predicting *pH* for a patient. On this plot, the dots represent single observations. The features are ordered by their effect on the model's output, so in this case the *PCO2* parameter had the biggest effect on predicting the *pH*. The color of a point shows how high or low value that observation has compared to other observations. It seems like both higher and lower *PCO2*, *tHb* and *Glu* values had a positive impact on the prediction, while for example in the case of *Na+* only high values had positive impact on *pH*. Low *Na+* values decreased the predicted *pH* value.

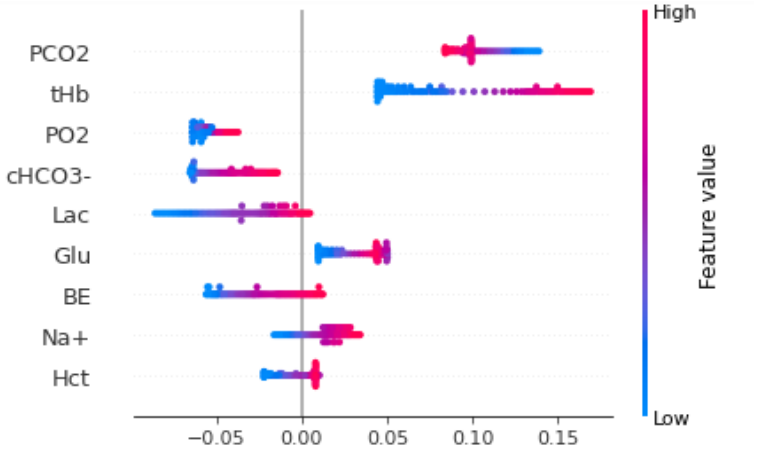


Figure 29: SHAP values of *pH* prediction for one patient

The SHAP values were collected for each patient's every parameter to see a general picture of the feature importance. On Figure 33 the average feature importance among all patients (*y* axis) is plotted for *pH*, *PO2*, *cHCO3-* and *Hct*. The directionality of the impact is represented by colors as well. From these plots general conclusions can be drawn, such as *BE* having the largest negative impact on average for *pH*, or *tHb* increasing the most predicted values for *Hct*. By looking at the plot for every blood gas parameter (including the other 6 which are not on Figure 30 it is possible to identify 1-3 features for each target parameter with much greater importance compared to other features.

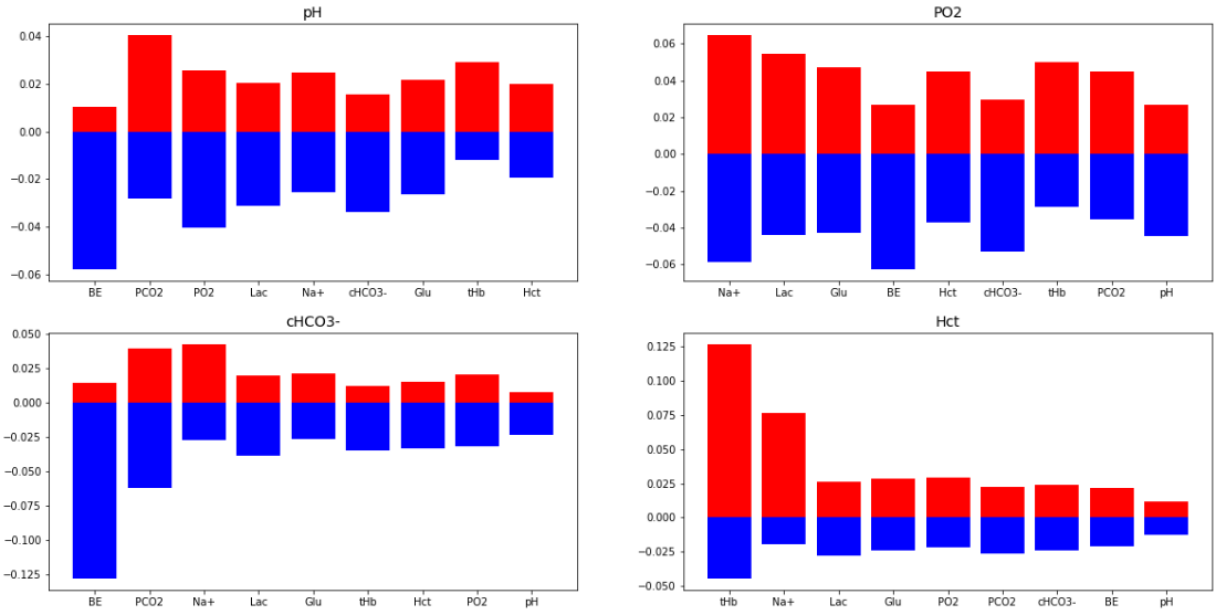


Figure 30: Average feature importances among all patients for *pH, PO2, cHCO3-, Hct* (y-axis: averaged SHAP values)

The feature importance was summarized with percentage of patients as well. The most important features in percentage of patients are presented on Figure 31 for each parameter, where darker cells show bigger importance. From the heatmap it can be observed that there is always a little percentage of patients for whom a certain parameter is the most important feature. The only case where a feature was not the most important for any patient is *pH* for predicting *tHb*. Other than that, even though some features had much stronger impact on average, when looking at patients individually, all features are most important for someone. Because of this, no features should be excluded from predictor variables.

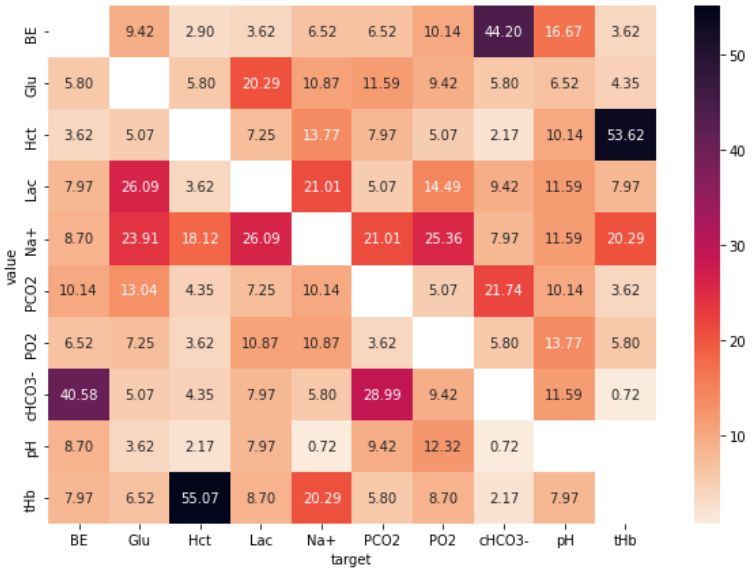


Figure 31: Most important features in percentage of patients

6.3 Predicting further in time

In this section, the *Parallel* MLP model was developed further to give an output for more time steps ahead. To achieve this, a multiple parallel input and multi-step output MLP was used, where the *m* input vectors, and *n* output vectors contained values for each parameter at the previous *m* or following *n* time steps. The difference between this model and the previous ones is that those only predicted for 1 time step ahead, and those predictions were summarized. However, this kind of model predicts *n* step further in time, which can be useful if data is only at hand for 300-time steps.

The model was tried out with different number of input and output vectors, always using 30% of time steps for testing and 70% for training. First, samples are created by splitting the data to a three-dimensional array with predictors (*X*) and another with target values (*Y*). The shape of array *X* is (300, *m*, 10) and for *Y* (300, *n*, 10) as the dataset contains 300 time steps for 10 parameters. Then both *X* and *Y* are split into *X_train*, *Y_train*, *X_test* and *Y_test* arrays. These arrays are also three-dimensional, but the first dimension is not 300. In case of an *n* length output, the number of times steps that can be used for training and testing equals 300-*n*, because when *n*=100 then 100 time step length prediction cannot be validated after the 200. time step. In each sample, the input and output time steps are shifted with one time step. The size of train and test sets for different length output models is on Table 8.

Num of time steps	m=1 n =5	m=1 n =20	m=1 n =50	m=1 n =100
train 70%	207	196	175	140
test 30%	88	84	75	60
sum (all-n)	295	280	250	200

Table 8: Size of train and test sets for multi-step output *Parallel* models

The RMSEs and MAPEs were calculated for each *n* length prediction on every patient's data. Then the error metrics among patients were averaged for different parameters and different time steps of the test set. How the average RMSEs changed for *Na+* when the model predicted for different *n* lengths based on 1 previous input, is plotted below. Average RMSEs are clearly getting higher when predicting further in time.

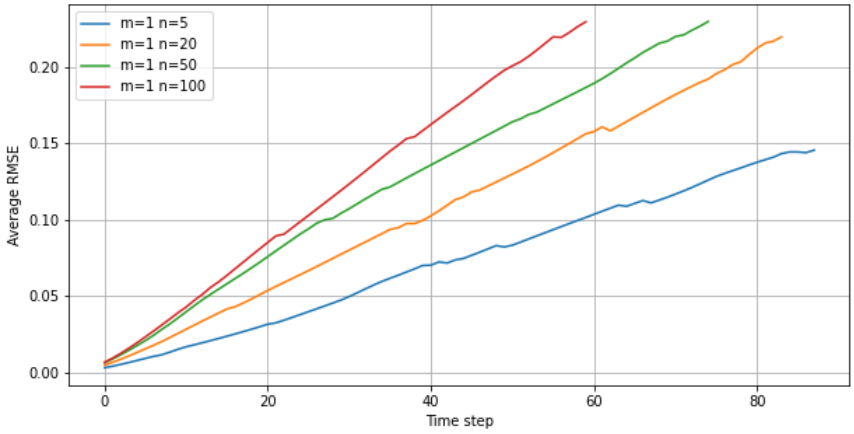


Figure 32: Change of average RMSE for Na+ parameter at different prediction lengths

Based on this, the model was tested with different number of input vectors as well. In this case, the number of times steps that can be used for training and testing equals 300-*n*-*m*. For example, if *m*=5 and *n*=100 predictions can only be validated until the 200. time step and because more time steps are used as input, predictions can only be made until the 195. time step. A calculation for each case is on Table 9, and the time step (ts) indexing goes from 0 to 299. For example, when *m*=5 and *n*=100, the first prediction is made using time steps 0-4 and the output is given for time steps 5-104. In the second prediction, the input and output intervals are shifted with 1 time step, as in the second row of the table.

m=5 n =100	input ts	output ts	m=20 n =100	input ts	output ts	m=50 n =100	input ts	output ts
1.	0-4	5-104	1.	0-19	20-119	1.	0-49	50-149
2.	1-5	6-105	2.	1-20	21-120	2.	1-50	51-150
3	2-6	7-106	3.	2-21	22-121	3.	2-51	52-151
.	.	.	.			.		
.	.	.	.			.		
195.	195-199	200-299	180.	179-199	200-299	150.	150-199	200-299

Table 9: Example of input and output time steps for multiple input multi-step output *Parallel* models (ts = time step)

The change of average RMSEs is plotted on Figure 33. There are only very small differences in average RMSEs when increasing the number of input time steps. By looking at results for other parameters as well, the conclusion is that it does not make much difference to predict parameters in parallel for next 100 time steps based on measurements in the previous 1 or 50 time steps.

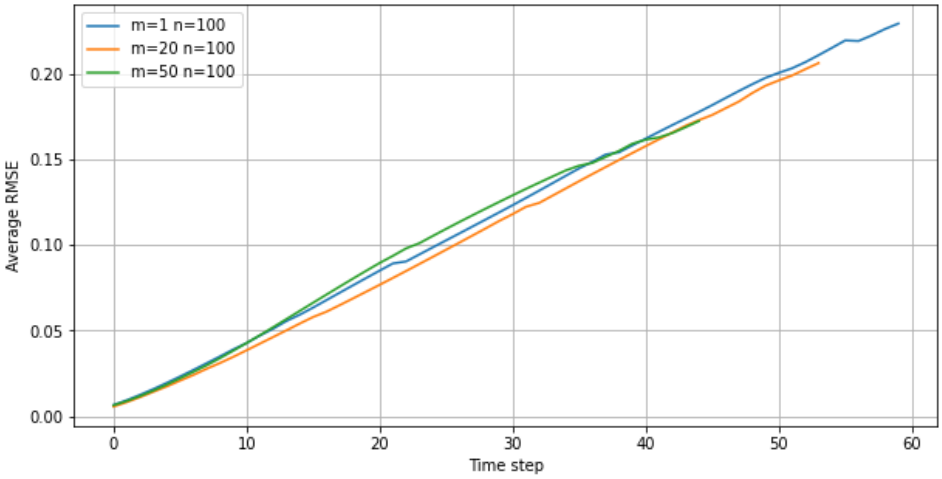


Figure 33: Change of average RMSE for Na+ parameter at different length inputs

On Table 10, the average RMSEs are summarized for parallel models with different length outputs. From the table it looks like the RMSE is not always increasing with the output period length. For example, at BE the average RMSE is lower when predicting for 100 time steps ahead, than for 50 or 20.

param	m=1 n=5 RMSE	m=1 n=20 RMSE	m=1 n=50 RMSE	m=1 n=100 RMSE
BE	0.055526	0.079567	0.090993	0.074801
PCO2	0.069845	0.101150	0.120205	0.088038
Lac	0.070200	0.112640	0.119679	0.091940
Na+	0.072200	0.106700	0.121129	0.089816
PO2	0.076329	0.132422	0.155444	0.094261
tHb	0.076716	0.099533	0.099007	0.090358
cHCO3-	0.077064	0.103750	0.109279	0.098312
Glu	0.078375	0.103577	0.122086	0.101321
Hct	0.087019	0.114361	0.129090	0.094824
pH	0.093431	0.140502	0.159369	0.112355

Table 10: Average RMSE for paralell models with different length output

On Table 11, the average MAPEs are also summarized for parallel models with different length outputs. For each blood gas parameter, this average error was below 1%, even when predicting for the longest length. Here it is also true, that the longest prediction period does not always mean the highest average MAPE.

param	m=1 n=5 MAPE	m=1 n=20 MAPE	m=1 n=50 MAPE	m=1 n=100 MAPE
BE	0.255588	0.293917	0.280922	0.396982
PCO2	0.472413	0.587297	0.682396	0.564236
PO2	0.492506	0.804749	0.932991	0.587270
Lac	0.501960	0.725019	0.693168	0.632349
Glu	0.514393	0.648557	0.698250	0.668438
tHb	0.517404	0.614879	0.608209	0.571473
Na+	0.520254	0.664649	0.661103	0.627039
cHCO3-	0.562952	0.704828	0.670433	0.686677
Hct	0.595250	0.754634	0.774756	0.658114
pH	0.631346	0.877223	0.936562	0.757105

Table 11: Average MAPE for paralell models with different length output

Based on this, it can be possible to predict blood gas parameters in parallel further in time with a low average error. The error might, but not necessarily increase with the length of the prediction period and more input time steps does not guarantee more accurate predictions.

6.4 Predicting unknown patients

In this section, the *Multivariate2* model is examined more deeply, as it could be interesting and useful if a model could make good predictions without any knowledge of a patient.

The model itself was not changed, only the way it was trained and tested. At the previous version in *Section 5,* it was trained on 70% (first 210 minutes) of all patient's data and tested on the remaining 90 minutes. It was tested on each patient's every parameter, by retraining every time using all other patient's data of the same parameter. This way the target patient was not fully unknown, as their data was used for training as well. In the new approach, 70% of patients is used for training and 30% for testing, ensuring that the model is tested on completely unseen patients. So first, the patients were randomly split to train and test group. Then during training, the model tried to predict all 300 minutes of a random patient in the train set based on the data of the others. After that,

the trained model was tested on all patients in the test set and the error metrics were calculated. This process was repeated 10 times, always shuffling the patients before creating the train-test groups, to achieve more stable results.

The error metrics were averaged for all 10 rounds among the test patients. These average metrics are in the table below. From this table, the *Glu* parameter has the lowest average RMSE and the *Na+* the lowest average MAPE.

param	RMSE	MAPE
Glu	0.247357	5.909671
cHCO3-	0.310293	3.606751
BE	0.343108	456.189176
Na+	0.387250	2.683705
tHb	0.414694	3.326417
pH	0.419066	4.714648
Hct	0.426473	7.647733
PCO2	0.427267	10.702046
PO2	0.428651	44.267492
Lac	0.470663	15.796436

Table 12: Average error metrics for *Multivariate2* model

However, the standard deviation of the error metrics was also collected to compare the coefficients of variation (CV) among blood gas parameters. The CVs are plotted below on Figure 34. for RMSEs and MAPEs separately, representing the relative variability of error metrics among different test patients.

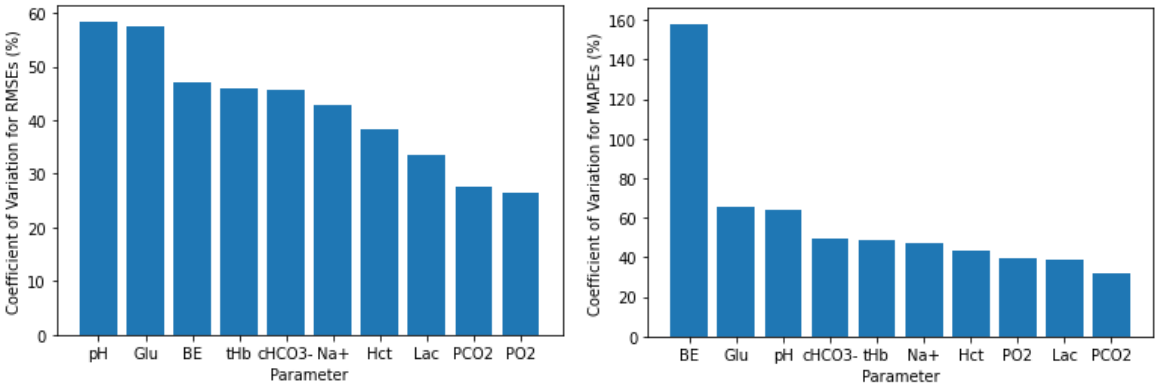


Figure 34: Coefficients of Variation for error metrics

Based on the comparison of CVs among blood gas parameters, the highest variability in RMSEs is at *pH* and for MAPEs it is at *BE*. High CV for errors means that when

predicting these parameters for test patients, the model's performance is not stable as it generates widely varying errors and there might be extremely large errors for some test patients. Considering the order of parameters for different metrics, although it is not the same, the same parameters are in the first and last group of five. From these plots it looks like the model can more reliably estimate some blood gas parameters than others for completely unknown patients.

The average MAPEs were also examined in terms of correctly being in or out of reference intervals at each time step with the prediction range (adding/subtracting the average error from the predicted value). There is an example of results for one patient on Table 13. It shows for example that for *PO2*, whether the original value was in or out of reference interval, the prediction range would also be there at each time step. On the other hand, for *PCO2* at 189 time steps the prediction range would be out of reference interval (FA), while the original value was in.

pt_id	param	Correct	FA	FN
24127	PO2	300	0	0
24127	Hct	300	0	0
24127	Na+	300	0	0
24127	tHb	300	0	0
24127	Lac	300	0	0
24127	Glu	219	81	0
24127	BE	204	96	0
24127	cHCO3-	165	135	0
24127	pH	160	140	0
24127	PCO2	111	189	0

Table 13: Prediction range correctness metrics for one patient

In the case of this patient there would be no prediction range that could be falsely estimated to be in the reference interval (FN) when the original values were abnormal. Averaging the number of Correct, FA and FN scores among all patients, shows similar results, see on Table 14.

param	Correct	FA	FN
Glu	289.746377	10.253623	0.000000
Hct	285.268116	14.731884	0.000000
PO2	276.739130	23.253623	0.007246
tHb	274.449275	25.159420	0.391304
Lac	265.782609	34.202899	0.014493
Na+	214.659420	85.340580	0.000000
pH	205.463768	94.536232	0.000000
BE	160.340580	139.659420	0.000000
cHCO3-	139.144928	160.855072	0.000000
PCO2	117.927536	182.007246	0.065217

Table 14: Averaged prediction range correctness metrics among patients

From this, the conclusion is that the average MAPE might be the least dangerous in case of the *Glu* parameter, where the prediction range is correctly in or out of reference interval at 289 time steps on average. It looks like the model was not capable of correctly learning patterns among patients to predict for example *BE* measurements for unknown patients. Another important conclusion is that the model does not really give prediction ranges with falsely normal predicted values, which makes all error ranges more acceptable.

7 Discussion

In this section, the results of the whole project are summarized, including the discoveries of the EDA, the modelling, the optimization, and the feature importance examination. First, the results of the EDA and *Multivarite1* model are presented, then the conclusions form the *Parallel* and *Multivariate2* models are summarized.

7.1 Main results

During the EDA, the focus was on finding differences between patients who survived or died. Based on the statistical test results, significant differences can be observed between these groups in the change of most blood gas parameters during the observed period. By examining the two groups in terms of being in or out of reference intervals, there were significant differences at 6 parameters. Furthermore, significant differences in the change of almost all parameters were also present between patients with different postoperative survival lengths. All these differences show that the time series data from blood gas measurements could be used for predicting postoperative survival length. The goal of the EDA was also to gain insights about the relationships among blood gas parameters to see if they can be good predictors for each other. First, the correlation of each parameter combination was examined and summarized among all patients. Based on this, *cHCO3-* and *BE* parameters had the strongest positive, while *pH* and *PCO2* the strongest relative correlation. Other parameter combinations also had significant correlation among patients, and the PPS matrix also showed that for each parameter there are some good predictors.

In the end of baseline modelling, the *Parallel* and *Multivariate1* models had the best average RMSEs. The final focus of the modelling phase was on the *Multivariate1* model that predicted one parameter using all the other parameters for each patient individually. This model produced average RMSEs between 0.0015-0.0025 for all the parameters, and average MAPEs under 1% for each parameter except *BE,* where it was 1,65%. The optimized model had even lower average errors, but with higher variances, so it was less stable when applying it on all patients. Based on this, it can be concluded that the blood gas parameters can be predicted from each other with small errors. From the feature importance examination, some features were identified for each parameter, that have larger impact on the prediction. However, in the case of all parameters every feature was most important for some (even very little) percentage of patients, so none of them can be excluded.

7.2 Additional conclusions

Additional directions for modelling included predicting the parameters in parallel for different time periods and predicting each parameter for unknown patients based on other patients' measurements of the same parameter.

Based on the results from parallel modelling, it is possible to predict blood gas parameters further in time with similarly low average errors, as in the main model where the parameters are not predicted in parallel. Also, a conclusion of this direction is that the error not always grows as the length of the prediction period is increased and using more input time steps did not always result in lower error. This might be because the most important part of the input is the closest time step to the chosen output period. Others [4] investigating similar problems used most recent samples for prediction instead of all historical data as well.

In case of predicting unknown patients the average RMSEs was higher compared to the other models. It is not surprising, as in the other two problems the patients' own measurements were used and patients can have very different personal dynamics. From the average RMSEs it seems like error for *pH* and *Glu* vary on a wider range than for other parameters, so to find patterns among patients might be the hardest for these parameters, while the easiest for *PO2* and *PCO2*. Looking at the average MAPEs, the *BE* parameter has errors on a very large range compared to the others, outlier errors for some patients. Also, with the prediction range the most false normal predictions would be in the case of *BE*, *cHCO3-* and *PCO2* on average. On the other hand, it looks like the model almost did not make any falsely normal prediction, which is a good result because it means that it can make safe predictions (regarding reference intervals) for completely unknown patients.

7.3 Further work

While working on this project, I came across many different algorithms, techniques, approaches, and questions that can form the basis of further analysis.

With a different approach, the model's hyper-parameters could be optimized in general for all patients, not just using one patient's data. This way the performance of the model could be improved further, and the predictions could be more accurate. In addition, the hyper-parameters related to the construction of the model could also be tuned and

different types of neural networks can be tried out as well, even more complex ones like LSTM.

As there were some significant differences found between groups with different survival length, predicting the survival outcome or length based on blood gas parameters could also be an area of further research. Furthermore, other variables about the heart transplant patients could be included (if possible). If connections could be found between blood gas measurements and for example personal characteristics or received treatment, then new features could be introduced to help make blood gas parameter prediction more precise.

8 Summary

At the beginning of this report, a brief overview was provided about the challenges in heart transplantation to show the importance of continuously searching for methods that can help making predictions about patients' condition more accurate. How data mining can help in clinical decision making was presented by reviewing some past applications of machine learning and statistical methods related to heart transplantation and blood gas parameter analysis.

The blood test data mining started with the explanation of blood gas parameters along with other variables in the data set, and by creating a clean and aggregated dataset. The final dataset used for modelling was created by handling missing values, filtering the dataset, transforming, and interpolating the values for a certain time interval. Before modelling, the dataset was examined through the Exploratory data analysis. The data was visualized with charts to gain different insights, such as the survival ratio of patients, the differences in the blood gas parameter values between survived and dead patients, or the percentage of patients with values outside of reference intervals. Correlation analysis among parameters was conducted and the non-linear relationships were examined as well, using the Predictive Power Score. Based on the results, there are some significantly correlating parameter combinations and there are some good predictors for each parameter among the others. Traditional and functional principal component analysis were proposed for discovering clusters in the data. According to these, there are some clustering tendencies, but no clear clusters present in the data.

The EDA was followed by the application of linear regression, tree-based, time series and neural network models. As most of the data did not have stationarity, which is a criterion for multivariate time series models, time-series models were finally not considered for use. The regression and neural network models were applied on every patient's data and according to the baseline model evaluation, the *Multivariate1* and *Parallel* models made quite accurate predictions. Based on the score comparison and considering which problems are more interesting, 3 questions with different MLP models were chosen for further investigation. The main results showed that parameters can be predicted from each other with average RMSEs between 0.0015-0.0025 and average MAPEs under 1%. The feature importance examination showed that although some

parameters are a lot more important than others, none of them should be left out. Additional conclusions include that it is possible to predict blood gas parameters further in time with similarly low average errors, while using more input time steps not always decreases the error. Also, it proved to be possible to make somewhat good predictions for unknown patients using data only from others, although the RMSEs were higher in this case. For some parameters the errors were lower than for others, meaning that in some cases it was easier for the model to find and learn patterns in the dynamics of different people. Another promising result is that the number of average false normal predictions this model would make was close to 0 in case of all parameters. This means that the model would rarely make predictions that are misleading in terms of reference intervals, so in most cases it could correctly draw attention to blood gas values that are reaching an abnormal value.

Using these predictive models, ABG tests could be performed less frequently. As ABG test is one of the most performed tests in ICU, costs could be seriously reduced, and the limited resources of ICUs could be managed more efficiently. Furthermore, clinicians could gain insights to expected trends and might be able to prevent life-threatening conditions too.

References

[1] Hunt, S. A., Haddad F.: The Changing Face of Heart Transplantation, Journal of the American College of Cardiology, Vol. 52, Issue 8, 2008, pp587-598, ISSN 0735-1097, date of access: 2022.11, https://www.sciencedirect.com/science/article/pii/S0735109708019347

[2] Medved, D., Ohlsson, M., Höglund, P. et al.: Improving prediction of heart transplantation outcome using deep learning techniques. Sci Rep 8, 3613, 2018, date of access: 2022.12, https://www.researchgate.net/publication/323399852_Improving_prediction_of_heart_transplantation_outcome_using_deep_learning_techniques

[3] Tonsho, M., Michel, S., Ahmed, Z., Alessandrini, A., Madsen, J. C.: Heart Transplantation: Challenges Facing the Field, Cold Spring Harb Perspect Med, 2014, date of access: 2022.11, http://perspectivesinmedicine.cshlp.org/content/4/5/a015636.full.pdf

[4] Wajs, W., Kruczek, P., Szymański, P., Bukowczan, M., Wais, P., Ochab, M.: "Newborn Arterial Blood Gas Prediction," 11th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), 2018, pp. 1-4, date of access: 2022.11, https://ieeexplore.ieee.org/document/8633015

[5] Wajs, W., Wais, P., Ochab, M., Wojtowicz, H.: Arterial Blood Gases Forecast Optimization by Artificial Neural Network Method. In: Piętka, E., Badura, P., Kawa, J., Wieclawek, W. (eds) Information Technologies in Medicine, 2016. Advances in Intelligent Systems and Computing, vol 471. Springer, Cham., pp433-444, date of access: 2022.11, https://link.springer.com/chapter/10.1007/978-3-319-39796-2_36

[6] Wernly, B., Mamandipoor, B., Baldia, P., Jung, C., Osmani, V.: Machine learning predicts mortality in septic patients using only routinely available ABG variables: a multi-centre evaluation, International Journal of Medical Informatics,

vol. 145, 2021, date of access: 2022.11, https://www.sciencedirect.com/science/article/pii/S138650562030976X

[7] Cismondi, F., Celi, L.A., Fialho, A.S., Vieira, S.M., Reti, S.R., Sousa, J.M.C., Finkelstein, S.N.: Reducing unnecessary lab testing in the ICU with artificial intelligence, International Journal of Medical Informatics, Vol. 82, Issue 5, pp345-358, 2013, date of access: 2022.11, https://www.sciencedirect.com/science/article/pii/S1386505612002420

[8] Han, J.W., Kamber, M. and Pei, J.: Data Mining Concepts and Techniques. 3rd Edition, Morgan Kaufmann Publishers, 2012, Waltham

[9] Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R.: Advances in Knowledge Discovery and Data Mining. International conference on knowledge discovery and data mining, 1996, Portland, date of access: 2022.11, https://www.aaai.org/Papers/KDD/1996/KDD96-014.pdf

[10] Fayyad, U., Ramasamy, U.: Evolving data mining into solution of insights. Communications of the ACM, Vol 45., No 8., 2002, pp. 28-31., date of access: 2022.11, https://sceweb.uhcl.edu/boetticher/ML_DataMining/p28-fayyad.pdf

[11] Wirth, R., Hipp, J.: CRISP-DM: Towards a Standard Process Model for Data Mining, 2000, date of access: 2022.11, http://www.cs.unibo.it/~danilo.montesi/CBD/Beatriz/10.1.1.198.5133.pdf

[12] Jackson, J.: - Data Mining; A Conceptual Overview. Communications of the Association for Information Systems, Vol. 8, Article 19, 2002, date of access: 2022.11, https://aisel.aisnet.org/cais/vol8/iss1/19/

[13] Chatfield, C.: The Analysis of Time Series: An Introduction. 5th edition, Chapman & Hall/crc, 1995

[14] Shetty, C.: Time Series Models, 2020, date of access: 2022.11, https://towardsdatascience.com/time-series-models-d9266f8ac7b0

[15] Smola, A., Vishwanathan, S.V.: Introduction to Machine Learning, 2017, date of access: 2022.11, https://alex.smola.org/drafts/thebook.pdf

[16] Hilt, D. E., Seegrist D. W.: RIDGE: A COMPUTER PROGRAM FOR CALCULATING RIDGE REGRESSION ESTIMATES, Research Note NE-236. Upper Darby, PA: U.S. Department of Agriculture, Forest Service, Northeastern Forest Experiment Station, Vol. 236, 1977, date of access: 2022.11, https://www.nrs.fs.usda.gov/pubs/rn/rn_ne236.pdf

[17] Tutorialspoint: Scikit Learn - Bayesian Ridge Regression, date of access: 2022.12, https://www.tutorialspoint.com/scikit_learn/scikit_learn_bayesian_ridge_regression.htm

[18] Scikit-learn: ExtraTreesRegressor, date of access: 2022.12, https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesRegressor.html

[19] Prasad, A.: Regression Trees | Decision Tree for Regression | Machine Learning, 2021, date of access: 2022.11, https://medium.com/analytics-vidhya/regression-trees-decision-tree-for-regression-machine-learning-e4d7525d8047

[20] Haykin S.: Neural Networks: a comprehensive foundation, 2nd Edition, ISBN 8120323734, 9788120323735, Prentice Hall, 1999

[21] Keras.io: Keras, https://keras.io/

[22] Thakur, A., Konde, A.: Fundamentals of Neural Networks, International Journal for Research in Applied Science & Engineering Technology, 2021, date of access: 2022.11, https://www.researchgate.net/publication/353827517_Fundamentals_of_Neural_Networks

[23] Yang, L., Shami, A. (2020) - On hyperparameter optimization of machine learning algorithms: Theory and practice. In : Neurocomputing, Volume 415, pp295-316, date of access: 2022.12, https://www.sciencedirect.com/science/article/pii/S0925231220311693

[24] Hutter, F., Kotthoff, L., Vanschoren, J. (2019) - Automated Machine Learning. Springer, pp3-10

[25] Yu, T., Zhu, H. (2020) - Hyper-Parameter Optimization: A Review of Algorithms and Applications, date of access: 2022.12, https://www.researchgate.net/publication/339898538_Hyper-Parameter_Optimization_A_Review_of_Algorithms_and_Applications

[26] Van Rossum, G. & Drake Jr, F.L: Python reference manual, 1995, Centrum voor Wiskunde en Informatica Amsterdam.

[27] Jupyter.org: Jupyter Notebook, https://jupyter.org/

[28] McKinney, W. & others: Data structures for statistical computing in python. 2010, In Proceedings of the 9th Python in Science Conference. pp. 51–56., https://pandas.pydata.org/

[29] Harris, C.R. et al.: Array programming with NumPy. 2020, Nature, 585, pp.357–362., https://numpy.org/

[30] Pauli Virtanen et.al. and SciPy 1.0 Contributors: SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods, 2020, 17(3), 261-272., https://scipy.org/

[31] Hunter, J.D.: Matplotlib: A 2D graphics environment, 2007, Computing in science & engineering, 9(3), pp.90–95., https://matplotlib.org/

[32] Waskom, M. et al.: mwaskom/seaborn: v0.8.1 (September 2017), Zenodo. 2017, Available at: https://doi.org/10.5281/zenodo.883859

[33] Pycaret.org: Pycaret, https://pycaret.org/

[34] Pedregosa, F. et al.: Scikit-learn: Machine learning in Python. 2011, Journal of machine learning research, 12(Oct), pp.2825–2830. https://scikit-learn.org/stable/

[35] Pypi.org: pmdarima 2.0.3, https://pypi.org/project/pmdarima/

[36] Pyclustertend: Welcome to pyclustertend's documentation!, date of access: 2022.12, https://pyclustertend.readthedocs.io/en/latest/

[37] Zhou, Y., Chen, S., Rao, Z., Yang, D., Liu, X., Dong, N., Li, F.: Prediction of 1-year mortality after heart transplantation using machine learning approaches: A

single-center study from China, International Journal of Cardiology, vol. 339, pp21-27, 2021, date of access: 2022.11, https://www.sciencedirect.com/science/article/pii/S0167527321011694

[38] Mohacsi, P., Pedrazzinia, G., Tanner, H., Tschanz, H.U., Hullin, R., Carrel, T.: Lactic acidosis following heart transplantation: a common phenomenon? Eur J Heart Fail, 2002, date of access: 2022.12, https://pubmed.ncbi.nlm.nih.gov/11959046/

[39] Braith, R. W.: Blood Gas Dynamics at the Onset of Exercise in Heart Transplant Recipients. CHEST Journal 103.6, 1993, date of access: 2022.12, https://www.academia.edu/50139533/Blood_gas_dynamics_at_the_onset_of_exercise_in_heart_transplant_recipients

[40] Docs.scipy.org: scipy.interpolate.Akima1DInterpolator , https://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.Akima1DInterpolator.html

[41] Akima, H.: A New Method of Interpolation and Smooth Curve Fitting Based on Local Procedures, J.ACM, vol. 17, no. 4, pp. 589-602, 1970

https://dl.acm.org/doi/pdf/10.1145/321607.321609

[42] Behrens, J.T.: Principles and procedures of exploratory data analysis. Psychological Methods, Volume 2, Issue 2, 1997, pp131-160., date of access: 2022.12, https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.362.8937&rep=rep1&type=pdf

[43] Jolliffe, I.T.: Principal Component Analysis. 2nd Edition, Springer, New York, 2002

[44] Bescond P-L.: Beyond “classic” PCA: Functional Principal Components Analysis (FPCA) applied to Time-Series with Python, 2020, date of access: 2022.12, https://towardsdatascience.com/beyond-classic-pca-functional-principal-components-analysis-fpca-applied-to-time-series-with-python-914c058f47a0

[45] Wetschoreck, F.: RIP correlation. Introducing the Predictive Power Score, 2020, date of access: 2022.12, https://towardsdatascience.com/rip-correlation-introducing-the-predictive-power-score-3d90808b9598

[46] Alkaline-ml: pmdarima.arima.auto_arima, date of access: 2022.12, https://alkaline-ml.com/pmdarima/modules/generated/pmdarima.arima.auto_arima.html

[47] Statsmodels: statsmodels.tsa.stattools.adfuller, date of access: 2022.12, https://www.statsmodels.org/dev/generated/statsmodels.tsa.stattools.adfuller.html

[48] Scikit-learn.org: sklearn.preprocessing.MinMaxScaler, https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html

[49] Keras: Layer weight initializers, date of access: 2022.12, https://keras.io/api/layers/initializers/

[50] Trevisan, V.: Using SHAP Values to Explain How Your Machine Learning Model Works, Towards Data Science, 2022 , date of access: 2022.12, https://towardsdatascience.com/using-shap-values-to-explain-how-your-machine-learning-model-works-732b3f40e137

List of figures

List of tables