Michael Sams, BSc

Visualization and Analysis of Telehealth Data for the Predictive Analytics Toolset for Healthcare (PATH)

MASTER THESIS

to achieve the university degree of
Diplom-Ingenieur

Master's degree programme: Biomedical Engineering

submitted to

Graz University of Technology

Supervisor

Priv.-Doz. Dipl.-Ing. Dr.techn. Günter Schreier, MSc

Institute of Neural Engineering

Graz, April 2019

This master thesis has been conducted
in cooperation with:

AIT Austrian Institute of Technology GmbH

Center for Health & Bioresources

Digital Health Information Systems

Supervisor

Priv.-Doz. Dipl.-Ing. Dr. techn. Günter Schreier, MSc

Reininghausstraße 13/1

8020 Graz

Austria

EIDESSTATTLICHE ERKLÄRUNG

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Das in TUGRAZonline hochgeladene Textdokument ist mit der vorliegenden Masterarbeit identisch.

STATUTORY DECLARATION

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources. The text document uploaded to TUGRAZonline is identical to the present master‘s thesis.

Graz, 24.04.2019

Datum / Date

Unterschrift / Signature

Danksagung

An dieser Stelle möchte ich all jenen danken, die durch ihre fachliche und persönliche Unterstützung zur Durchführung dieser Diplomarbeit und somit auch zum Abschluss meines Studiums beigetragen haben.

Zuerst gebührt mein Dank Dipl.-Ing. Alphons Eggerth, meinem primären Betreuer am AIT Austrian Institute of Technology GmbH. Er stand mir vom ersten Tag an mit Rat und Tat in allen Belangen zur Seite und hat sich immer sofort Zeit genommen, wenn seine Hilfe benötigt wurde.

Ein besonderer Dank gilt Herrn Priv.-Doz. Dr. Günter Schreier, meinem Betreuer vom Institut für Neurotechnologie der TU Graz und Thematic Coordinator der Arbeitsgruppe ‚Predictive Healthcare Information Systems' am AIT Graz. Er hat für die übergeordnete fachliche Betreuung am AIT und gleichzeitig auch für die Betreuung von universitärer Seite in besonders wertschätzender Weise immer Zeit für meine Anliegen gefunden.

Des Weiteren danke ich allen Mitgliedern des ‚Predicitve Healthcare Information Systems' Teams, sie haben mich sehr gut in ihre Arbeitsgruppe eingebunden.

Darüber hinaus möchte ich mich bei allen Mitarbeitern des AIT am Standort Graz recht herzlich für die Aufnahme als vollwertiges Mitglied in einem hervorragenden Arbeitsumfeld herzlich bedanken.

Dies hat dazu geführt, dass ich im Zuge meiner Diplomarbeit eine außerordentlich bereichernde Zeit beim AIT verbringen durfte.

Abstract

Background: With the ever-increasing amount of available data in the healthcare sector, there is also an increasing interest in computer-aided analysis and predictive models. A comprehensive understanding of obtained models and their predictions is indispensable for the development and later acceptance of such systems.
Objectives: A general concept of a toolset that supports data scientists in the development of predictive models in the telehealth context had to be developed and subsequently implemented.
Methods: Based on surveys, the user requirements were determined. The concept development was based on a test dataset derived from de-identified data from the 'HerzMobil Tirol' telehealth program. The implementation was conducted in MATLAB.
Results: A list of requirements was identified, based on which an interactive viewer concept was developed and implemented.
Conclusion: The developed viewer concept and its implementation facilitate a deeper insight and a better understanding of the development process of predictive models in the telehealth context.

Keywords: Telehealth, Predictive Analytics, Visual Analytics, Human-in-the-loop

Zusammenfassung

Hintergrund: Mit der immer größer werdenden Menge an verfügbaren Daten im Gesundheitsbereich steigt gleichzeitig auch der Bedarf an computergestützten Analysemethoden und prädiktiven Modellen. Ein umfassendes Verständnis solcher Modelle und ihrer Vorhersagen ist für die Entwicklung und spätere Akzeptanz derartiger Systeme unerlässlich.
Ziele: Es war ein allgemeines Konzept eines Toolsets zu entwickeln und anschließend zu implementieren, welches Datenwissenschaftler bei der Entwicklung prädiktiver Modelle im telemedizinischen Kontext unterstützt.
Methoden: Es wurden Erhebungen hinsichtlich der Nutzeranforderungen durchgeführt. Die Konzeptentwicklung basierte auf einem Testdatensatz, der aus anonymisierten Daten des Telemedizinprogramms "HerzMobil Tirol" abgeleitet wurde. Die Implementierung wurde mit MATLAB durchgeführt.
Ergebnisse: Es wurde eine Liste von Anforderungen erarbeitet, auf deren Grundlage ein interaktives Viewer-Konzept entwickelt und implementiert wurde.
Schlussfolgerung: Das entwickelte Konzept und seine Implementierung ermöglichen einen tieferen Einblick und ein besseres Verständnis des Entwicklungsprozesses von prädiktiven Modellen im telemedizinischen Kontext.

Schlagwörter: Telemedizin, Prädiktive Modellierung, Visual Analytics, Human-in-the-loop

Contents

List of Abbreviations

AIT Austrian Institute of Technology

CinC Computing in Cardiology

DS4H Data-driven decision support system for health and care

ECG Electrocardiogram

FN False negative

FP False positive

GUI Graphical user interface

GUIDE Graphical user interface development environment

HF Heart Failure

ICT Information and communication technologies

ID Identifier

KIT Keep In Touch

NFC Near Field Communication

PATH Predictive Analytics Toolset for Healthcare

TGKK Tiroler Gebietskrankenkasse

TUG Timed Up-and-Go

TN True negative

TP True positive

UI User interface

1. Introduction

New technologies and increasing digitisation are leading to huge amounts of data being available in the health and care sector. Affordable and portable measuring devices also enable continuous monitoring of patients at home in telehealth settings. In order to really benefit from all this data, computer-aided processing methods are increasingly applied, especially machine learning approaches are currently considered in many healthcare areas. While technological progress offers promising possibilities, fully automatic analysis and modelling approaches are prone to be incomprehensive solutions. To date, human involvement is still an essential part in the modelling process. On the one hand, specialists need to contribute their knowledge and skills during the development process of predictive models. On the other hand, presentation of modelling results needs to be comprehensible and understandable for humans in order to achieve legitimacy and acceptance of such systems. Therefore, interactive visual analytics tools are needed to fuse human intelligence with computational processing power to achieve optimum results.

At the Austrian Institute of Technology (AIT), the 'Predictive Healthcare Information Systems' team is carrying out research on decision support systems and predictive analytics in the healthcare domain. For this purpose, a software package, called 'Predictive Analytics Toolset for Healthcare' (PATH) is being developed. A current development focus thereby lies on tools that support result visualization, interpretation and validation.

The aim of this thesis was to develop and implement an interactive visualization tool that supports data scientists in the course of predictive model development and optimization within the telehealth domain. To this end, it was necessary to find out, which design and functionality requirements such a tool had in order to support the workflow of data scientists and to cope with the special characteristics of telehealth data. Based on these requirements, a viewer was designed and subsequently implemented in MATLAB.

Although there is still potential for further development, the current concept and its concrete implementation play an essential role in gaining a deeper understanding of the model development process. Thus, the developed viewer makes an important contribution on the way to future more data-driven telehealth systems.

1.1. Background

1.1.1. Telehealth

According to the World Health Organization, telehealth involves the use of information and communication technologies (ICT) 'to deliver healthcare outside of traditional health-care facilities'. [1] In general, the term covers thereby a wide range of application areas including long-distance clinical healthcare, patient and professional health-related education as well as public health and health administration. [2] In the context of this thesis, the term telehealth is used in the scope of home healthcare, where ICT technologies can be utilized to remotely monitor, diagnose and treat patients such as elderly or chronically ill.

Mobile communication devices enable a two-way communication where the patient is at home and a health professional is at a distant site. Portable measuring devices can be used by the patient to collect and send data to health professionals in order to support continuous monitoring of health-related parameters. Telehealth in general may thereby include synchronous interactions as well as asynchronous interactions by using store and forward technologies. [3] Through improved self-care and support services, telehealth also helps patients in better coping with their own disease by enhancing patient self-management.

In a time when the healthcare system is confronted with an increasing number of elderly as well as chronically ill people, telehealth applications offer a great deal of possibilities to improve healthcare. [4]

1.1.2. HerzMobil Tirol

A specific application of telehealth is the heart failure (HF) disease management program HerzMobil Tirol. It is a joint project of the province of Tyrol, Tiroler Gebietskrankenkasse (TGKK), Tirol Kliniken and AIT, which was established in 2012. HerzMobil is a collaborative post-discharge disease management program for HF patients, with a telemedical monitoring system that includes physician-controlled telemonitoring and nurse-led care in a multidisciplinary network approach.

After being discharged from hospital the patients are provided with a Near Field Communication (NFC)-enabled blood pressure and heart rate monitor and a weight scale. In the first three to six months after discharge, the patients daily transmit data on blood pressure, heart rate, body weight, well-being and medication to a health data center via a NFC-enabled smartphone.

The HerzMobil Tirol network thereby realizes a closed-loop healthcare approach, where the cycle of HF management starts with the collection of data followed by interpretation and subsequent adjustments of the treatment as well as impact monitoring.

The interconnection of all involved stakeholders is ensured by the so-called 'Keep In Touch' (KIT) solution (see Figure 1). The KIT telehealth system enables a temporally and spatially independent exchange of information between patients and those involved in the treatment. [5]

Figure 1: The Keep In Touch (KIT) telehealth solution is a patient centered collaborative network including all health and care stakeholders [6]

Within this system, patients can transmit their self-measurement values from the measuring devices to the data center via special smartphone apps. The processed and visualized data is provided to stakeholders and following the closed-loop healthcare principle, health professionals can write feedback and recommendations which are transmitted to the patient's smartphone. This close interconnection of all the stakeholders leads to an improvement in communication and therefore also in terms of integrated care. The program also includes patient training, to strengthen patients' self-competence and to ensure the sustainability of the program. The continuous monitoring makes it possible to detect an upcoming deterioration at an early stage and to enable early intervention. Prompt and continuous therapy optimization furthermore ensures a long-term stabilization of the disease and thus can lead to a reduction of hospital readmission. [7]

In order to support health professionals in their workflow, computer-aided processing methods have already been integrated into the system. Signal processing and specific analyzing

algorithms already support early identification of upcoming adverse events. Automatic event detection in terms of missing values and off-limit measurements, highlights the need for therapeutic decisions and facilitates optimized allocation of resources to those patients who might need imminent support. [6][8]

In the future, an extension of these computer-assisted methods will be necessary in order to handle the amount of data even more effectively and efficiently. On the way to an autonomous telehealth system, predictive analytics solutions will also play an increasingly important role.

1.1.3. Predictive Analytics

The term predictive analytics encompasses a variety of statistical and analytical techniques that are applied to retrospective data in order to develop models that predict certain events or behavior in the future. Technological progress and ever new technical possibilities are leading to more and more data being collected and stored nowadays. Finding ways to really use all this data in a meaningful way is becoming more and more challenging. In this context, predictive analytics plays an important role. With the growing amount of data being available, the relevance of predictive analytics increases in many industries (e.g. insurance, financial services, marketing etc.). [9] Also, in the health and care sector the amount of recorded and stored data is increasing rapidly. Therefore, the importance of predictive solutions, also with regard to the previously described telehealth possibilities, will increase strongly in this domain. [10]

Predictive analytics is a multidisciplinary field, involving various techniques like data mining, machine learning and predictive modelling. Data mining is the process of identifying underlying trends, patterns, or relationships in the examined data. It is an essential step in predictive analytics, because the data that is singled out by the data mining process is subsequently used to develop predictive models. [11]

Machine learning basically involves computer algorithms that learn from experiences (i.e. retrospective data). The performance of these algorithms increases with the amount of data they are provided with for learning. Machine learning can be divided into supervised and unsupervised learning (see Figure 2).

An algorithm of supervised learning uses both a known amount of input data and known outputs associated with this data. The model trained with this data is then used to predict the outcome on the basis of new input data. Supervised learning is therefore used in cases where information about the outcomes is available. In supervised learning, classification and regression techniques are used to develop predictive models.

Classification techniques (e.g. Support Vector Machine, Naive Bayes, Nearest Neighbor, etc.) predict discrete outputs by classifying input data into categories. With regression techniques (e.g. Linear Regression, Decision Trees, Neural Networks, etc.) continuous outputs can be predicted.

Compared to supervised learning, unsupervised learning is only based on known input data. These algorithms are used to search for hidden patterns or internal structures in the available input data. The most common technique of unsupervised learning is clustering, e.g. K-Means, Gaussian Mixture, Hidden Markov Model. With the help of clustering techniques, data points are grouped together depending on their similarity. [12][13]

In the context of this thesis, the term predictive modelling is used to refer primarily to supervised machine learning techniques. Which means there is both input and output data available.

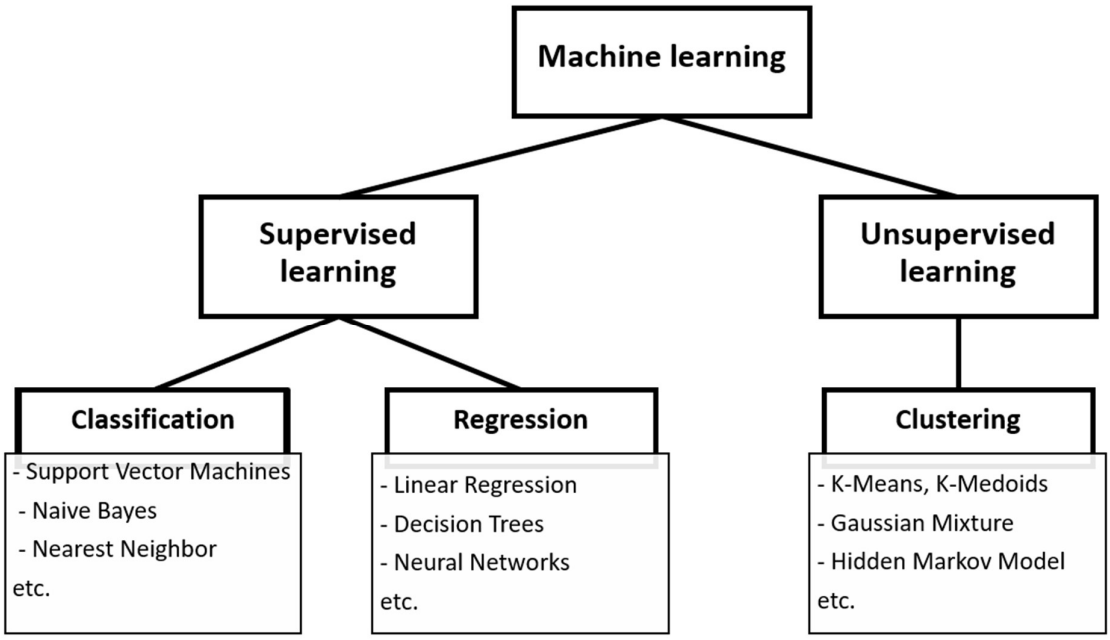


Figure 2: Machine learning techniques [12]

These machine learning techniques make it possible to develop predictive models using the ever-increasing amount of data. With the progressive use of these solutions in real-world scenarios, however, it is becoming increasingly important to develop more comprehensive and reliable models. In the context of these development and optimization processes, another field is becoming increasingly important, namely visual analytics.

1.1.4. Visual Analytics

An early definition of the term visual analytics can be found in the research and development agenda, Illuminating the Path [14], where it is described as 'the science of analytical reasoning facilitated by interactive visual interfaces'. A more recent and specific definition can be found in [15], which says 'Visual analytics combines automated analysis techniques with interactive visualizations for an effective understanding, reasoning and decision making on the basis of very large and complex datasets'.

So, the key aspect is again the large amount of data, because as already mentioned, the acquisition of raw data is no longer the main problem. Instead, the main challenge is to find appropriate methods and models to transform this data into reliable and comprehensive knowledge. Because the raw data itself has no value, only the information derived from it brings a real added value. For this it is necessary to have effective methods to exploit the potential that lies in the data. [15]

A key aspect of visual analytics is to facilitate analytical reasoning by building on human capabilities to visually process and understand complex information. This requires appropriate visualizations that show the most valuable and relevant information to help the analyst better understand the data, discover hidden relationships, and thereby gather important insights. [16]

In this process of information retrieval, computer-aided data processing also plays an important role. Methods from knowledge discovery in databases (KDD), statistics and mathematics together with the increasing computational capacities offer a wide range of possibilities for automatic data analysis. [17]

In many cases, however, purely automated processing is not a sufficient solution. Fully automated methods are only appropriate for well-defined and well-understood problems. But the data analysis process and the way from raw data to sophisticated decisions is usually rather complex with several degrees of freedom. The more complex the problem, the more necessary is a collaborative effort of humans and computers. [16] In this context, human background knowledge, intuition and creativity are of crucial importance and can neither be automated nor replaced during the optimization of automatic processing procedures. After all, it is essential to integrate expert knowledge into these systems. [17]

One important aspect is to present the results of the analysis processes in such a way that they are easy to understand. Since subsequent decisions are made on the basis of these results, it is also particularly important to be able to understand and explain more precisely the processes that led to these results. With the help of visual analytics, the way in which data is processed

and analyzed should become more transparent. The visualizations therefore go beyond a simple presentation of results and encompass all the processing steps involved. This insight into the individual stages of processing and modelling enables a comprehensive understanding of the situation. But visual analytics is not only about simple visualization. Visual analytics is highly interdisciplinary and is an integral approach which combines various related research areas such as visualization, statistics, data mining, data management, modelling and human factors (see Figure 3). [15]

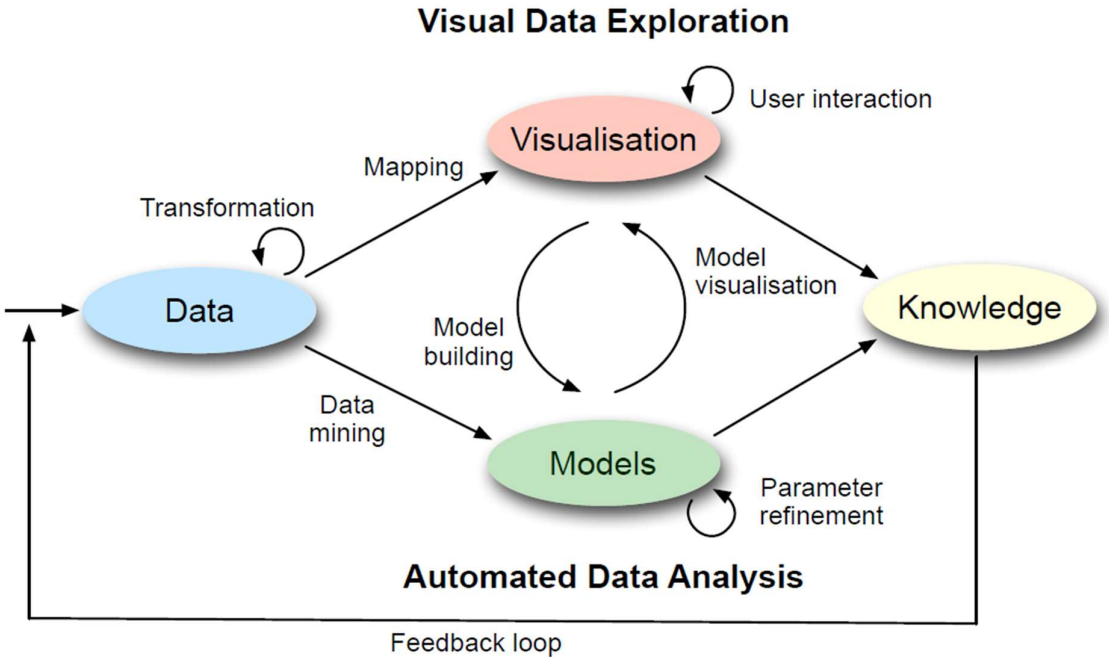


Figure 3: The visual analytics process combines visual data exploration and automated data analysis and includes the interaction between data, visualizations, models and derived knowledge [15]

It is an iterative process that ranges from information gathering to data preprocessing, knowledge representation and interaction down to decision making. The visual representations and interaction functionalities thereby pose a gateway into the data. It is a semi-automated analytical process, where the distinct capabilities of humans and computers are combined in order to reach the most effective results. As the analytic process is not serial, a characteristic is the alternation between visual and automated methods, which leads to a continuous refinement and verification of produced results. [15] [17]

The integration of automatic analysis methods before and after the interactive visual representations is especially important for two reasons. The datasets which are of interest often

are multidimensional and very complex on the one hand and too large to be directly visualized on the other hand. Therefore, D. Keim et al. [17] postulated the following visual analytics mantra:

'Analyse First - Show the Important - Zoom, Filter and Analyse Further - Details on Demand'

These strategies not only help data scientists analyze data and develop models but can also be used for explanatory purposes against third parties. Visual analytics poses therefore a promising approach that has the potential to make an essential contribution in dealing with the challenge of data and information overload. The possible areas of application include, for example the engineering domain, financial analysis, public safety and security as well as environmental and climate related topics. [17][18]

But, of course, visual analytics is also very important in the health and care sector. These methods can help to derive practically relevant insights from raw data and the results of computational models. These insights can subsequently be used to better evaluate treatments and their outcomes and thus improve overall patient care. [19]

1.2. State of the Art

1.2.1. State of the Art in visual analytics

A good overview of the state of the art in predictive visual analytics is given by Y. Lu et al. [20]. The field of application of visual analytics thereby is very broad. In the following, examples which comprehensively support the visual analytics pipeline are presented. These examples show how expert knowledge can be integrated into the analysis process in various application areas by using visual analytics methods.

F. Heimerl et al. [21] use a visual analytics approach for the purpose of machine learning based text document classification. An interactive desktop is presented that facilitates visual classifier training. Their classifier training system thereby includes document preprocessing, classification feature engineering, visual analytics supported active learning and result analysis.

In [22], J. Choo et al. present an interactive visual analytics system for classification, based on a supervised dimension reduction method, linear discriminant analysis (LDA). A case study is conducted on the basis of facial image data for the purpose of facial recognition. The system supports data encoding as well as other preprocessing steps, visualization of the reduced dimensions and cluster structures resulting from the analysis process.

Another visual analytics application that allows domain experts to incorporate their knowledge into the analysis process is the scatter/gather clustering approach of M. Hossain et al. [23]. An interactive visual interface enables users to iteratively restructure clustering results in order to meet their expectations.

An application of visual analytics in the healthcare domain is described in C. Stolper et al. [24]. Besides a concept of progressive visual analytics, a tool for pattern analysis in a collection of event sequences is described. The tool, which is depicted in Figure 4, can be utilized for the identification of medical patterns based on the information of electronic medical records. The goal of the specific use case was to determine if certain sequences of surgical events correlate with health outcomes of patients. Besides of the investigation of how certain surgeries correlate with readmissions, the tool was also applied to the research question of how certain patterns correlate with gender, age, and length of hospital visit. The findings can subsequently be used for the revision or enhancement of surgical guidelines.

The main interface of the tool includes a scatterplot view, two list views, a tree view and two panels for information display. In the scatterplot view the patterns are visualized. The patterns are thereby displayed depending on their respective rank in the form of colored circles or a heat map. The list view is used to list the patterns that were detected by algorithms and are sorted by selected ranking measures. The two instances of the list views enable the comparison of two different ranking methods. The tree view shows a hierarchical representation of each pattern. Furthermore, a number of interactive functionalities are available that allow the user to intervene in the analysis process. [24]

The tool with its elements and functionalities is designed to support the concept of progressive visual analytics. The basic idea of progressive visual analytics is that partial results are already produced during the execution of the analytical process. These progressive results are then integrated into interactive visualizations that allow data scientists to immediately explore the partial results, examine new results as soon as they are produced, and manipulate certain parameters while the process is still running.

Prerequisite for this is that the analytical algorithms have to be designed in a way that they produce meaningful partial results during execution on the one hand. On the other hand, the visualization procedures must be designed in a way that they enable the integration of these partial results as they are produced without distracting analysts by constant updates. This approach enables optimization measures already during the execution of the analysis process, which leads to an improvement and time saving in the workflow of data scientists. [24]

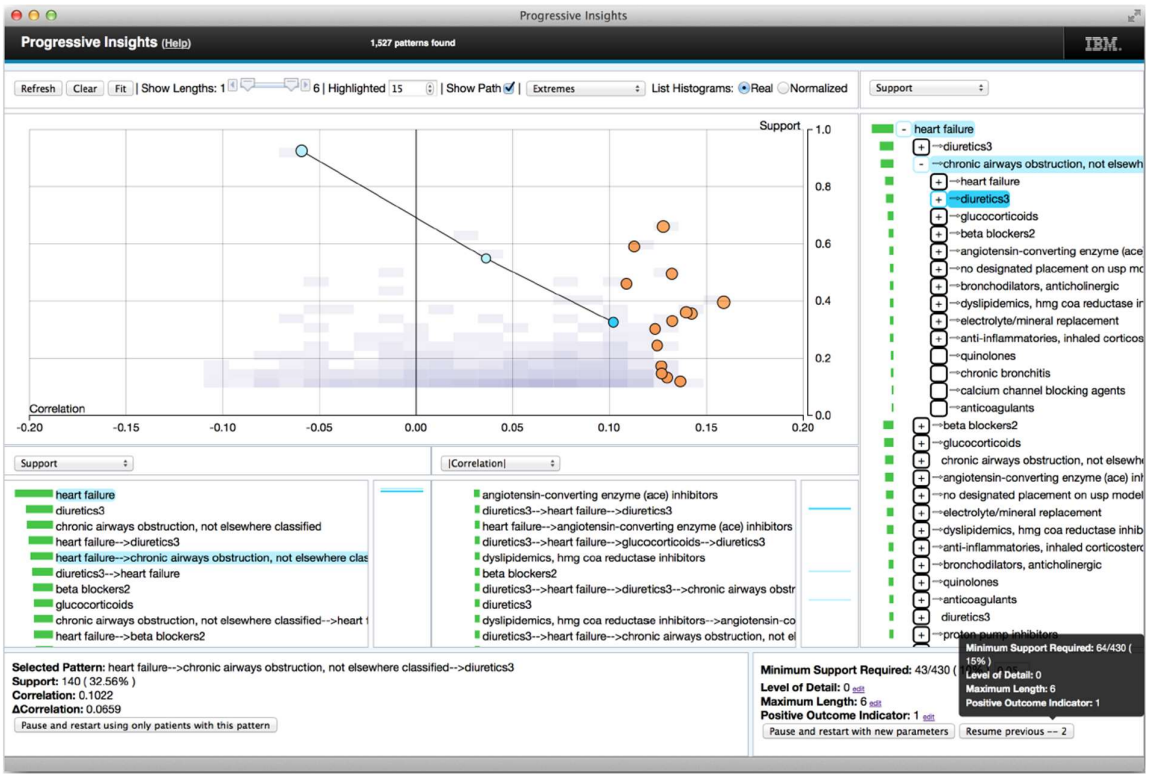


Figure 4: Interface of the progressive visual analytics tool 'Progressive Insight' [24]

1.2.2. Development and implementation of a predictive model

At the AIT, the 'Predictive Healthcare Information Systems' team is carrying out research on decision support systems and predictive modelling in the healthcare domain. Among other activities, they are dealing with the question of how a data-driven decision support system for health and care (DS4H) should optimally be designed and implemented. In [25], D. Hayn et al. describe healthcare specific requirements for such a DS4H.

They postulate, that the development of a DS4H should follow a two-level process with two continuously repeating cycles, which are depicted in Figure 5. Cycle 1 is the actual modelling process, including data cleaning & pre-processing, feature engineering, model training, evaluation and visualization, interpretation & validation. Cycle 2 is intended to support the surrounding processes, which are present in a real-world scenario, i.e. objective definition, data collection & de-identification as well as deployment. This outer cycle 2 needs to be repeated in regular, but significantly longer intervals of time compared to cycle 1.

To implement the process depicted in Figure 5, a software package, called 'Predictive Analytics Toolset for Healthcare' (PATH) is currently developed. PATH is a MATLAB (The MathWorks, Natick, US) based predictive modelling toolset. The fundamental idea is to come from retrospective data to prospective predictions. PATH thereby supports the implementation of various scenarios, including regression models, binary and multiple classifications.

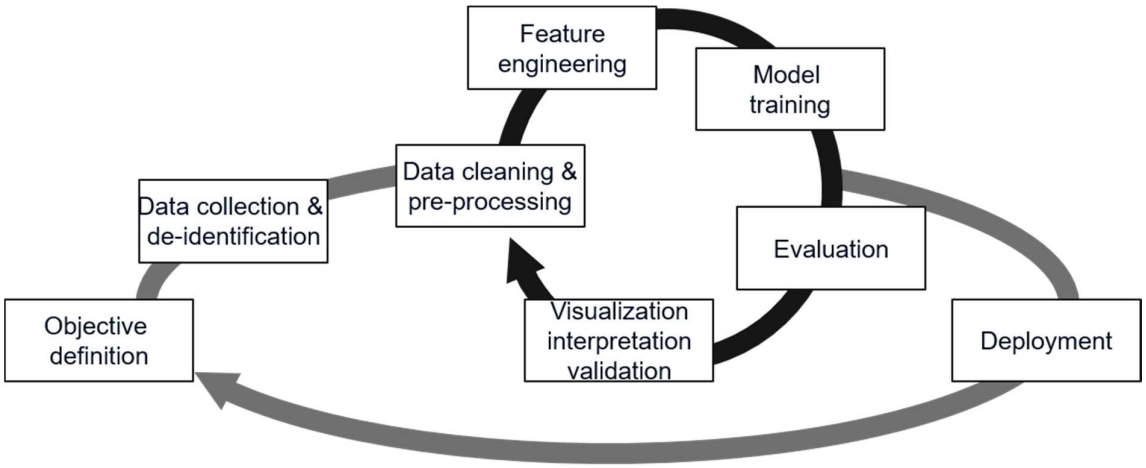


Figure 5: Two-level process of data driven decision support in health and care. [25]

The specification of the models, i.e. feature settings and all other adjustable parameters for each step of the modelling process is performed with the help of three Microsoft Excel files (i.e. Source Definition-, Feature Set Definition- and Modelling Definition file). The actual processing is conducted in MATLAB.

The predictive analytics toolset supports various types of data and various relations in between data objects. Within the Excel Source Definition file, the different data types and their relation to one another (1:1, 1:N, N:M, etc.) are specified. This specification file is also used to specify data cleaning procedures (e.g. outlier removal) which are then performed in MATLAB. Signal processing algorithms (e.g. electrocardiogram (ECG) biosignal processing) can also be specified within this Excel settings file.

A key aspect in the modelling process are the so-called 'features'. These are specific attributes or properties of the data and are essential to the predictive model. Features can be measured values themselves, but features can for example also be derived from other parameters (e.g. the Body Mass Index (BMI) can be derived from body weight and height). The individual features are combined to a feature set, which corresponds to the learning dataset of the model. How this feature set is composed is specified in the Feature Set Definition file.

With the Modelling Definition Excel file, a variety of different models can be specified based on the predefined feature sets, including Decision Trees, Random Forests, Support Vector Machines, Linear Regression Models and Logistic Regression.
Within this Excel file also sets of methods for model evaluations, which should be performed, can be specified. The toolset includes various statistical tools, including box plots, scatter plots, confusion matrices, etc. with which the results of different models can easily be compared to one another. PATH also ensures reproducibility of previous model results due to automated storage of models, results and specifications.
PATH has already been applied to several targets including the prediction of occurrence of delirium in the course of hospital admissions [26], patient blood management including benchmarking [27][28] and healthcare resource utilization based on health insurance claims [29][30][31]. Furthermore, the toolset is used for the prediction of hospital re-admissions and adherence during telemonitoring (not published yet).
Up to now, none of the developed models have been deployed to a real-world scenario. Since PATH is only used within research environments at this point, the software is not certified as a medical device yet. In its current state, PATH is primarily utilized for the inner cycle of the process illustrated in Figure 5. However, the toolset is highly adaptable to meet all the requirements of different healthcare scenarios.
Among other developments, a clear focus lies on gaining a better understanding of the generated models. Therefore, user interfaces (UI) for result visualization, interpretation and validation as well as task-specific interaction functionalities are required to give more insights into data, results and models.

An example of a recent adaptation is PATH's 'ECG viewer', which was developed in the course of participating in the Computing in Cardiology (CinC) challenge 2017. The basic objective of the CinC challenge was to develop algorithms, which were able to automatically detect cardiac anomalies with high accuracy. [32] For this purpose Kropf et al. [33] developed a combined method of classical signal analysis and machine learning.
In order to develop such an algorithm, the data scientists had to iterate through the various involved levels, including raw data (e.g. ECG), preprocessed data (e.g. averaged heartbeat), extracted features (e.g. QT interval), built models (e.g. classify as normal / pathological), evaluation outcome (e.g. a false positive case), and to assess the relevance of different features. [25] To support data scientists in this process, a MATLAB-based software package for ECG viewing was developed. Besides the original raw ECG, the main window of the viewer contains

the intervals and the classification of the detected heart beats, the averaged beats of up to four classes and a table of the features (see Figure 6).

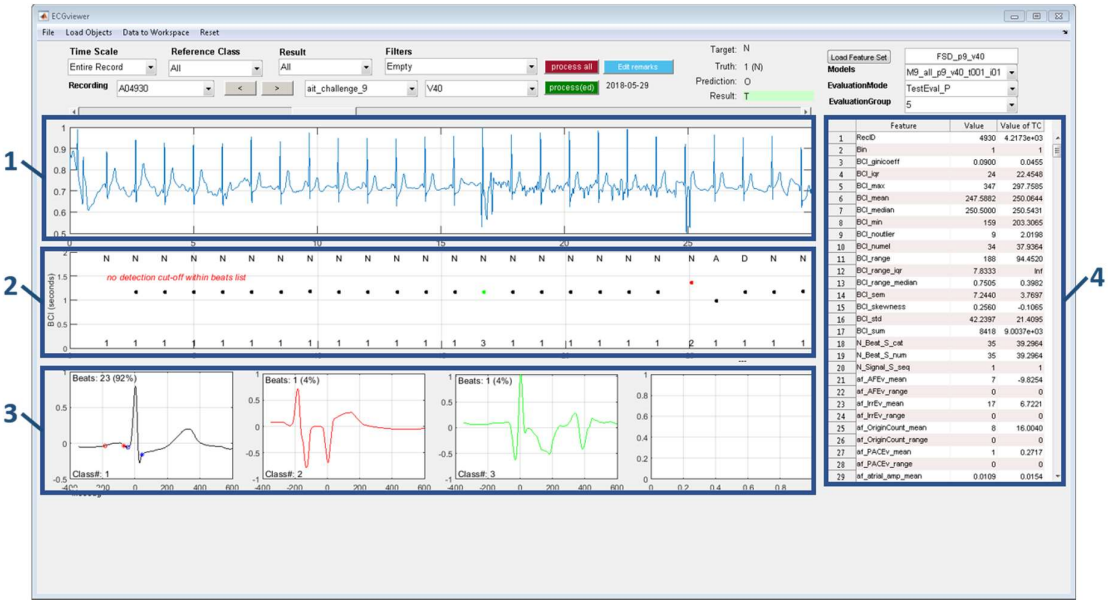


Figure 6: ECG viewer: 1) unfiltered ECG signal, 2) heart beat intervals and classification, 3) averaged beat view for a maximum of four classes, 4) feature table [33]

Within the viewer, quick navigation from one ECG recording to the other is enabled including the most important elements regarding each observation. Thus, this tool plays an essential role in the development of algorithms in the context of ECG analysis. However, although for ECG visualization and classification the ECG viewer extension is highly suitable, PATH lacks tools for supporting the predictive modelling process of other domains, like e.g. the telehealth domain. There is the need for a tool that is adequately designed to the characteristics of telehealth data and the related modelling process. Especially in view of the vision of an autonomous telehealth system, in which predictive solutions play an important role, a tool that supports the development and optimization of such models is required.

1.3. Objectives

The objective of this thesis was to develop and implement a tool that supports data scientists in developing predictive models in the telehealth domain. This tool had to be developed on the basis of the predictive toolset PATH, which was introduced in section 1.2.2. The tool should be integrated into the model development process, which basically corresponds to the inner cycle of Figure 5 (see Figure 7).

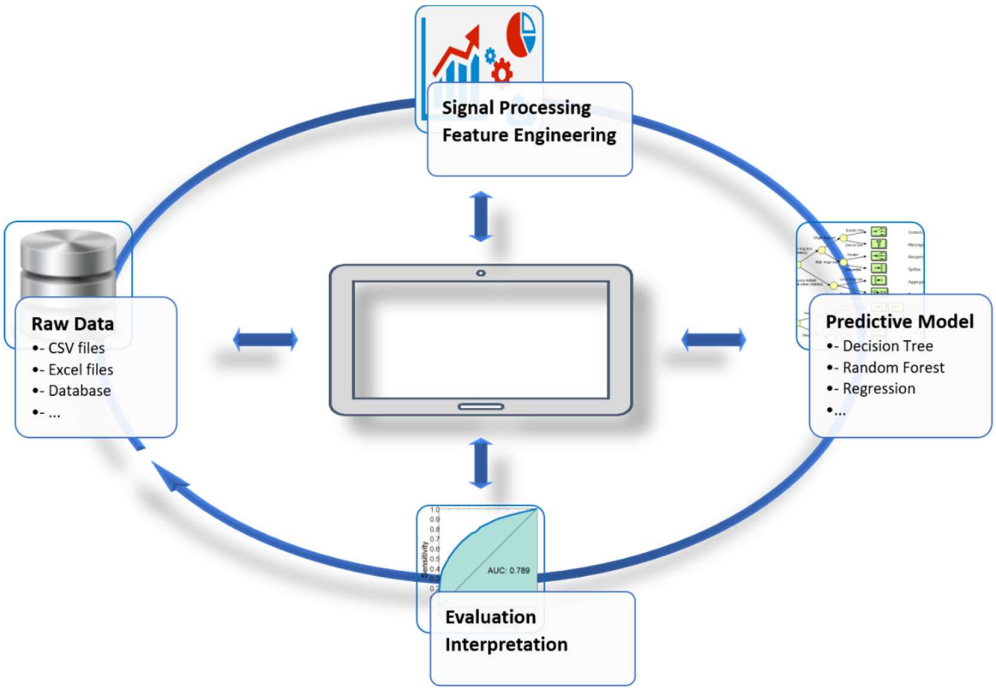


Figure 7: Integration of an interactive viewer into the PATH program

It should be an interactive visualization tool that effectively supports the workflow of data scientists. Thus, the viewer had to cover all important process steps and through comprehensive visualization and interaction functionalities allow the user to gain better insights in the process of model development. The viewer should be developed primarily on the basis of telehealth data. At the same time, however, it had to be taken care of that the developed framework is designed as generic as possible, so that it could also be applied to other use cases. It should also be noted that the viewer is not intended for laymen, but for data scientists who have experience with the underlying tools and processes. It was necessary to find out, which design and functionality requirements such a tool has, in order to support the workflow of data scientists. The implementation based on these requirements had to be carried out with MATLAB in order to achieve the highest possible interoperability with the PATH program.

2. Methods

2.1. Orientation

As a preparation for the concrete realization of the objective, there was a general induction into the areas of predictive analytics and visual analytics. For this purpose, a first literature research was carried out in order to gain a certain understanding and an overview of these topics as a basis for the further implementation of the objective.

Another part of the orientation phase consisted of getting a deeper understanding of the PATH program. In addition to viewing the individual program modules, several workshops were held with the developers to better understand the program processes and structures. Furthermore, the existing ECG visualization tool, which was described in section 1.2.2, was considered for a more detailed analysis of the actual state of development. In the course of this step, not only the existing code was analyzed, but also selective adaptations of the ECG viewer were carried out. These included general update routines of the tool, which were modified. Thereby a special focus was put on the existing program routines and the interfaces between the PATH software and the integrated visualization tool.

2.2. Concept Development

In preparation for the concept development, research on general design guidelines and dashboard design principles was carried out in order to make fundamental considerations in this regard.

It became very clear that one of the first and most critical steps is to determine the right information to include on the interface. A fundamental challenge of dashboard design is thereby to display the most relevant information on a single screen, clearly and without distraction, in a way that critical aspects are quickly revealed. This can be achieved, among other things, by minimizing the number of objects on the screen, keeping graphical icons sparse, displaying context in abbreviated form, and keeping the number of utilized colors to a minimum. The information should thereby be presented intuitively as a combination of textual content and graphics. In order to avoid unnecessary distractions, it is also important to not integrate all the existing information in one window. Here it is important to find a compromise and work out solutions where more detailed information can be obtained on request. [34][35]

However, given that the visualization tool had to be developed and applied to support a specific context, the application specific requirements play an essential role. The development and implementation of the viewer therefore followed a user-centered design process. [36] This means, that the users had a major impact on the design by being deeply involved throughout the whole development process. The user-centered design process thereby included several phases. First of all, it was important to be aware of the context of use. This included the specific consideration of the people who will use the product, what they will use it for, and under which circumstances they will use it. Furthermore, in order to specify the requirements a requirements analysis was conducted which is discussed in more detail in section 2.3. In the course of this analysis, the essential elements and functionalities as well as overall user expectations, which had to be met for the final result, were identified.

The considerations regarding concept and design of the viewer were strongly influenced by the results of the requirements analysis. It needed to be considered how all the elements and functionalities could best be integrated into one tool in order to achieve the greatest possible usability. On this basis, a first concept was worked out. Subsequently, the initial design was continuously adapted and refined as the implementation process advanced.

Finally, the design also had to be evaluated. On the one hand, the design was repeatedly tested in the course of the stepwise overall development. On the other hand, a final evaluation was carried out in the course of the entire validation and test phase with the strong involvement of the users, which is described in more detail in section 2.6.

2.3. Requirements Analysis

The viewer that had to be developed should primarily be adapted to the application in the context of telehealth. In order to achieve broad applicability, however, the framework should be kept as generally applicable as possible. Thus, to get a comprehensive basis for the requirements analysis, the general characteristics of health data were first examined in more detail. Subsequently, this basis was extended and refined by taking a closer look at the specific projects, which currently were in progress at the AIT, in order to get a complete overview of potential data sources. Regarding viewer development, however, the focus was clearly on data from the HF telehealth program HerzMobil Tirol and the corresponding prediction of critical events during the monitoring period. In the course of this investigation, a focus was also on determining the differences compared to the specific use case of ECG classification.

Apart from the nature and characteristics of the underlying data, knowledge about the workflows and distinct steps of the predictive modelling process was a key aspect for the requirements analysis. In addition to the theoretical considerations regarding the predictive modelling process, the specific workflows of the data scientists at the AIT were analyzed in detail and their experiences regarding the development of predictive models were considered. A series of workshops were held, in which the work procedures were explained in more detail. There was also an in-depth exchange about expectations towards such a tool and its capabilities. On the basis of these investigations, the essential elements and functionalities of an interactive visualization tool were identified.

The in-depth exchange with the users, which was of great importance in the initial phase of the work, was thereby not limited to the early stages of the thesis but was rather an important and ongoing component throughout the entire development and implementation process.

2.4. Dataset

The development of the viewer was based on a de-identified telehealth dataset from the HF disease management program HerzMobil Tirol. This test dataset included the records of 137 patients. In addition to demographic data of the patients and measurements taken by the patients at home (i.e. body weight, heart rate and blood pressure), the dataset also included clinical notes of health professionals, information about medication compliance and prescriptions as well as information on hospital admissions.

This data had been extracted by data scientists from the HerzMobil Tirol project, prepared for secondary use and made available in the form of three Excel files. One file contained all the metadata of the patients. The main file contained all measured values, information about medication, well-being as well as existing text notes of the health professionals with the respective time stamps. The third file contained a summary of events and interventions with their respective start and end times. In addition to hospitalization, this also included changes in medication or the patient's absence due to vacation etc. In the case of hospitalizations, a distinction was made between those due to HF and those that had occurred for various other medical reasons.

All data had been de-identified by omitting direct patient identifiers such as first name or last name, and by transforming indirect identifiers (e.g. transforming date of birth to age in years, omitting rare diseases, etc.).

To suit for the development of various functionalities, the test dataset was extended by artificially adding ECG recordings to obtain a further data level. The ECG recordings used for this purpose were taken from the publicly accessible dataset of the CinC 2017 challenge. For the development purpose it was sufficient to add ECGs to the datasets of three selected patients. Therefore, a separate folder structure was added to the raw data directory (see Figure 8).

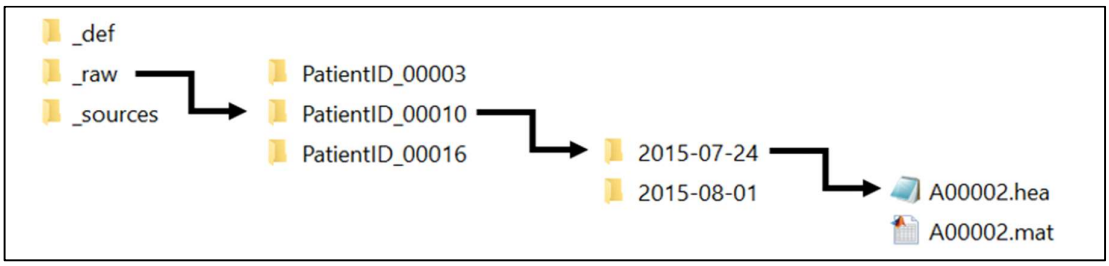


Figure 8: Directory structure of added ECG recordings

The first level of the directory structure shown in Figure 8 contained three folders. The '_def' folder contained the definition files described in point 1.2.2. (i.e. Source Definition-, Feature Set Definition- and Modelling Definition file). The '_sources' directory contained the data from the HerzMobil program (i.e. the above described Excel files). The '_raw' directory was created for the ECG extension, which contained three selected patient directories. The example of patient 10 shows that an ECG was added for two distinct dates. The respective date was selected so that it lied within the monitoring period of the HerzMobil program. The ECG raw signal was provided in the form of a header-file and a mat-file. In the same way, ECG recordings were added to patient 3 and patient 16.

2.5. Implementation

2.5.1. Implementation tool

According to the given requirements, the implementation of the viewer was carried out in MATLAB R2018a (The MathWorks Natick, US). [37] For the development of a graphical user interface (GUI), MATLAB provides two specific tools, i.e. 'GUIDE' (graphical user interface development environment) and 'App Designer'.[38] The initial step was to decide which of these two tools was better suited for the implementation of the viewer. For this purpose, the two options were analyzed in more detail with regard to their suitability for the current problem.

Because even though they are similar and compatible in many ways, each of the approaches offers a different workflow and a slightly different set of functionalities.
GUIDE, which has been available for many releases, is a drag-and-drop environment for laying out UIs. The interactive behavior of the interface is coded separately in the MATLAB editor. Applications created with GUIDE are compatible with almost all other releases. App Designer is the more recent development environment which was introduced in version R2016a. Compared to GUIDE, App Designer provides a larger set of interactive controls, including gauges, lamps, knobs, and switches. With App Designer it is for example also possible to directly create a tab panel, which in GUIDE is only possible with the help of workarounds.
However, as of R2018a, not everything that is possible in GUIDE is also possible in the App Designer. It turned out that App Designer hereby has two major limitations compared to GUIDE. On the one hand this concerns the available plot functionalities. GUIDE supports all the graphics functionality in MATLAB, whereas in App Designer, not all graphics functionality is supported. This concerns among other things various restrictions concerning axes, subplots and heatmaps. And, since a focus of the viewer lied on the visualization of various data with the aim of being as flexible as possible, this posed a decisive limitation.
Furthermore, there is a restriction regarding the availability of interactive functionality. As of R2018a, in App Designer there are limitations concerning mouse and keyboard callbacks for figures and axes. This resulted in the decision to use GUIDE for the implementation of the viewer.

2.5.2. Implementation procedure

The implementation was initially based on the framework of the ECG viewer presented in Section 1.2.2. This framework was subsequently further developed and generalized on the basis of the requirements developed according to the HerzMobil telemonitoring data.
The first phase of the implementation included the establishment of the interfaces of the viewer and the PATH program. Another part of this phase was the implementation of the basic loading procedures in order to have data and other relevant information regarding the modeling process available in the viewer framework. These structures could be built mostly on the basis of the ECG viewer framework, under consideration of certain adaptations.
The second phase of the implementation included the main visualization elements and the associated interactive functionalities. These were gradually implemented in the form of separate work packages. The individual elements were implemented and tested as far as possible before

they were fully integrated into the overall framework. In the course of this phase, other functionalities of the viewer, such as ‘details on demand’ concepts, were also implemented.
The third phase of the implementation comprised the various update procedures of the viewer, in particular the external start-up of the viewer through callbacks from the evaluation.
Basically, the entire implementation was conducted in such a way that, in a first step, the respective elements and functionalities were adapted to the specific requirements of the telehealth data. In a further step, it was then attempted to find a realization that could be applied as generally as possible.

2.6. Validation and Testing

The functionalities and usability of the viewer developed on the basis of the telehealth data were not only tested at the end, but also continuously during the development, in the course of joint workshops with the users. The implemented modules were presented and demonstrated to the users on a regular basis. In the subsequent discussions, adjustments and improvements to the presented tool were discussed.
The validation of the generalized framework was carried out with a dataset containing records of ergometric performance tests conducted by a cohort of cardiac rehabilitation patients. This was a dataset of approximately 1500 patients with about 29.000 performance tests. The files contained parameters such as heart rate, workload curves, ECG data and blood pressure. The performance tests were carried out in different phases of the rehabilitation program. One possible research question, the viewer could be utilized for, would for example be, whether the patients keep a constant performance level over time. In order to answer this question, various visualizations would be required that allowed a more detailed analysis of the data. In Figure 9 a draft of a viewer interface is depicted, which contains the necessary elements for this purpose. This draft contains a time series visualization of the performed ergometric performance tests. By selecting a certain performance test, the heart rate and workload curves over time are plotted as well as the heart rate over the workload and, also, the corresponding ECG is displayed. Data scientists of the AIT used the developed viewer concept to implement this setup.

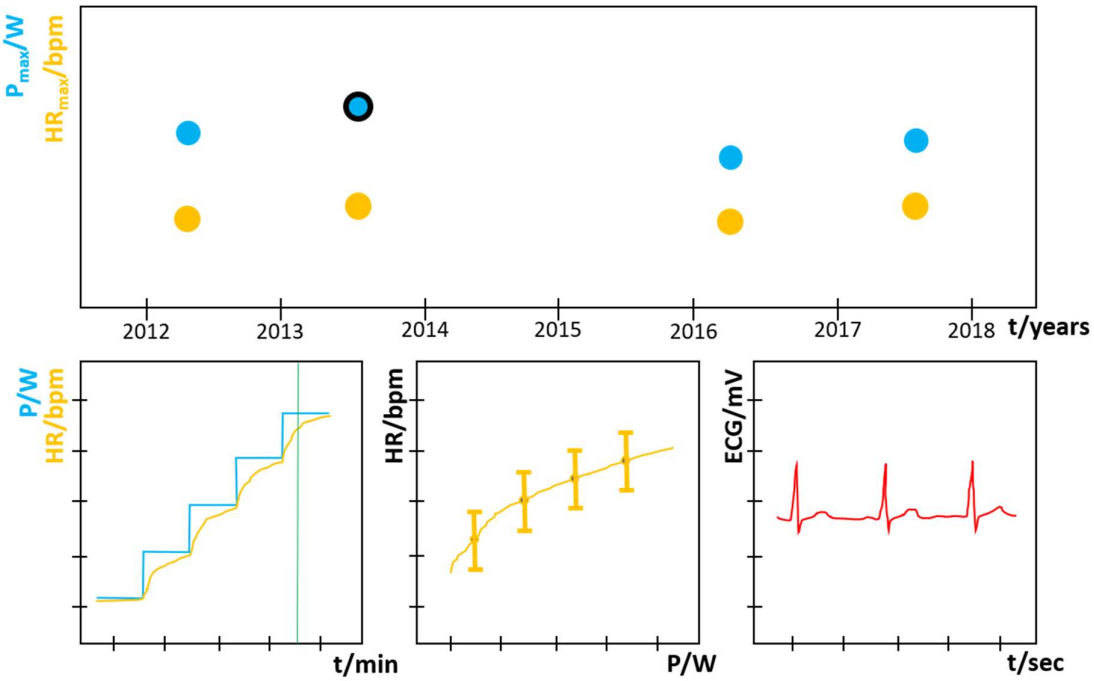


Figure 9: Viewer interface for ergometry data (draft). (source: AIT)

3. Results

3.1. Results of the Requirements Analysis

3.1.1. Data characteristics

Healthcare data in general concerns a very broad field of applications, ranging from large-scale population health data and whole healthcare systems to the genomics of a single patient for example. But even in the narrow field of patient-centered health and care applications, which poses the focus of this thesis, the variety and the amount of data can be enormous. Because healthcare data, such as in Electronic Health Records, cover not only observations about the condition of a distinct patient, but also information about the various treatment measures. Often data across years of a patient's longitudinal medical record have to be considered to understand the evolution of a given medical condition. The data can thereby include all kinds of measurements, various medical events (e.g. diagnoses, procedures, medications, and lab tests) and unstructured clinical notes. In order to assess a patient's health condition, all these data and information need to be analyzed collectively, and the correlations of multiple parameters have to be considered.

Another essential aspect is the temporal characteristic, as a lot of the considered variables are temporal by nature. Medical events are unfolding over time as patient's conditions evolve. Especially temporal patterns are thereby of great interest involving discrete events, interval events or various parameters recorded repeatedly over time. A parameter value at a single point in time is less informative than its evolution over time. Especially the tracking of changes in a patient's condition as a reaction to applied treatments or interventions of health professionals are of great interest. [19][39] These characteristics can be found in most health and care applications and therefore must be taken into account for comprehensive considerations.

When looking at telehealth applications respectively the specific case of the HerzMobil Tirol program, there is also a great variety of available data. This data includes the results of the measurements the patient carries out at home in the course of the telemonitoring program on a daily basis (e.g. body weight, blood pressure or heart rate). Additionally, there might be statements about the well-being of the patient. There might be information on the one hand about the prescribed medication and on the other hand on the actual medication intake of the patient i.e. related to the compliance. Furthermore, there might also be unstructured data like clinical notes or textual communication between the patient and the health professional.

Another important aspect is the different kind of events that can occur during the telemonitoring period as for example a hospitalization, a change in the medication or any other kind of intervention by a health professional, which all can have a significant impact on the patient's condition.

Alongside the data from devices which deliver a distinct value per measurement e.g. weight scale or a blood pressure monitor, there might also be other procedures or measurements during the monitoring period, that do not consist of individual values. For example, biosignals from an ECG recording or gait analysis from a timed up-and-go device. There might also be laboratory results or medical imaging data.

3.1.2. Workflow of data scientists

In the course of the requirements analysis survey regarding the workflow of data scientists, the importance of some aspects was emphasized once again.

Predictive models are very much dependent on the raw input data and the features used to train the model. The features either consist of or are derived from raw data. Therefore, it is very important to know, analyze and prepare the data properly. Poor data or input leads to poor results, regardless of the utilized model approach. Particularly with health data, there are often incomplete data and outliers that can corrupt the result. In addition, the data often comes from different sources and therefore has to be merged to a compact dataset before it can subsequently be presented to the model. Therefore, model development involves a great deal of data analysis and preprocessing.

Another important aspect is, that the human being still has an indispensable role to play in the development of predictive models. The human expertise on the respective subject area poses an essential input that has to be incorporated into the development. Furthermore, human perception and visual capabilities are a major asset in the daily work routine to gain new connections and insights. These advantages that the human factor brings must be integrated and utilized at every step of the development process.

But to really benefit from these capabilities, in order to get a comprehensive understanding and to draw conclusions, a good overview of the available data is required. The investigation of single parameters is not sufficient. The interconnection of multiple parameters have to be observed in order to assess a patient's health condition as well as to draw conclusion regarding the model development. At the same time, however, care must also be taken to ensure that the overview is maintained given the diversity of data.

Another important aspect that must be considered is the actual procedure of the development process. Because the process, which includes the already mentioned steps from raw data to signal processing, feature engineering, modelling and evaluation, is by no means straight forward or a single run. The individual development stages must not be considered as isolated steps. There are many different adjustable parameters along the different stages, which affect each other. This means that not only an overview of the various data is required, but also an overview across the various development steps to get an overall understanding and to detect possible interrelationships.

The model development as well as the evaluation and optimization are an iterative procedure, where the data scientists repeatedly intervene in the model at different stages and repeatedly perform certain analysis steps. Especially when it comes to model evaluation and optimization it is essential to really understand the outcomes of the model. It must be possible to retrace why the model came to the respective result. This means that starting from the produced result, the individual steps of the modelling have to be reexamined repeatedly in more detail. This is a prerequisite for the implementation of optimization measures. Thus, the workflow of a data scientist is an interactive, iterative multistep process that involves trial-and-error, human judgment, and also intense exchange with colleagues.

3.1.3. Essential elements and functionalities

Based on the gained knowledge concerning the characteristics of telehealth data and the workflow of a data scientist, the following essential elements and functionalities of a visualization tool have been identified and are being described in this section (see Figure 10).

Time Series

The above mentioned surveys showed that the data in a telehealth use case as well as in healthcare in general consist largely of time series. Therefore, an appropriate time series visualization poses a core element of the requirements. The number and type of time series to be visualized depends very much on the respective application or the current interest of the user. Also, the number of axes required for the time series visualization is not always the same. The setup of the time series visualization should therefore be as adaptable as possible so that it can be adjusted to changing requirements. An additional aspect is the time range that should be plotted. Again, it should be possible for the user to adapt the visualized time frame to his needs,

so that on the one hand a complete overview is possible, but on the other hand also only a restricted range may be displayed.

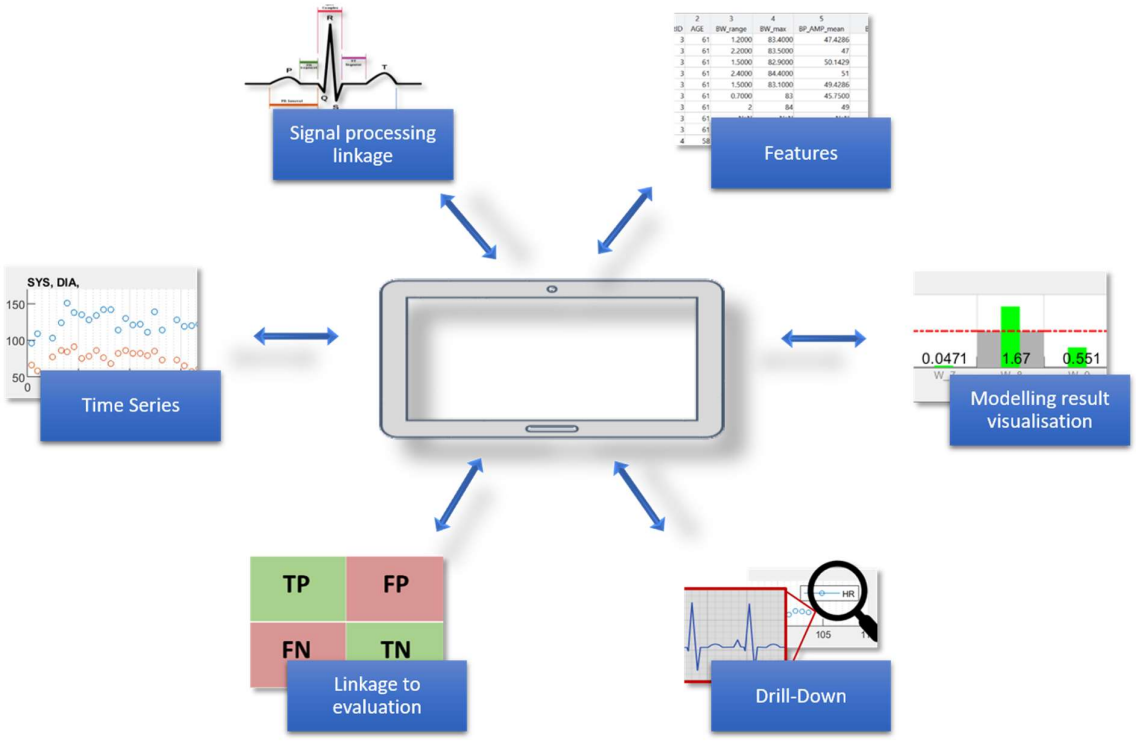


Figure 10: Essential elements and functionalities of the visualization tool

Signal Processing linkage

Time series as well as other data typically have to be pre-processed or transformed before they can be used in the modelling process. Therefore, it is necessary to support an interaction of the viewer and the signal processing functionality of the software. Selection of different available signal processing algorithms should be supported along with the possibility to launch them directly out of the UI combined with automatically reloading all affected viewer elements to keep the visualizations up-to-date. This enables a convenient environment for testing different algorithms and for analyzing different outcomes.

Features

As already mentioned, the features respectively the thereof composed feature set is the essential input of a predictive model. The feature set is the link between the raw data and the model. The performance of the model depends heavily on the composition of the feature set. On the one hand it is about the type of features selected and on the other hand about their respective

parameterization. In addition to that, different features are of different importance regarding model performance (i.e. feature importance). In the course of the model development process it is therefore important to test different versions of feature sets. Thus, to keep a good overview of the current modelling processes, all features should be at hand, which can e.g. be realized through presenting them as a list.

Modelling result visualization

When it comes to distinct model result visualization of a single patient, the pure listing of numerical values is not sufficient. The model result should be presented in an intuitive and comprehensive way in order to instantly provide the required information for further model improvement. For example, a threshold violation could be indicated by color coding depending on whether it is a positive or negative outcome. Another important aspect is, that if the modelling results are related to a certain time interval, such as weekly intervals, the time synchronous relation to the underlying raw data must be given.

Drill-Down

Typical datasets for predictive modelling consist of several data levels. Along with raw data from measuring devices, derived data or any other kind of supplementary information on a measurement can be available. For example, a time series of heart rate values could have been derived from ECG recordings. Thus, while initially the actual heart rate values are shown, there should be a functionality to go further into detail and to take a closer look at the underlying biosignal, i.e. the ECG. Other examples would be various biosignals, lab reports or even imaging data. Thus, some kind of 'drill-down' functionality is required that allows the user to obtain a higher level of details if necessary.

Linkage to evaluation

Major challenges regarding the overall model evaluation are understanding why a given result has been obtained as well as finding potential issues, which might lead to errors or unsatisfactory effects. This is a prerequisite to subsequently being able to carry out measures for model optimization. Therefore, it should be possible to launch the viewer directly out of the evaluation process. A specific example would be a confusion matrix with its four cells i.e. true positive (TP), false positive (FP), true negative (TN) and false negative (FN). By choosing an interesting cell (e.g. the FP cases only), just the chosen sub-selection of cases is presented. In order to then once again allow a closer look at the underlying data, the underlying

measurements, the features that were used for the model etc., to enable and support the process of gaining insights. Thus, the viewer has to have a start-up and update procedure which is triggered by external interactions. All the relevant elements of the viewer have to be initialized and/or updated according to the trigger event.

3.2. Results of the Implementation

3.2.1. Elements and functionalities of the viewer

Figure 11 shows the interface of the 'PATHviewer' with loaded content. In the following, the individual elements and functionalities will be described in more detail.

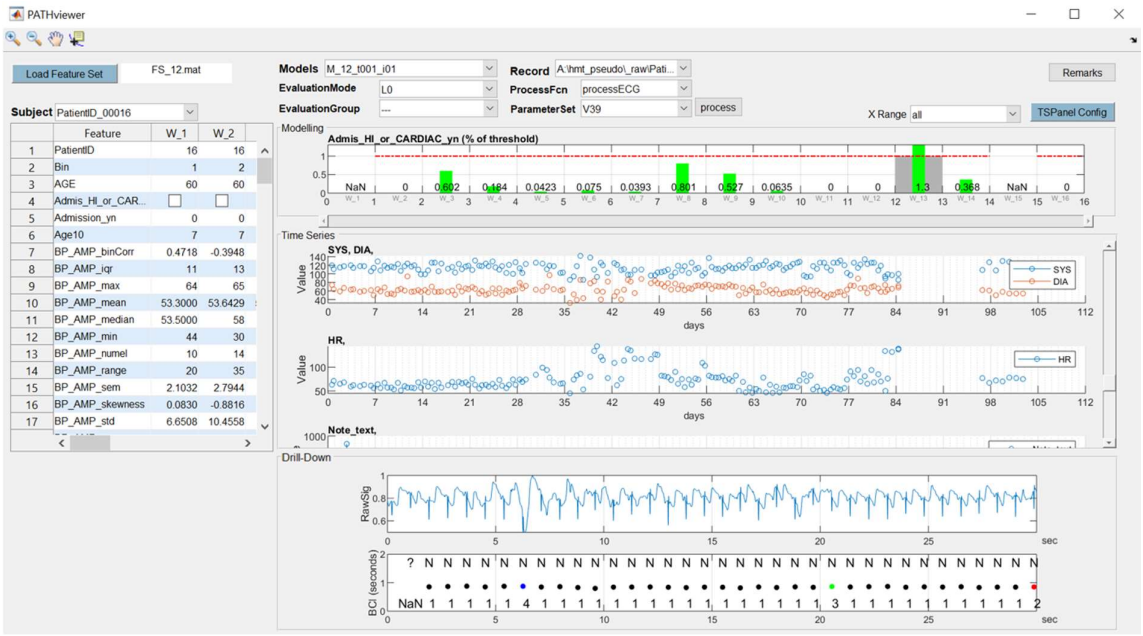


Figure 11: Main window of the PATHviewer

Menu area

The menu section at the top of the viewer serves as a superior selection area that contains various buttons and drop-down menus that are used to perform loading operations, make selections or carry out other actions (see Figure 12).

With the 'Load Feature Set' button a feature set can be loaded. Via a file selection dialog box, the desired feature set is selected. This action also triggers the initial loading procedure of the viewer. In the course of this loading procedure, the raw dataset related to this case is automatically loaded. The 'Subject' menu is thereby filled with the respective list of patients.

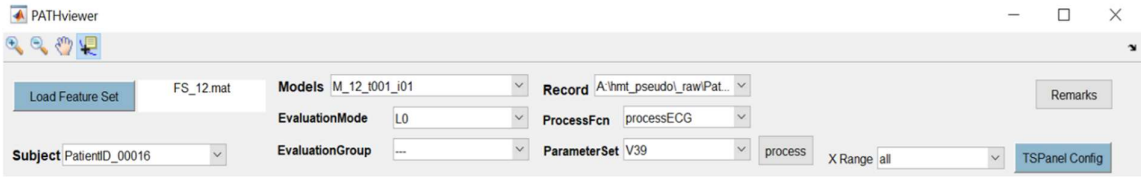


Figure 12: Menu section of the PATHviewer

Furthermore, in the course of this loading procedure, the system automatically searches for already existing models. If models are available that belong to the selected feature set, they are loaded and the drop-down menus 'Models', 'EvaluationMode' and 'EvaluationGroup' are updated accordingly.

The 'Record' menu lists records associated with the currently selected patient, such as ECG recordings. The connection to the signal processing functionalities of the software is established via the drop-down menus 'ProcessFcn' and 'ParameterSet'. The thereby selected signal processing algorithm can be applied to the respective records via the 'process' button.

Via the 'Remarks' button, the user has the possibility to take notes. The button callback opens an Excel file out of the project directory, which is dedicated for this purpose. Via the 'X Range' drop-down menu, settings can be made regarding the range of the x-axis which is to be plotted. With the 'TSPanel Config' button, the 'Time Series' panel can be configured. Both these elements will be described in more detail in the 'Time Series Panel' section.

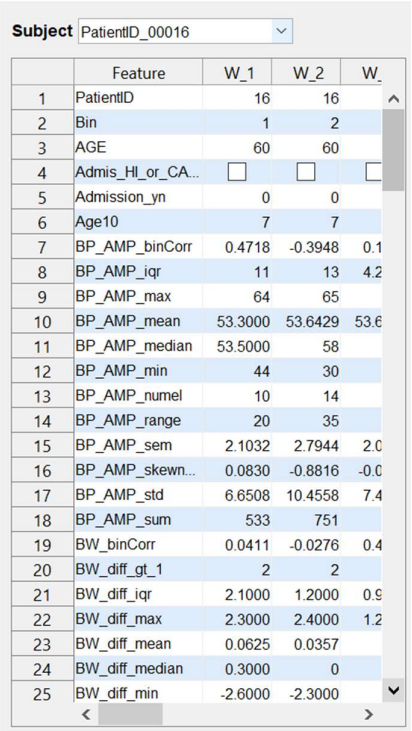
Feature Table

In the feature table, all the features of the currently selected patient are listed (see Figure 13). The first column of the table lists the names of the features. The other columns contain the values of the respective feature for the corresponding time intervals. In this case, these are weekly intervals. It is possible to select one of the features in the table by mouse click, in order to fix the slider position of the feature table when switching between different patients.

Modelling Panel

The modelling result panel, which is depicted in Figure 14 is for the presentation of the model output regarding a distinct patient. In this example, the prediction was about, whether an admission to a hospital will happen or not concerning the respective patient.

Along the x-axis, the time is plotted, which in this case are weekly intervals. The model output for each of these intervals is a threshold and a predicted probability, whether an admission to the hospital will happen.



Subject PatientID_00016

	Feature	W_1	W_2	W_
1	PatientID	16	16	
2	Bin	1	2	
3	AGE	60	60	
4	Admis_HI_or_CA...	☐	☐	☐
5	Admission_yn	0	0	
6	Age10	7	7	
7	BP_AMP_binCorr	0.4718	-0.3948	0.1
8	BP_AMP_iqr	11	13	4.2
9	BP_AMP_max	64	65	
10	BP_AMP_mean	53.3000	53.6429	53.6
11	BP_AMP_median	53.5000	58	
12	BP_AMP_min	44	30	
13	BP_AMP_numel	10	14	
14	BP_AMP_range	20	35	
15	BP_AMP_sem	2.1032	2.7944	2.0
16	BP_AMP_skewn...	0.0830	-0.8816	-0.0
17	BP_AMP_std	6.6508	10.4558	7.4
18	BP_AMP_sum	533	751	
19	BW_binCorr	0.0411	-0.0276	0.4
20	BW_diff_gt_1	2	2	
21	BW_diff_iqr	2.1000	1.2000	0.9
22	BW_diff_max	2.3000	2.4000	1.2
23	BW_diff_mean	0.0625	0.0357	
24	BW_diff_median	0.3000	0	
25	BW_diff_min	-2.6000	-2.3000	

Figure 13: Feature table of the PATHviewer

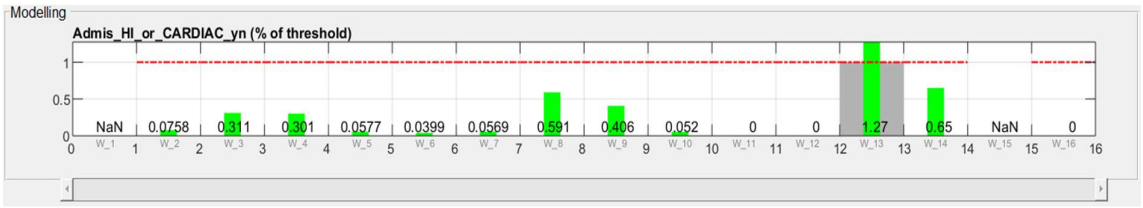


Figure 14: Modelling result panel of the PATHviewer

The predicted probability is visualized on the one hand in the form of numerical values, but also in the form of bars with corresponding height. The red dotted line represents the threshold. If the predicted probability exceeds the threshold, the model indicates that there will be an admission. An actual admission to a hospital is indicated by a grey background of the respective week. The color of the bars is indicating whether the prediction was true (green) or false (red).

The example of Figure 14 shows the bar heights relative to the respective thresholds. Since the model calculates a threshold for each interval, the absolute thresholds are not the same for all intervals. Via a context menu of the axis the display mode can be switched between relative and absolute representation.

Time Series Panel

The time series panel is dedicated for the visualization of time series data (see Figure 15). Along the x-axis, the time is plotted, which in this case are daily time steps. The selected parameters are plotted on the y-axis. By right-clicking on a graph, the plot mode of the respective graph can be switched between stem- and line-plot.

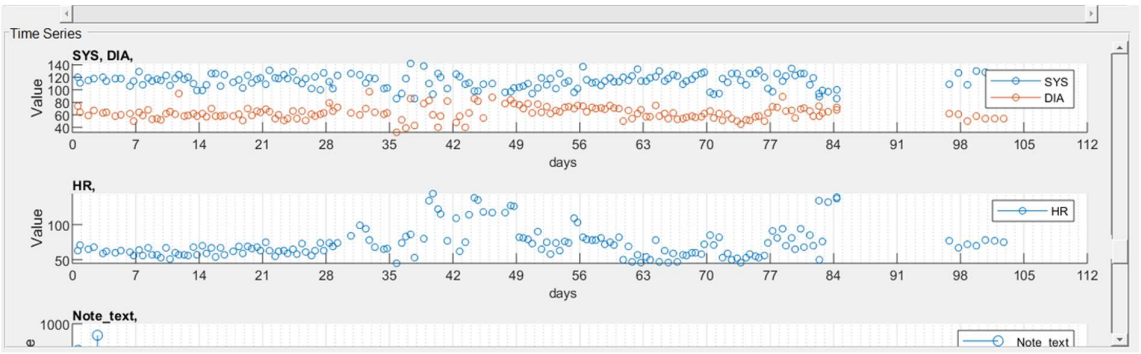


Figure 15: Time series panel of the PATHviewer

The time series panel is implemented in such a way that it is user-configurable. The corresponding configuration menu can be opened either via the ‘TSPanel Config’ button in the menu section or by right-clicking in the panel. The configuration menu is a separate GUI, which consists of three tabs, i.e. ‘New’, ‘Save’ and ‘Load’.

In the ‘New’ tab, the user is able to create a new configuration (see Figure 16). In a first step the number of axes that should be displayed in the time series panel has to be defined. Subsequently the axes type of all the axes objects has to be defined. This is because the data structure of the PATH program is structured in such a way that the different time series are grouped into different categories (e.g. measured values, interventions, records). For the visualization, this has been implemented in such a way that each axes object must be assigned to one of these categories, which corresponds to the ‘AxesType’. Another parameter that has to be defined is the size of the respective axes. There are three different sizes available, minimum (min), medium (med) and maximum (max).

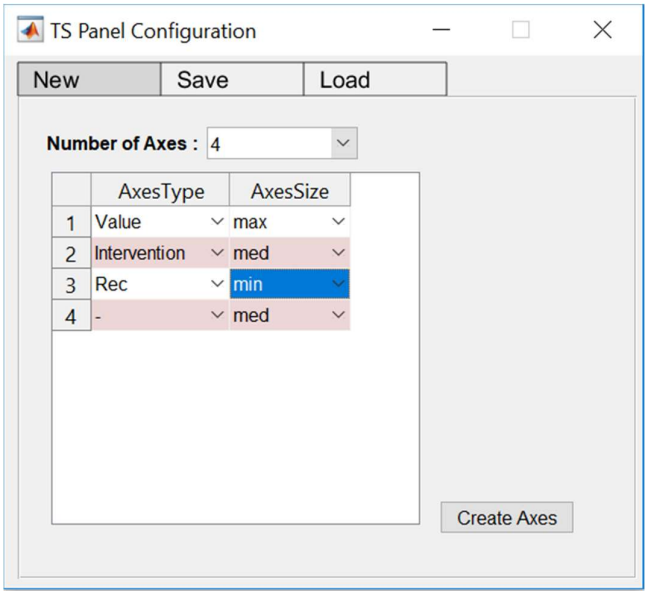


Figure 16: Time Series Panel configuration menu: New

The ‘Save’ tab of the configuration menu is depicted in Figure 17. The specification of the current time series panel configuration, which is going to be saved, is listed in a table. This specification includes the axes types, the axes sizes and the plotted x- and y-variables. When specifying a configuration name, the system checks whether the selected name already exists to prevent possible name conflicts. By pressing the ‘Save Configuration’ button, the specifications are written to an Excel file.

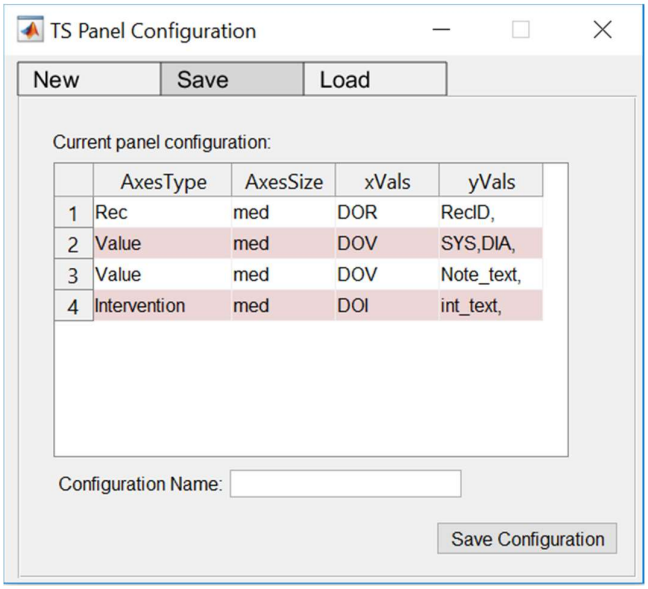


Figure 17: Time Series Panel configuration menu: Save

In Figure 18 the ‘Load’ tab is depicted. The load functionality accesses the Excel configuration file to which the specifications were previously saved. In a drop-down menu all the existing configurations are listed. In a table, a preview of the currently selected configuration is listed. In this preview, the different axes of the respective configuration are listed with information about the axes types, the axes sizes and the selected x- and y- variables. Through clicking the ‘Load Configuration’ button, the selected configuration is loaded, and the viewer content gets updated depending on the currently selected patient.

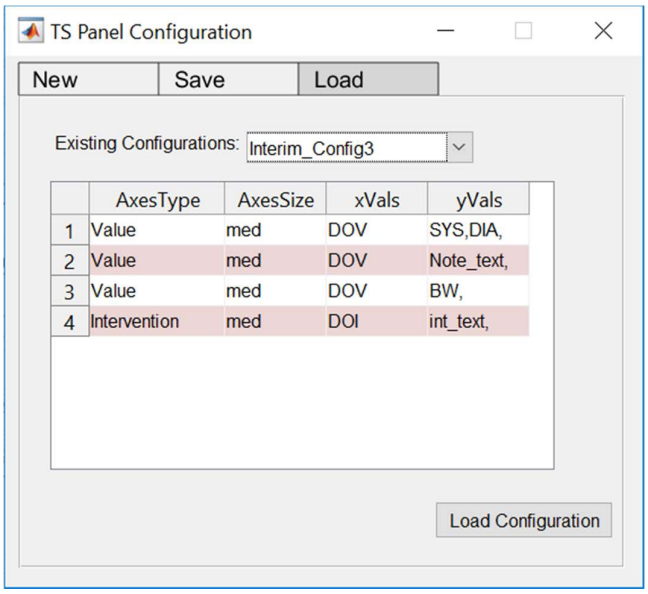


Figure 18: Time Series Panel configuration menu: Load

The selection, which parameters are to be plotted per axes, is carried out via a separate menu, which is opened by double-clicking on the respective graph (see Figure 19). Under ‘Raw dataset’ the category of the corresponding graph is listed, which corresponds to the category of time series that can be plotted in it. For the x-axis, the possible time parameters are listed, of which one must be selected. (e.g. ‘DOV’ for Day of value, ‘WOV’ for Week of value, etc.) For the y-axis, on the other hand, any number of parameters can be selected, which are then plotted in different colors in the respective graph.

Figure 19: Selection of the parameters to be plotted via the time series axes data menu

The time series graphs are arranged in time synchronism with the model result graph. Settings regarding the time range to be displayed can be made via the 'X Range' drop-down menu. At the current state, there are two different options available. On the one hand, the visualization can extend over the entire time range for which raw data is available. (i.e. 'all'). On the other hand, the range, which is to be visualized, can be restricted to the period, for which modeling was performed (i.e. 'modelled bins only'). In the case of restricted visible time range, a scrollbar is activated with which the user is able to shift the visible time frame along the entire time range.

Data cursor function

Details of the plotted data points can be displayed by a data cursor interaction. A customized data cursor function was implemented, which generates a text information window containing specific details of the selected data point (see Figure 20). The popup window shows the identifier (ID) of the selected data point, the patient ID, the absolute as well as the relative day of the value (DOV), and the data point value itself. In the example illustrated in Figure 20, an interaction with a data point of a time series of clinical notes is depicted.

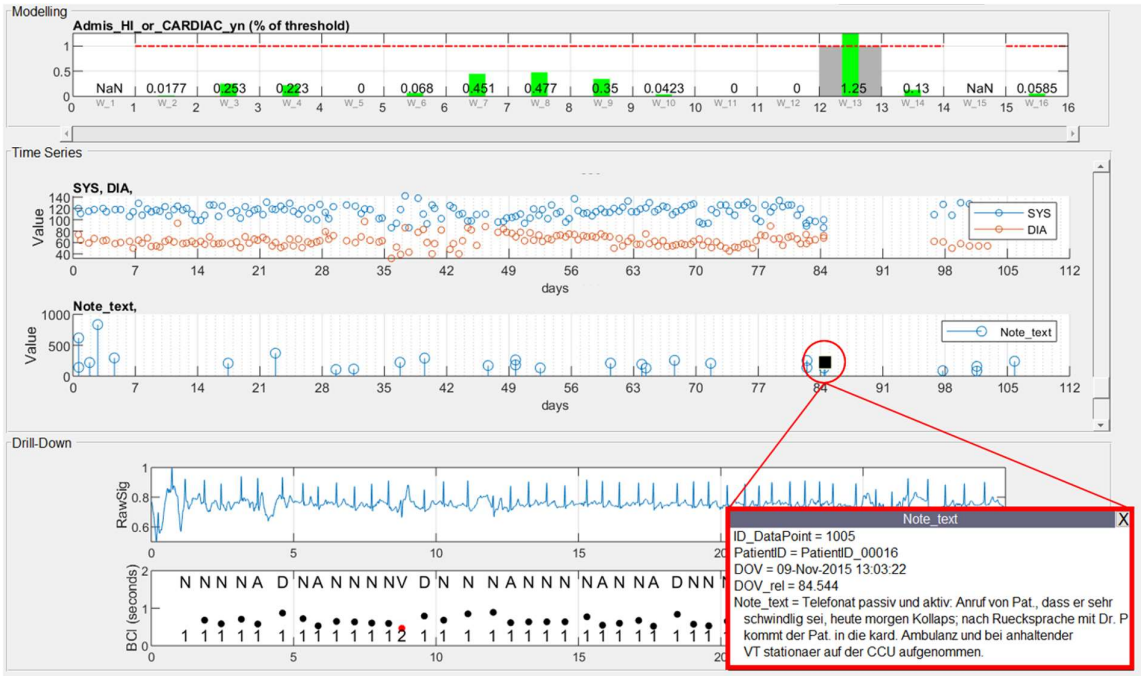


Figure 20: PATHviewer data cursor information window displaying the metadata and content of a textual note

Drill-Down Panel

The purpose of the drill-down panel is to allow the user to take a deeper look into the data. If there is a time series of measurements or events, for which an underlying data level exists, the individual data points can be selected via mouse click. The layout and content of the drill-down panel then adapts to the respective measurement or event type. In the case of the example shown in Figure 21, it is an ECG recording with the corresponding signal analysis. The first graph shows the raw ECG, the second the beat intervals and the beat classification.

The underlying program sequence is implemented in such a way that this drill-down process is divided into two steps. There is a ‘drill-down setup’ which defines the basic elements and layout of the drill-down panel. Then there is a ‘drill-down procedure’ in which various processes, corresponding to the respective measurement or event, are defined and which are to be executed in the course of the drill-down process.

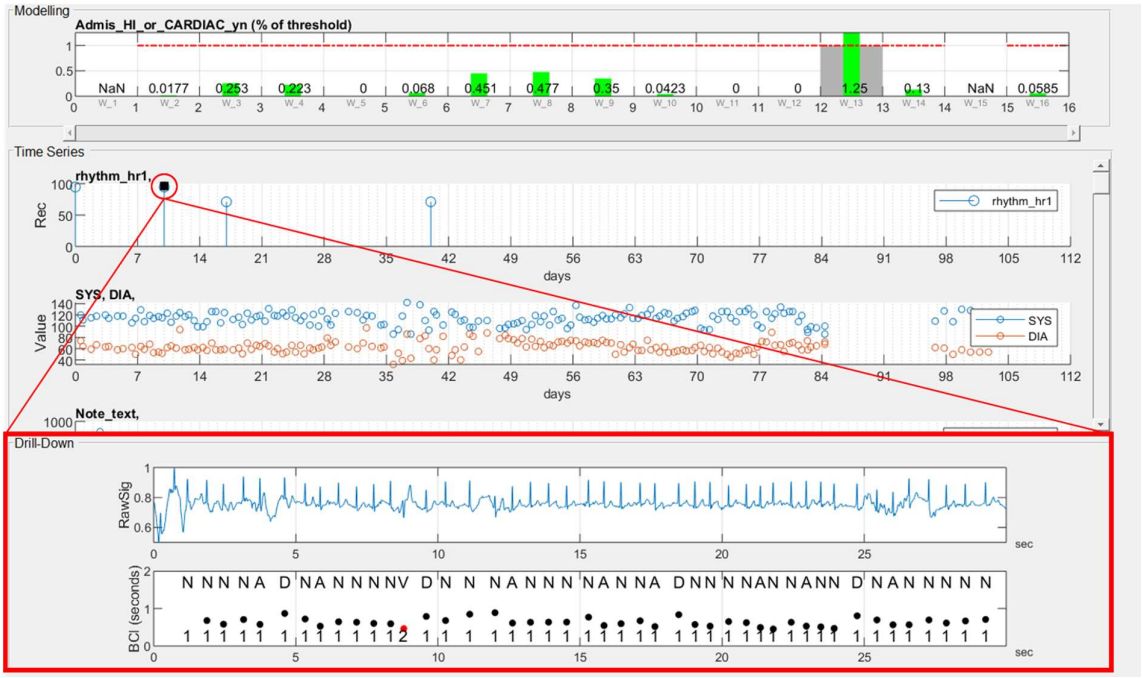


Figure 21: Drill-Down panel of the PATHviewer

3.2.2. Startup procedures

There are two possibilities for the start-up procedure of the PATHviewer. On the one hand, the viewer can be opened from the MATLAB Command Window using the command 'PATHviewer' (see Figure 22). The main window of the viewer appears, where all menu fields and panels are empty at first.

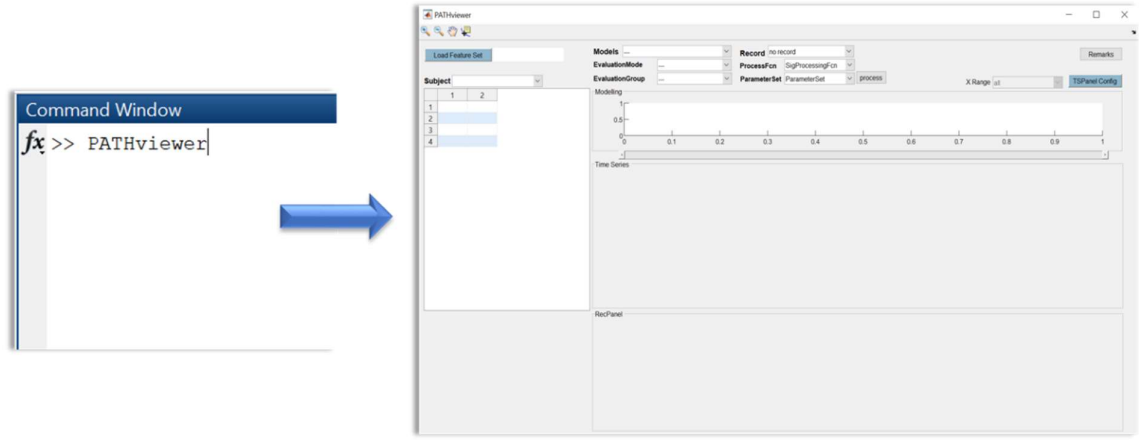


Figure 22: Start PATHviewer out of the MATLAB Command Window

The initial step is to load a feature set via the ‘Load Feature Set’ button as described under ‘menu area’ in section 3.2.1. As a result of this loading process, the ‘Subject’ menu is filled with all available patients, of which the first patient of the list is selected, and the feature table is updated accordingly. In the course of this loading process, the program also checks whether there are already existing models for the selected feature set. If this is the case, the model is loaded and the modelling result graph is updated accordingly. Subsequently, the time series panel can be configured using the configuration menu described above and all other functionalities can be utilized.

On the other hand, the viewer can be called directly from the evaluation after a modeling run. Two scenarios of this kind are shown in Figure 23. There are two confusion matrices, one of which is a scatter plot representation (a) and the other a fourfold table (b). They show the evaluation of the model output of all patients for a specific week (i.e. week 14). This means that in the scatter plot representation, each data point corresponds to the prediction outcome of a specific patient. By selecting one of the data points from a), exactly this patient is loaded into the viewer and the content of the viewer elements is updated accordingly.

In b), each field of the fourfold table represents a group of patients (i.e. FP, TP, FN, and TN). By clicking on one of these fields, the corresponding group of patients is loaded into the viewer.

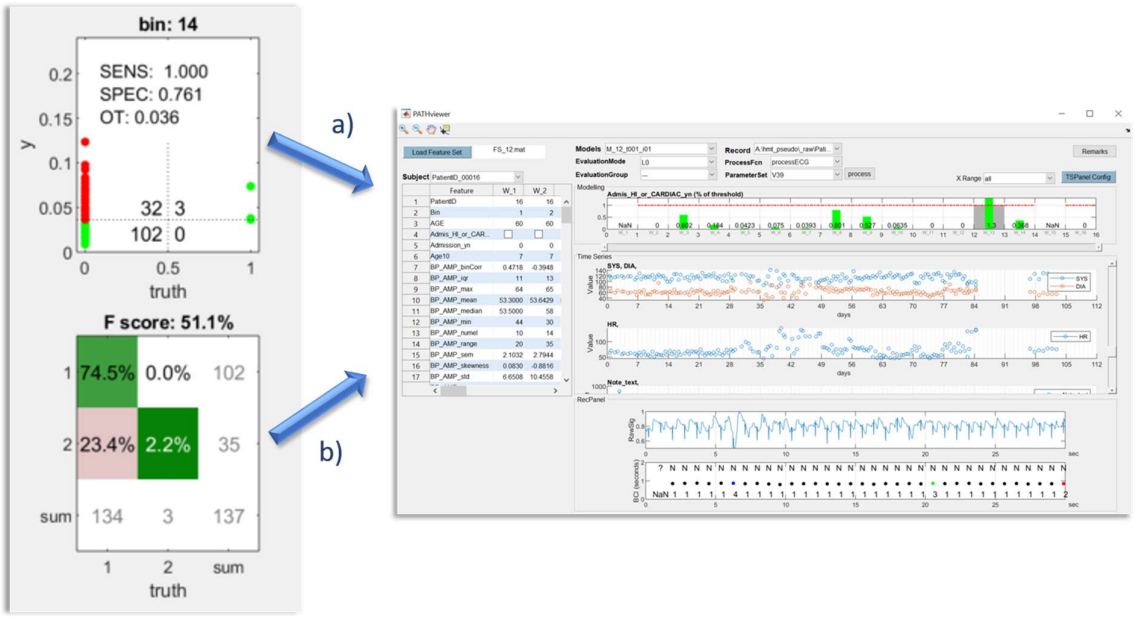


Figure 23: Load PATHviewer from confusion matrices a) scatter plot representation b) fourfold table

The cases, which have just been described are two concrete examples of how the viewer can be loaded from the evaluation. In the course of the implementation, however, a general load and update routine was implemented, which makes it possible to call the viewer from various other representation forms of evaluation or statistical analysis.
However, two cases have to be distinguished here. If the viewer is opened initially through interaction with the evaluation, the configuration of the time series and the drill-down panel must be carried out in a separate step. If, on the other hand, there is an interaction with the evaluation when the viewer is already open, the viewer is updated including the previously selected configuration.

3.2.3. Software architecture

The basic software architecture of the viewer is shown in Figure 24. In addition to the main GUI 'PATHviewer' there are two extra GUIs. On the one hand the configuration menu of the time series panel and on the other hand the menu for selecting the parameters to be plotted, which have already been described in more detail in section 3.2.1.
As illustrated in Figure 24, the functions of the toolset can basically be divided into four categories. The setup functions, such as 'createTSAxes' or 'drillDownSetup', are generally called only once in the course of a configuration process. The update functions, such as 'plotTimeSeries', on the other hand, are executed every time patients are switched. These functions are used to update the previously established setup.
A further category are the functions that serve the interactive functionalities. The 'dataCursorFcn' function is used, for example, to retrieve and display detailed information on individual data points of time series. Furthermore, the drill-down process can also be started via this function. Then there are supplementary functions that are required for specific functionalities. With the help of 'getXRange', for example, the time range of the different graphs to be plotted is determined.

The core element of the structure is the main interface of the viewer, the MATLAB GUI 'PATHviewer'. This GUI consists of an m-file, which corresponds to a MATLAB script file, and a fig-file, for the corresponding MATLAB figure.
The PATHviewer figure, which is created when the program is executed, contains in its properties all relevant information about the main window itself as well as the handles to all existing children (i.e. sub-elements as buttons, drop-down menus or graphs).

An extract of these properties is shown in Figure 25. Additionally, to the properties, it is also possible to store data to the GUI and subsequently retrieve it. The PATHviewer GUI thus serves as a base and central storage location for all processes running in the toolset.

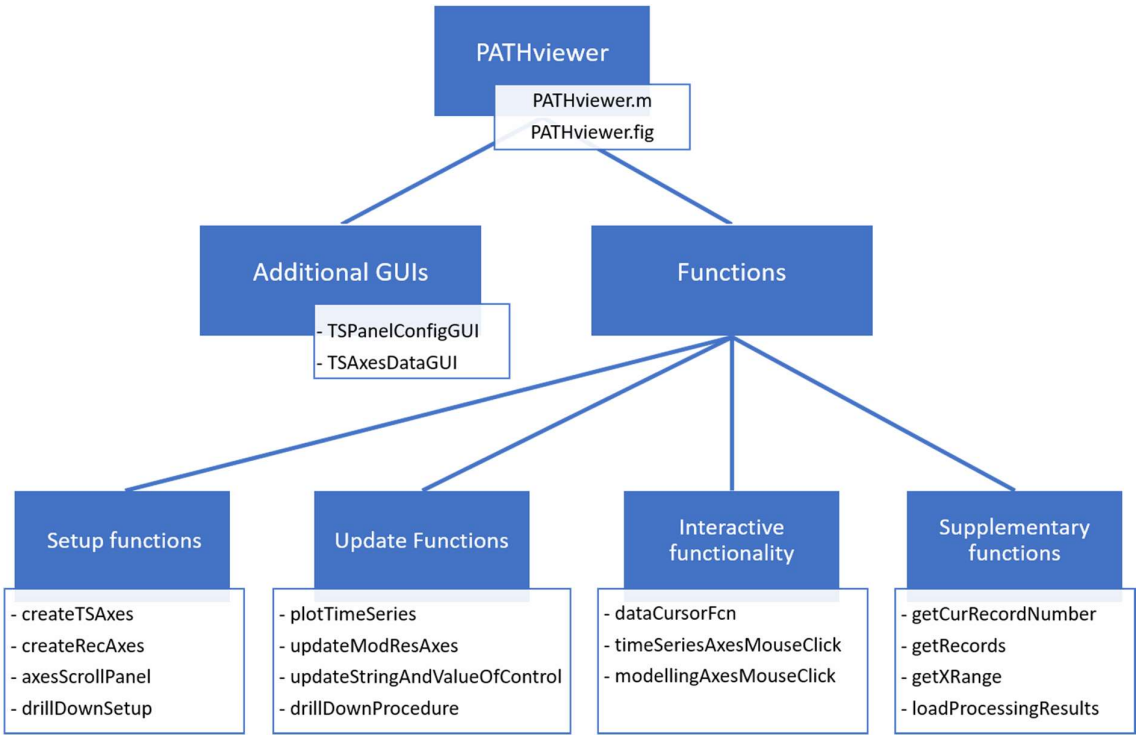


Figure 24: PATHviewer software structure

```
Command Window
K>> Properties = get(hViewer)

Properties =

  struct with fields:

              Position: [6.8750 0.6190 224.2500 46.1429]
         OuterPosition: [5.5000 0.0952 227 50.6667]
         InnerPosition: [6.8750 0.6190 224.2500 46.1429]
                  Type: 'figure'
                   Tag: 'PATHviewer'
              UserData: []
              Clipping: 'on'
           WindowStyle: 'normal'
           WindowState: 'normal'
          DockControls: 'on'
                Resize: 'on'
         PaperPosition: [0.0294 0.2273 0.9412 0.5455]
     PaperPositionMode: 'manual'
              Children: [29×1 Graphics]
                Parent: [1×1 Root]
      HandleVisibility: 'callback'
```

Figure 25: Properties of PATHviewer figure (extract)

Besides the MATLAB predefined processes, which are necessary for the basic structure and initialization of the GUI, the PATHviewer.m script also contains other program parts. This includes the initialization processes defined by the user. Furthermore, the basic loading processes, such as those of the raw data, the feature set or those of already existing models are performed in this script. It also contains all callback functions of the UI controls (i.e. buttons, drop-downs, slider, etc.). Additionally, this script also contains the load and update process, which enables an external launch of the viewer (e.g. from evaluation).

In the following section, the role of the PATHviewer GUI as a central storage element is described in more detail. As already mentioned, various loading processes take place in the PATHviewer.m file. The loaded data must be saved in order to have it available in the program for subsequent access. In Figure 26 the respective pseudo code section is shown. To be able to use the PATHviewer figure as a central storage object and to be able to access it later from anywhere in the program, it has to be initially stored in the application data of the root object (i.e. command window). This is done with the ‘setappdata(0, ‘hPATHviewer’, gcf);’ command. ‘0’ stands for the root object, ‘hPATHviewer’ for the name identifier and ‘gcf’ gets the current figure (i.e. PATHviewer).

The feature set, the time series data and, if available, the model results are loaded during program execution. The results of the respective loading process (i.e. F, T and M) are appended as fields to the ‘Data’ structure. This ‘Data’ structure is then stored in the graphics object ‘hPATHviewer’ with the name identifier 'data' using the ‘setappdata’ command. The ‘setappdata’ function is generally used to store data in a UI.

```
function varargout = PATHviewer(varargin)
% PATHVIEWER MATLAB code for PATHviewer.fig
% ...
% -- Define application data for data sharing among GUIs (and callbacks)
% -- store the handle to the root named hPATHviewer
setappdata(0,'hPATHviewer',gcf);

% ...loadFeatureSet -> F
% ...
Data.F  = F;

% ... loadTimeSeries -> T
% ...
Data.T = T;

% ...loadModelResults -> M
% ...
Data.M = M;
```

setappdata(hPATHviewer,'data',Data); ⟹ hPATHviewer

Figure 26: PATHviewer.m pseudo code example for the load and save procedures concerning feature set, time series and model results

To retrieve the data elsewhere in the program the 'getappdata' function is utilized. One such example is shown in Figure 27 by means of the 'plotTimeSeries' function. First in line 7 the handle of the PATHviewer figure is retrieved from the root using the 'getappdata' function. In line 8 the 'guidata' command loads the entire handles structure of the viewer (i.e. handles of the children). In line 10, the previously saved 'Data' structure is reloaded using 'getappdata' with the name identifier 'data'. Via dot notation all the previously saved data including the time series data can be accessed (line 12).

hPATHviewer ⟹

```
1
2   function [] = plotTimeSeries(hObject, hViewerName)
3   % PLOTTIMESERIES plots the respective time series
4   % of the selectet Patient
5
6   % get the root and the handles structure of the Viewer
7   hViewer         = getappdata(0,hViewerName);
8   hStructViewer   = guidata(hViewer);
9
10  Data = getappdata(hViewer,'data');
11
12  T = Data.T;
13  % ...
14
```

Figure 27: (pseudo) code example to retrieve data from the PATHviewer figure

Both of these functions, the ‘setappdata’ and the ‘getappdata’ make it possible to share data between callbacks or between the separate GUIs. Therefore, these two functions also play an important role for many other parts of the program.

An example for this, which is illustrated in Figure 28, are the metadata of the time series axes (i.e. axes type, axes size, x and y variables, stem mode). The settings that are made, for example, via the ‘TSPanelConfigGUI’ or via the ‘TSAxesDataGUI’ are combined in an ‘AxesMetaData’ structure and assigned to the respective axes objects via ‘setappdata’ and the name identifier ‘amd’. Also, the stem/line-plot setting is saved in this structure. This structure is retrieved via ‘getappdata’ in the ‘plotTimeSeries’ function and the required parameters can be accessed.

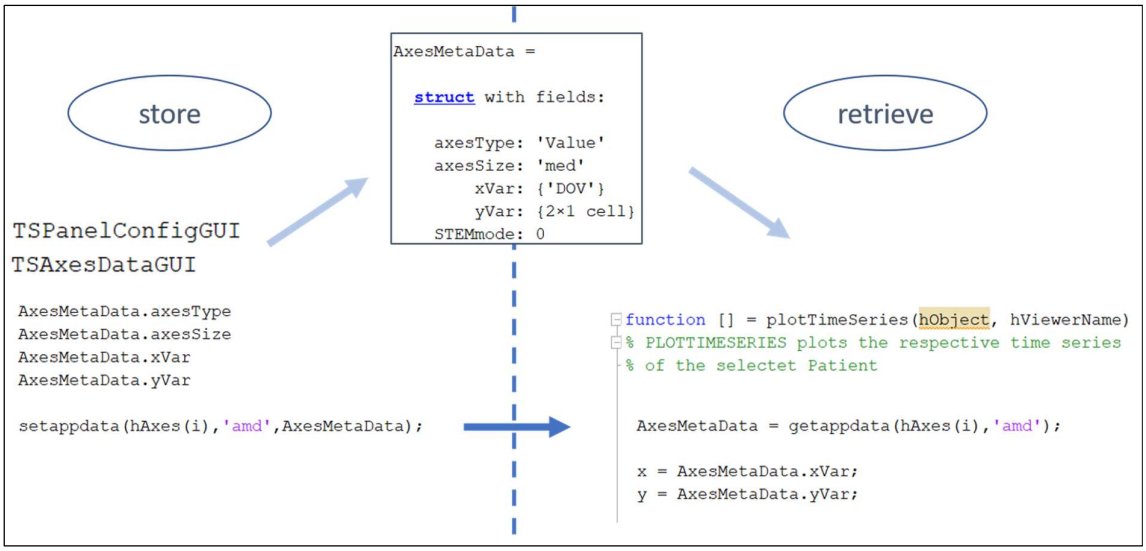


Figure 28: Storage and retrieval of the axes metadata of time series axes (pseudo code)

The ‘User Data’ of the individual UI controls of the viewer also plays an important role in the functionality of the program. Each UI control element (e.g. button, drop-down, slider etc.) has a field ‘User Data’ in its properties. This ‘User Data’ can be used to store information and access it from elsewhere in the program.

A specific example where this is the case is the slider, which determines the time range which should be plotted. The ‘User Data’ of the slider stores the start and end point of the range to be displayed. This information is then accessed by the corresponding functions (i.e. plotTimeSeries and updateModResAxes).

Another case, where the ‘UserData’ plays an important role is when the viewer is called from the evaluation (see Figure 29). The figure that is created during the evaluation (e.g. confusion matrix) provides a structure ‘model’. It contains all relevant data and information related to the corresponding evaluation, including the name of the used feature set, the model name, patient IDs, etc. With the help of ‘getappdata’, this information is retrieved from the current figure and stored in ‘M’. The relevant information is stored in the ‘UserData’ of the corresponding UI control elements of the PATHviewer, which is subsequently called with the external update routine ‘UpdateExternal’. In the course of this update routine, the information stored in the ‘UserData’ is accessed again and the viewer is loaded or updated accordingly.

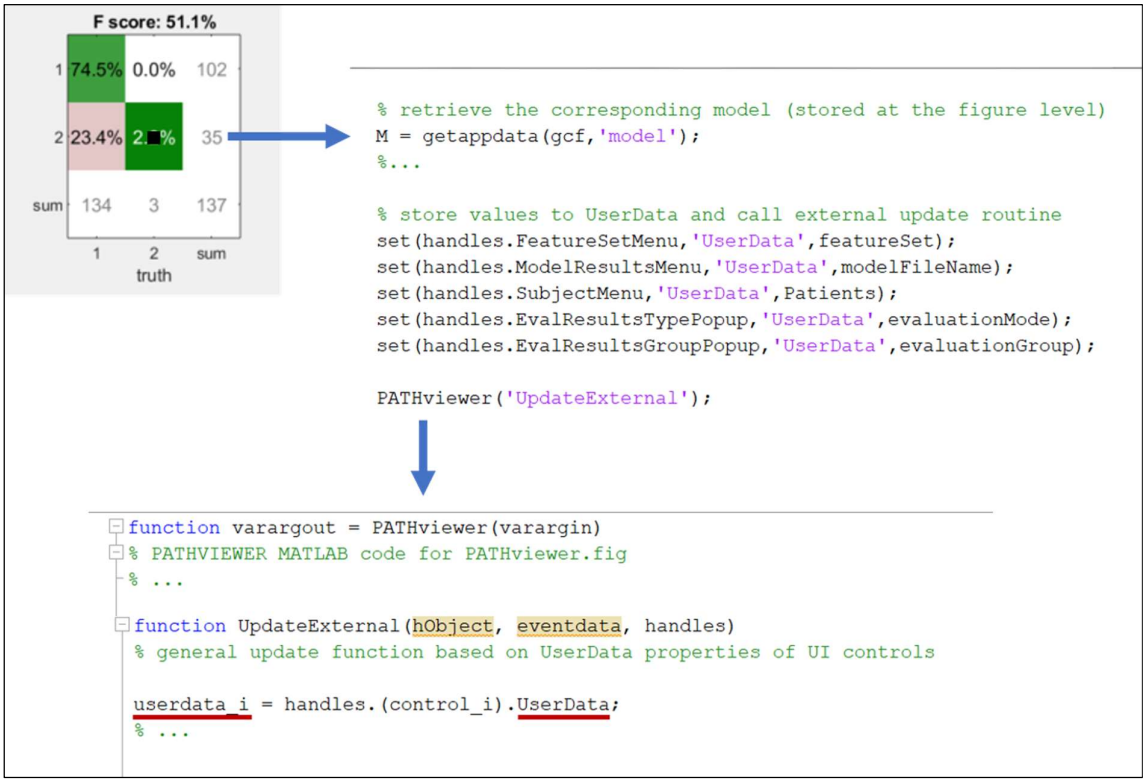


Figure 29: Usage of ‘UserData’ property for evaluation startup procedure

Another important component of the PATHviewer is the drill-down functionality. The corresponding schematic program sequence is shown in Figure 30. The starting point of this functionality is the data cursor function of the viewer (i.e. dataCursorFcn). This function not only creates an info window with detailed information about the selected data point, it also checks, if there is a drill-down process for the respective data point. Whether such a process

exists for a certain data type is specified in the definition files of the PATH program. This specification is stored in the metadata of a data point. In the data cursor function this is being checked and the corresponding drill-down setup and procedure is executed accordingly.

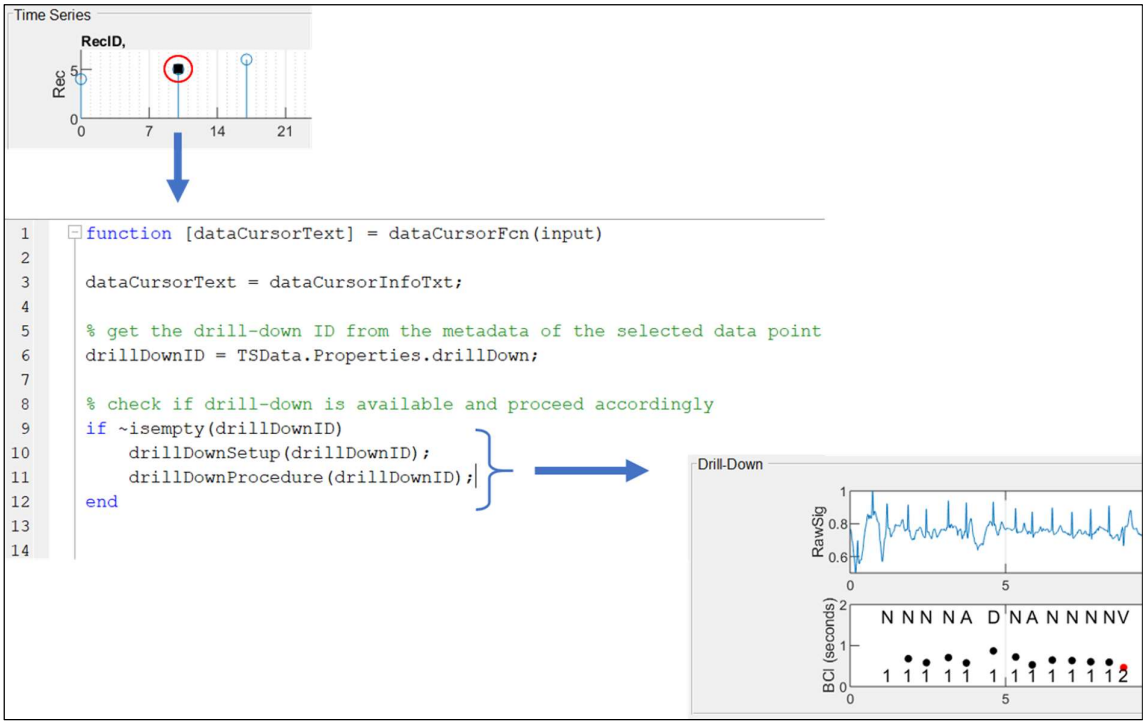


Figure 30: Process of drill-down functionality (pseudo code)

4. Discussion

4.1. Interpretation of the results

The requirements analysis regarding an interactive visualization tool for the predictive analytics domain was carried out primarily on the basis of telehealth data. Nevertheless, the potential application areas of the identified elements and functionalities described in Section 3.1.3 are not limited to the telehealth domain. This is because they cover many aspects that are relevant for a variety of applications, since the basic steps of a model development process are included. The concept development of a tool that supports data scientists in the development of predictive models in the telehealth domain also revealed many aspects that have to be considered. A major challenge in terms of the amount and variety of available data is finding the right balance. On the one hand, a good and comprehensive overall overview of the situation should be available, providing simultaneous visualization across several data levels. On the other hand, however, care must also be taken to ensure that there is no overload of information that overwhelms the user.

The developed tool takes the former aspect into account by simultaneously presenting the model result, the underlying time series data as well as the features used for modelling on a single interface. Whereby the model result visualization is not a simple presentation of numerical values. The simultaneous representation of the probabilities as bars as well as their color coding, which indicates a right or wrong result, leads to the fact that multiple important information is contained at the same time in a visualization. This presentation of the results enables a fast and comprehensive perception by the user.

Another aspect that contributes to a good overview is the time-synchronous arrangement of model results and time series. The fact that the time series are plotted with exact synchronization to the model results makes it possible to determine specific correlations between model output and the underlying raw data. This can especially help to identify specific patterns or anomalies in the data, e.g. around a hospitalization event. On the basis of these insights, existing hypotheses can be tested, or new hypotheses can be formulated in order to subsequently develop new features for the modelling process.

An important aspect in this context is also, which time series should be plotted to achieve these insights. The configurable time series panel offers the user the necessary flexibility to adapt the visualization of the time series data to her/his current requirements and research question. In addition to the number, type and order of the axes, also their size can be selected. This size

selection of the axes makes it possible to achieve optimal use of the limited space by adapting the axes to the respective data type. For the representation of binary data, for example, a small graph is sufficient. If, on the other hand, the data has a broad value range over the y-axis or if several parameters are to be displayed in one axis, a larger graph can be selected accordingly in order to obtain a better overview. The actual decision, which parameters are to be displayed in the respective axes, can be made via the separate GUI and can be adjusted later at any time. This and the save and load functionality give the user a high degree of flexibility in time series visualization.

The second aspect mentioned before, prevention from information overload, is ensured on the one hand by this configurability of the viewer and on the other hand by the concept of 'details on demand'. The implemented data cursor function makes it possible to provide more detailed information about data points. This can be used to retrieve certain metadata of the measured values. Another important application of this functionality is, when there is text information behind data points. For example, as shown in Figure 20, notes of health professionals can be retrieved. The insight into these notes enables the data scientist to gain a deeper understanding of the situation at hand and to better understand the course of treatment. This 'details on demand' functionality is therefore an important building block to gain a comprehensive understanding.

Another type of 'details on demand' offers the drill-down functionality. It enables the user to go one step deeper into the data. This functionality was developed using ECG recordings which were added to the test dataset. Therefore, the drill-down panel in Figure 21 consists of a graph for the raw ECG and a graph for the processed ECG in its current state. However, the structure of this functionality was implemented in such a way that it can be applied as generally as possible. By dividing the program sequence into a part for setup and a part for procedures, this panel can be easily adapted to the respective use case. Thus, this drill-down panel offers a freely configurable area which is of particular importance with regard to the variety of possible data types.

The individual elements and functionalities of the viewer represent important building blocks in the work process of data scientists. Furthermore, the two available startup procedures ensure that the viewer can be effectively integrated into the general workflow of data scientists. The startup from the MATLAB command window, where the data is loaded step by step into the viewer and subsequently analyzed, is particularly useful for initial data analyses. Loading or updating the viewer from the evaluation, on the other hand, is very well suited for the model optimization process. The user can load specific cases or groups of patients (e.g. FP cases) into

the viewer in order to analyze them in more detail. This ability supports above all the iterative nature of a model development process in which the goal is gradually approached.

The viewer was developed and designed to support data scientists in their daily work. The primary field of application thereby lies in the development and optimization of predictive models. But the developed visualization toolset, with all the presented elements and functionalities, allows further application scenarios. The toolset can also be used for pure data analysis or statistical evaluations. Because even without carrying out actual modeling, datasets can be visualized and interactively examined.
However, the field of application of the viewer is not exclusively limited to data scientists. The tool can also be used for explanatory purposes or joint workshops with other stakeholders of a project. Due to the comprehensive visualization and analysis possibilities, datasets can be jointly discussed, and the background of developed models can be explained in more detail to ultimately facilitate collaborative analytical reasoning.

The framework of the ECG viewer presented in section 1.2.2 was successfully extended in the course of the development of the PATHviewer. While the ECG viewer is a good tool for the specific analysis of ECG data, the PATHviewer offers visualization and analysis options for a wider range of applications. Due to the drill-down functionality it is thereby also possible to integrate certain aspects of the ECG viewer into the new tool. Because in addition to the display of ECG raw signals, the signal processing together with visualization of the processing results can be integrated in the PATHviewer.

Concerning the implementation, the use of MATLAB's GUIDE provided all required types of visualization as well as the necessary possibilities regarding interactive functionalities. However, in the course of the implementation of the viewer it turned out that GUIDE has some drawbacks. In certain cases (e.g. fixing the slider position of the feature table) it was necessary to use undocumented solutions and Java to achieve the desired result. Furthermore, in some cases, such as the implementation of a tab concept, which is used in the configuration menu, workaround solutions were required. In this respect, after further updates and extensions, the 2016 introduced App Designer could possibly provide a more convenient solution for the implementation of such tools in the future.

4.2. Limitations

In the course of the testing and validation phase, certain limitations were identified regarding the practical use of the viewer. One issue concerns the initial start-up process of the PATHviewer from the evaluation. The viewer is thereby started without preset configurations. The setup of the time series panel as well as the drill-down panel has to be carried out by further interactions of the user. At this point it turned out that a default configuration would be beneficial, which is loaded automatically during the start-up process.
The validation of the viewer with regard to its generic applicability has shown that the developed framework is basically well suited for other use cases. The concept described in section 2.6 was successfully implemented with the structures of the PATHviewer, especially the drill-down functionality. However, some adjustments were necessary in the course of this implementation. It was necessary to modify the code at certain points to adapt the viewer to the new use case. Currently, a separate individual drill-down setup and procedure has to be established for a new use case. For this, the PATHviewer lacks the availability of modular building blocks, with which a newly adapted setup can be conveniently configured.

In its current state of development, the viewer concept also offers only limited interaction possibilities with regard to direct interventions in the model development process. The viewer provides comprehensive visualization and analysis capabilities across the various steps in a model development process. However, it is not yet possible to really navigate through the process step by step with the help of the viewer and to interactively intervene in the various process steps directly from the interface. In order to be able to implement such a concept, however, also adaptations to the underlying PATH program would be necessary.

4.3. Outlook

The viewer concept developed on the basis of the HerzMobil telehealth data has a number of additional application scenarios. The adaptation of the viewer to a dataset of ergometric performance tests of cardiac rehabilitation patients did not only served the purpose of validation, it also poses a real-world application of the viewer. The viewer will be used to analyze the available data in more detail, with the goal of drawing conclusions about the effectiveness of certain measures in the course of the rehabilitation program.

Another concrete use case is the analysis of Timed Up-and-Go (TUG) measurements. A TUG test is a test performed to assess the mobility and the risk of falls of a patient. At the start, the patient sits on a chair. After being instructed to stand up, the patient walks a certain distance, turns around and sits down on the chair again. A device developed by AIT measures the time while continuously recording the distance the patient is walking. The output of these measurements are movement curves of the patients. With the help of the viewer, time series of measurements and the corresponding movement curves can be visualized and further analyzed. On the basis of these analyses, models can subsequently be developed, which allow certain statements to be made based on these movement curves. In addition to displaying time series and movement curves, videos that could be recorded during the measurements might also be integrated into the viewer. This additional type of information would allow an even better interpretation of the data.
Apart from these two concrete use cases, the viewer concept can be applied to various datasets that contain time series and have a multidimensional characteristic. This opens up a very broad field of application in the context of health and care data.

In connection with the variety of possible use cases for the viewer, there are also a number of possible further developments of the toolset. On the one hand, adaptations are necessary to overcome the limitations mentioned above. This includes the implementation of a default configuration for the evaluation start-up procedure. Moreover, the current framework can be further optimized so that it becomes even more generic, in order to be able to apply the viewer with its current functionalities even faster to other use cases.
If the viewer concept is not used for model development, but purely for general data analysis purposes, there are other priorities regarding the elements of the main interface. In order to enable more flexibility in this respect, further development towards a more modular setup is necessary.
A long-term goal could be to further develop the viewer concept in a way that it is not only used as a visualization and analysis tool, but that it can also be used to actively interact with the modeling process. Which means that the viewer becomes a more comprehensive UI, which is integrated even deeper into the PATH program. This would enable direct and specific interventions in the various stages of the model development process.

5. Conclusion

In the course of this thesis it became clear that, with all the possibilities predictive modelling brings with it, the human being still plays an indispensable role in this process. The developed viewer aims to combine the computational capabilities with the skills of human beings to achieve an optimal result.

Even if there are still many possibilities for further development, the current state of the viewer already allows better management of the model development process and thereby helps to gain deeper insights and a better understanding regarding complex data analytics challenges. This is of crucial importance in two respects. On the one hand, it is a key aspect for the development and optimization of predictive models. On the other hand, the generated models and their results can be better interpreted and explained to third parties. This is especially critical regarding the acceptance of data-driven decision support systems in real world applications. Thus, the developed viewer makes an important contribution on the way to a more data-driven healthcare with the ultimate goal of autonomous telehealth systems.

Bibliography

[1] World Health Organisation WHO, “Telehealth.” [Online]. Available: https://www.who.int/sustainable-development/health-sector/strategies/telehealth/en/. [Accessed: 18-Feb-2019].

[2] U.S. Department of Health and Human Services, “Telehealth Programs,” *Health Resource and Services Administration*, 2019. [Online]. Available: https://www.hrsa.gov/rural-health/telehealth/index.html. [Accessed: 18-Feb-2019].

[3] Center for Connected Health Policy CCHP, “About Telehealth.” [Online]. Available: https://www.cchpca.org/about/about-telehealth. [Accessed: 18-Feb-2019].

[4] J. Kvedar, M. J. Coye, and W. Everett, “Connected Health: A Review Of Technologies And Strategies To Improve Patient Care With Telemedicine And Telehealth,” *Health Aff.*, vol. 33, no. 2, pp. 194–199, 2014.

[5] J. Morak, H. Kumpusch, D. Hayn, R. Modre-Osprian, and G. Schreier, “Design and Evaluation of a Telemonitoring Concept Based on NFC-Enabled Mobile Phones and Sensor Devices,” *IEEE Trans. Inf. Technol. Biomed.*, vol. 16, no. 1, pp. 17–23, 2012.

[6] AIT Austrian Institute of Technology GmbH, “Keep in touch telemedical solutions.” [Online]. Available: https://kit.ait.ac.at. [Accessed: 19-Feb-2019].

[7] R. Modre-Osprian, “HERZMOBIL TIROL Presentation,” in *Vienna Healthcare Lectures*, 2018.

[8] A. der Heidt *et al.*, “HerzMobil Tirol network: rationale for and design of a collaborative heart failure disease management program in Austria,” *Wien. Klin. Wochenschr.*, vol. 126, no. 21, pp. 734–741, Nov. 2014.

[9] A. Guazzelli, “What is predictive analytics?,” *developerWorks, IBM Corporation*, 2012. [Online]. Available: https://www.ibm.com/developerworks/analytics/library/ba-predictive-analytics1/index.html?ca=drs-. [Accessed: 21-Feb-2019].

[10] J. Hu, A. Perer, and F. Wang, “Data Driven Analytics for Personalized Healthcare,” in *Healthcare Information Management Systems: Cases, Strategies, and Solutions*, C. A. Weaver, M. J. Ball, G. R. Kim, and J. M. Kiel, Eds. Cham: Springer International Publishing, 2016, pp. 529–554.

[11] C. Nyce, “Predictive Analytics White Paper,” *Am. Inst. Chart. Prop. Casualty Underwrit. Inst. Am.*, 2007.

[12] The MathWorks Inc., “Introducing Machine Learning,” in *Machine Learning ebook*, 2016.

[13] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data*

mining, inference and prediction, 2nd ed. Springer, 2009.

[14] J. J. Thomas and K. A. Cook, “Illuminating the path: The research and development agenda for visual analytics,” *IEEE-Press*, 2005.

[15] D. Keim, J. Kohlhammer, G. Ellis, and F. Mansmann, *Mastering the Information Age*. 2010.

[16] J. J. Thomas and K. A. Cook, “A Visual Analytics Agenda,” *IEEE Comput. Graph. Appl.*, vol. 26, no. 1, pp. 10–13, 2006.

[17] D. A. Keim, F. Mansmann, J. Schneidewind, J. Thomas, and H. Ziegler, “Visual Analytics: Scope and Challenges,” in *Visual Data Mining: Theory, Techniques and Tools for Visual Analytics*, S. J. Simoff, M. H. Böhlen, and A. Mazeika, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 76–90.

[18] D. Keim, G. Andrienko, J.-D. Fekete, C. Görg, J. Kohlhammer, and G. Melançon, “Visual Analytics: Definition, Process, and Challenges,” in *Information Visualization: Human-Centered Issues and Perspectives*, A. Kerren, J. T. Stasko, J.-D. Fekete, and C. North, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 154–175.

[19] D. Gotz and D. Borland, “Data-Driven Healthcare: Challenges and Opportunities for Interactive Visualization,” *IEEE Comput. Graph. Appl.*, vol. 36, no. 3, pp. 90–96, 2016.

[20] Y. Lu, R. Garcia, B. Hansen, M. Gleicher, and R. Maciejewski, “The State-of-the-Art in Predictive Visual Analytics,” *Comput. Graph. Forum*, vol. 36, no. 3, pp. 539–562, Jun. 2017.

[21] F. Heimerl, S. Koch, H. Bosch, and T. Ertl, “Visual Classifier Training for Text Document Retrieval.,” *IEEE Trans. Vis. Comput. Graph.*, vol. 18, no. 12, pp. 2839–2848, Dec. 2012.

[22] J. Choo, H. Lee, J. Kihm, and H. Park, “iVisClassifier: An Interactive Visual Analytics System for Classification Based on Supervised Dimension Reduction,” in *VAST 10 - IEEE Conference on Visual Analytics Science and Technology 2010, Proceedings*, 2010, pp. 27–34.

[23] M. S. Hossain, P. K. R. Ojili, C. Grimm, R. Müller, L. T. Watson, and N. Ramakrishnan, “Scatter/Gather Clustering: Flexibly Incorporating User Feedback to Steer Clustering Results,” *IEEE Trans. Vis. Comput. Graph.*, vol. 18, no. 12, pp. 2829–2838, 2012.

[24] C. D. Stolper, A. Perer, and D. Gotz, “Progressive Visual Analytics: User-Driven Visual Exploration of In-Progress Analytics,” *IEEE Trans. Vis. Comput. Graph.*, vol. 20, no. 12, pp. 1653–1662, 2014.

[25] D. Hayn *et al.*, “Predictive analytics for data driven decision support in health and care,” *it - Information Technology*, vol. 60. pp. 183–194, 2018.

[26] D. Kramer *et al.*, “Development and Validation of a Multivariable Prediction Model for the Occurrence of Delirium in Hospitalized Gerontopsychiatry and Internal Medicine Patients,” *Stud. Health Technol. Inform.*, vol. 236, 2017.

[27] D. Hayn *et al.*, “Development of Multivariable Models to Predict and Benchmark Transfusion in Elective Surgery Supporting Patient Blood Management,” *Appl. Clin. Inform.*, vol. 8, pp. 617–631, Jun. 2017.

[28] D. Hayn *et al.*, “Data Driven Methods for Predicting Blood Transfusion Needs in Elective Surgery,” *Stud. Health Technol. Inform.*, vol. 223, pp. 9–16, Jan. 2016.

[29] Y. Xie *et al.*, “Analyzing Health Insurance Claims on Different Timescales to Predict Days in Hospital,” *J. Biomed. Inform.*, vol. 60, Jan. 2016.

[30] Y. Xie *et al.*, “Predicting Days in Hospital Using Health Insurance Claims,” *IEEE J. Biomed. Heal. informatics*, vol. 19, Feb. 2015.

[31] Y. Xie, S. Neubauer, G. Schreier, S. Redmond, and N. H Lovell, “Impact of Hierarchies of Clinical Codes on Predicting Future Days in Hospital,” in *37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 2015, pp. 6852–6855.

[32] G. D. Clifford *et al.*, “AF Classification from a Short Single Lead ECG Recording: the PhysioNet/Computing in Cardiology Challenge 2017,” *Comput. Cardiol. (2010).*, vol. 44, p. 10.22489/CinC.2017.065-469, Sep. 2017.

[33] M. Kropf *et al.*, “Cardiac anomaly detection based on time and frequency domain features using tree-based classifiers,” *Physiol. Meas.*, vol. 39, no. 11, p. 114001, 2018.

[34] S. Few, “Dashboard Design: Beyond Meters, Gauges, and Traffic Lights,” *Bus. Intell. J.*, no. 10, pp. 18–24, 2005.

[35] A. Kwapien, “10 Dashboard Design Principles & Best Practices To Enhance Your Data Analysis,” 2016. [Online]. Available: https://www.datapine.com/blog/dashboard-design-principles-and-best-practices/. [Accessed: 23-Jan-2019].

[36] “User-Centered Design Basics.” [Online]. Available: https://www.usability.gov. [Accessed: 23-Jan-2019].

[37] The MathWorks Inc., “Matlab.” [Online]. Available: https://www.mathworks.com/products/matlab.html. [Accessed: 06-Feb-2019].

[38] The MathWorks Inc., “Ways to Build Apps.” [Online]. Available: https://de.mathworks.com/help/matlab/creating_guis/ways-to-build-matlab-guis.htm. [Accessed: 05-Feb-2019].

[39] W. Aigner, P. Federico, T. Gschwandtner, S. Miksch, and A. Rind, “Challenges of Time-oriented Data in Visual Analytics for Healthcare,” in *IEEE VisWeek Workshop on Visual Analytics in Healthcare*, 2012.

List of Figures