DEPARTMENT OF ELECTRICAL
AND COMPUTER ENGINEERING

JOÃO PEDRO ALVES OLIVEIRA
BSc in Electrical and Computer Engineering

ADOPTION OF BIG DATA AND AI METHODS TO MANAGE MEDICATION ADMINISTRATION AND INTENSIVE CARE ENVIRONMENTS

MASTER IN ELECTRICAL AND COMPUTER ENGINEERING

NOVA University Lisbon
March, 2023

DEPARTMENT OF ELECTRICAL
AND COMPUTER ENGINEERING

ADOPTION OF BIG DATA AND AI METHODS TO MANAGE MEDICATION ADMINISTRATION AND INTENSIVE CARE ENVIRONMENTS

JOÃO PEDRO ALVES OLIVEIRA

BSc in Electrical and Computer Engineering

Adviser: Carlos Manuel de Melo Agostinho
Principal Researcher, UNINOVA

Co-adviser: Vasco Delgado-Gomes
Researcher, UNINOVA

MASTER IN ELECTRICAL AND COMPUTER ENGINEERING

NOVA University Lisbon
March, 2023

Adoption of Big Data and AI methods to manage medication administration and intensive care environments

This document was created with the (pdf/Xe/Lua)LaTeX processor and the NOVAthesis template (v6.9.5) [1].

Dedicated to my family, girlfriend and friends.

Acknowledgements

I would like to start by thanking my supervisor and co-supervisor, Professor Carlos Agostinho and Vasco Gomes, for the opportunity to work in an area that I find attractive and for all the support provided throughout this dissertation, motivation and help in the most difficult moments.

A special thanks to the Uninova centre, with which I had the opportunity to collaborate during the dissertation development and to all the people involved whose feedback and support helped me with the thesis development. An additional thanks to Nazanin, Carlos Lopes and Paulo Figueiras, for the feedback provided, the suggestions given, and the analyses that helped me reach the final goal.

I would like to express my gratitude to Phillips Portugal for their support and resources provided without which the development of this thesis would not be possible.

A thank you to the friends I made throughout college, with whom I shared many battles and lived many good moments, which will remain in my memory.

To the friends I made during my high school days. You are the most enduring and fraternal friendships of mine, and I could not let this pass without leaving a note of gratitude for sticking by my side, for the laughs shared, for the moments lived and for your support in the most difficult moments. A simple thank you is not enough, but that is all I can say here.

To my paternal grandparents, who unfortunately are no longer with us, but whom I hold great esteem and will always have a special place in me.

To my maternal grandparents, who will celebrate this achievement, and have always taken care of me and supported me since I was a little child.

To my aunt and uncle, who have supported me during my whole life, with whom I will celebrate the end of this chapter of my life, and for the care and affection given.

To my little brother, who has to put up with my nonsense since little (but that's life) but who is one of my biggest companions and with whom I have shared many moments.

To my parents, for the sacrifices they made so that I could get where I am. For their unconditional support and concern, and for giving me the life I had. Thank you for everything.

Finally, a thank you to the person who has most recently entered my life, my favourite artist. My life has gotten better since we've been together and I couldn't end this without a word of gratitude. For the laughs shared, the moments lived, the support you gave me in this final stage and the motivation. Thank you for making my days better.

Thank you to everyone who has crossed my path and made me get where I am and who I am.

“I choose a lazy person to do a hard job. Because a lazy person will find an easy way to do it.” (Bill Gates)

Abstract

Artificial Intelligence (AI) has proven to be very helpful in different areas, including the medical field. One important parameter for healthcare professionals' decision-making process is blood pressure, specifically mean arterial pressure (MAP). The application of AI in medicine, more specifically in Intensive Care Units (ICU) has the potential to improve the efficiency of healthcare and boost telemedicine operations with access to real-time predictions from remote locations. Operations that once required the presence of a healthcare professional, can be done at a distance, which facing the recent COVID-19 pandemic, proved to be crucial.

This dissertation presents a solution to develop an AI system capable of accurately predicting MAP values. Many ICU patients suffer from sepsis or septic shock, and they can be identified by the need for vasopressors, such as noradrenaline, to keep their MAP above 65 mm Hg. The presented solution facilitates early interventions, thereby minimising the risk to patients.

The current study reviews various machine learning (ML) models, training them to predict MAP values. One of the challenges is to see how the different models behave during their training process and choose the most promising one to test in a controlled environment. The dataset used to train the models contains identical data to the one generated by bedside monitors, which ensures that the models' predictions align with real-world scenarios. The medical data generated is processed by a separate component that performs data cleaning, after which is directed to the application responsible for loading, classifying the data and utilising the ML model. To increase trust between healthcare professionals and the system to be developed, it is also intended to provide insights into how the results are achieved.

The solution was integrated, for validation, with one of the telemedicine hubs deployed by the European project ICU4Covid through its CPS4TIC component.

Keywords: Telemedicine, Artificial Intelligence, Machine Learning, Decision Support Systems

Resumo

A Inteligência Artificial (IA) é muito útil em diferentes áreas, incluindo a saúde. Um parâmetro importante para a tomada de decisão dos profissionais de saúde é a pressão arterial, especificamente a pressão arterial média (PAM). A aplicação da IA na medicina, mais especificamente nas Unidades de Cuidados Intensivos (UCI), tem o potencial de melhorar a eficiência dos cuidados de saúde e impulsionar operações de telemedicina com acesso a previsões em tempo real a partir de locais remotos. As operações que exigiam a presença de um profissional de saúde, podem ser feitas à distância, o que, face à recente pandemia da COVID-19, se revelou crucial.

Esta dissertação apresenta como solução um sistema de IA capaz de prever valores de PAM. Muitos pacientes nas UCI sofrem de sepse ou choque séptico, e podem ser identificados pela necessidade de vasopressores, como a noradrenalina, para manter a sua PAM acima dos 65 mm Hg. A solução apresentada facilita intervenções antecipadas, minimizando o risco para doentes.

O estudo atual analisa vários modelos de machine learning (ML), e treina-os para preverem valores de PAM. Um desafio é ver o desempenho dos diferentes modelos durante o seu treino, e escolher o mais promissor para testar num ambiente controlado. O dataset utilizado para treinar os modelos contém dados idênticos aos gerados por monitores de cabeceira, o que assegura que as previsões se alinhem com cenários realistas. Os dados médicos gerados são processados por um componente separado responsável pela sua limpeza e envio para a aplicação responsável pelo seu carregamento, classificação e utilização do modelo ML. Para aumentar a confiança entre os profissionais de saúde e o sistema, pretende-se também fornecer uma explicação relativa à previsão dada.

A solução foi integrada, para validação, com um dos centros de telemedicina implantado pelo projeto europeu ICU4Covid através da sua componente CPS4TIC.

Palavras-chave: Telemedicina, Inteligência Artificial, Machine Learning, Sistemas de Apoio à Decisão

Contents

List of Figures

List of Tables

Acronyms

AI	Artificial Intelligence
AIM	Artificial Intelligence in Medicine
ANN	Artificial Neural Network
API	Application Programming Interfaces
CDSS	Clinical Decision Support Systems
CEP	Complex Event Processing
CNN	Convolutional Neural Network
CPS4TIC	Cyber-Physical System for Telemedicine and Intensive Care
DNN	Deep Neural Network
DSRPAI	Dartmouth Summer Research Project on Artificial Intelligence
EHR	Electronic Health Records
FHIR	Fast Healthcare Interoperability Resources
FN	False Negatives
FP	False Positives
HDFS	Hadoop Distributed File System
HL7	Health Level 7
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ICU	Intensive Care Units
IoT	Internet of Things
IP	Internet Protocol
JSON	JavaScript Object Notation

KNN	K-Nearest Neighbor
LR	Logistic Regression
MAP	Mean Arterial blood Pressure
MCDM	Multi-criteria decision making
MIMIC	Multi-parameter Intelligent Monitoring in Intensive Care
MIMIC-II	Multiparameter Intelligent Monitoring in Intensive Care II
MIMIC-III	Medical Information Mart for Intensive Care III
ML	Machine Learning
NHIC	National Health Insurance Corporation
NoSQL	Not Only SQL
PDMS	Patient Data Management Systems
PIIC iX	Philips IntelliVue Information Center iX
PPG-BP	photoplethysmography-blood pressure
ReLU	Rectified Linear Unit
SHAP	SHapley Additive exPlanations
SOFA	Sequential Organ Failure Assessment
SQL	Structured Query Language
SVM	Support Vector Machine
TCP	Transmission Control Protocol
TN	True Negatives
TP	True Positives
URL	Uniform Resource Locators
VPN	Virtual Private Network
XAI	Explainable Artificial Intelligence
XML	Extensible Markup Language

1 Introduction

Computational processing power has been constantly increasing and can potentially help humans in many different areas. This constant upgrade enabled machines to match or even surpass humans in some cognitive tasks. As a result, the impact that Artificial Intelligence (AI) can have on everyday life is highly promising [2].

According to Haenlein and Kaplan it is hard to pinpoint the roots of AI however, it can be traced back to the 1940s. It was from that time onwards, that these new technologies began to be discussed and developed. In 1942, an American science fiction writer, Isaac Asimov, published a short story, Runaround, about a robot. This story ended up inspiring some scientists in the robotics area. At the same time, Alan Turin developed a machine that was able to decipher the Enigma code used by the Germans during the second world war. He ended up publishing a seminal article in 1950, "Computing Machinery and Intelligence", where he described how to test and develop an intelligent machine [2].

The word "Artificial Intelligence" was coined in 1956 during the workshop Dartmouth Summer Research Project on Artificial Intelligence (DSRPAI) at Dartmouth College in New Hampshire, hosted by Marvin Minsky and John McCarthy: two computer scientists [2].

The field of AI has been evolving since then, as well as its definition. Even though the meaning of AI revolves around a machine's ability to demonstrate human capabilities, scientists have not yet found a unique consensus. In another article, Kaplan and Haenlein defined AI as "a system's ability to correctly interpret external data, to learn from such data, and to use those learnings to achieve specific goals and tasks through flexible adaptation"[3] whereas Poole and Mackworth described it as "the synthesis and analysis of computational agents that act intelligently"[4]. The foundation of the presented definitions is that AI involves the study and development of agents to achieve goals. Some AI models are very flexible and can even learn through experience, improving performance without human intervention. It might be said that these types of programs are more intelligent than programs that just follow a fixed set of rules that guides into a specific set of actions. Because of its advantages, AI is currently being implemented in many areas of relevant social impact, including medicine and health related domains.

Medicine evolved heavily with the help of AI, especially with the arising of Machine Learning (ML) and other AI techniques. Artificial Intelligence in Medicine (AIM) can create personalised medicine. According to Kaul, Enslin, and Gross [5], AIM "may improve diagnostic accuracy, improve efficiency in provider workflow and clinical operations, facilitate better disease and therapeutic monitoring, and improve procedure accuracy and overall patient out-comes". One example of an AIM application is IBM's artificial intelligence business unit Watson for Health. It analyses large amounts of data and, by efficiently giving relevant data to clinicians, it helps to improve informed medical decisions and care experiences [6]. For instance, at Minnesota's Mayo Clinic Watson for Health is assisting doctors in choosing the best treatment for their patients and is supporting doctors at 16 leading cancer institutes to provide individualised therapy options to the patients [7].

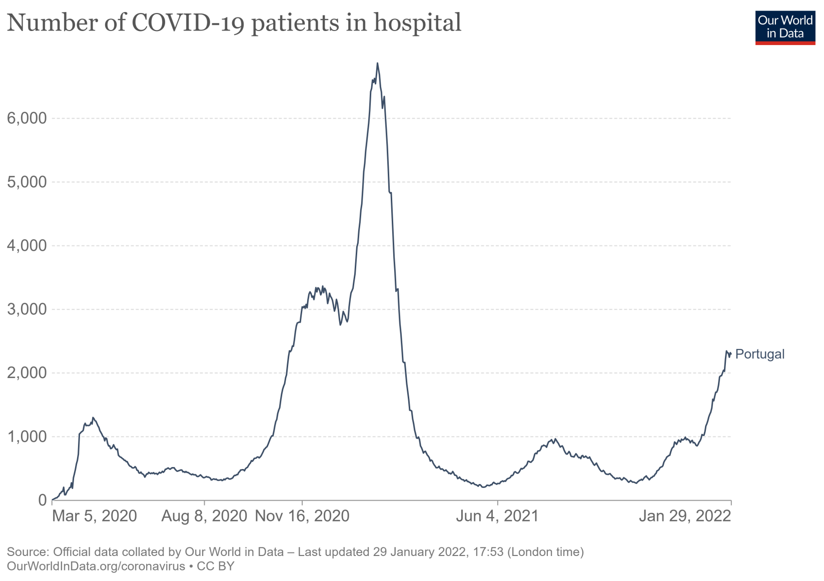
AI applications tend to grow in the medical field with the principal objective of improving people's lives. This dissertation aims to train an AI algorithm that, through the analyses of data generated by sensors present in a Intensive Care Units (ICU) environment, can predict future values of Mean Arterial blood Pressure (MAP), giving valuable insights to healthcare professionals.

1.1 Motivation

With the technological progress enabling better data analysis and processing in real-time, with proven results in many areas that range from heavy industry, such as car manufacturing, to professional sports with the analysis of multiple scenarios and statistics, it is necessary to take advantage of these resources to also improve the health-related areas, specifically, hospitals' ICU where decisions need to be taken very quickly, and the more data is available the better the decision can be. The available resources have the capability of improving patient care and helping clinician's work. This would allow the ICU to be less overcrowded in critical situations while also providing better treatment, hence improving the healthcare service to citizens.

A properly designed program, capable of easing and decreasing human work and error, respectively, can be a solution for these problems and many others. Furthermore, it can also help physicians to make better-informed decisions. According to an article published in The Harvard Gazette, "in May 2019, researchers at Google and several academic medical centres reported an AI designed to detect lung cancer that was 94 percent accurate, beating six radiologists and recording both fewer false positives and false negatives"[8]. In another article, O'Meara [9] states that China developed an intelligent algorithm that has been put into use to reduce the burden of radiologists during the pandemic, capable of detecting multiple respiratory diseases, including COVID-19. Both examples show that AIM can be a valuable asset. On top of that, knowing that machines can outperform humans without the side effects (e.g., fatigue), a human-machine collaboration would have a great impact since it could be more efficient, less exhausting, and help reduce costs.

Additionally, the COVID-19 pandemic outbreak in 2019 brought novel crisis management scenarios and many adversities to society. It was clear that healthcare systems had enormous flaws, with hospitals being constantly congested. This proved to be a major problem not only because of doctors' work and information overload, which lead to burnout situations, but also because of non-covid patients who needed to have appointments and could not have them due to the situation. According to Jornal de Notícias [10], in Portugal, hospitals were overpopulated with patients in ICU and outside. In the next Figure, 1.1, it is possible to observe two charts: on top, the number of COVID-19 patients hospitalised, and on the bottom, the number of COVID-19 patients in the ICU.



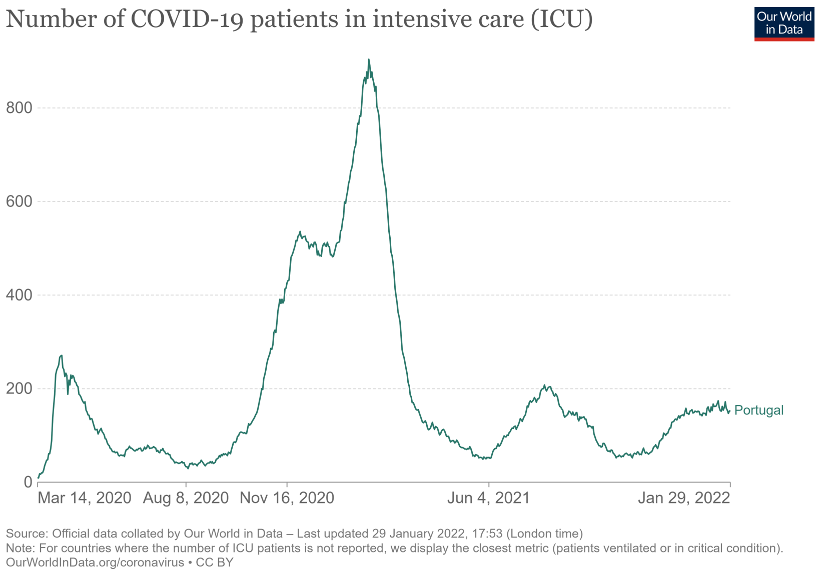


Figure 1.1: Covid Patients in hospital [11] and ICU [12].

This data shows many concerns and proves that, despite the advancements made in technological areas, humans were not properly prepared for a pandemic. Suddenly, remote working started to be used, and when equipped with the necessary technology, also medical staff benefited from it, promoting the execution of healthcare services with a reduced physical presence of healthcare providers next to the patient, hence reducing the risk of COVID propagation. This technology is called telemedicine (is also referred to as telehealth or e-medicine) and provides the ability to monitor patients remotely and have consultations virtually (through the computer and at long distances). Figure 1.2 depicts a telemedicine context where it is possible to have in-house consultations or do remote monitoring.

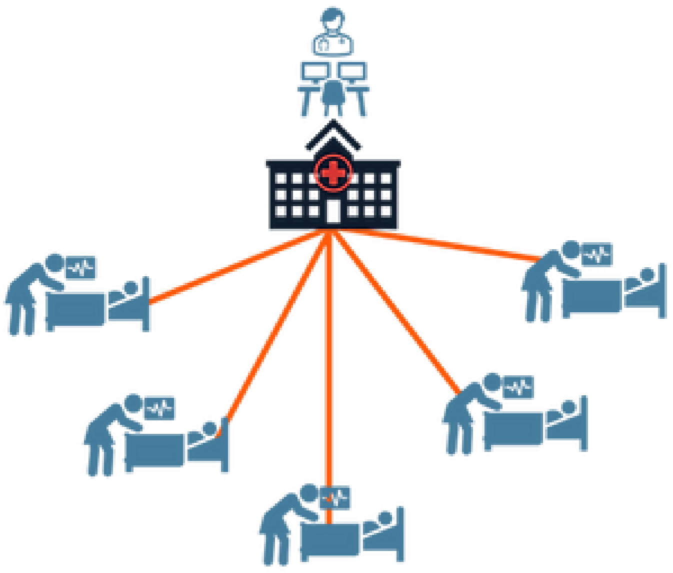


Figure 1.2: In-house consultations or remote monitoring [13].

The potential benefits of telemedicine associated with AI were on the genesis of the ICU4Covid project [14], which started in January 2021 and ended in December 2022, intending to build an effective solution to fight against the difficulties caused by COVID-19. It is an innovative solution that uses the concept of telemedicine to overwatch patients found in the ICU. It consists of a new set of AI-enabled technologies that are placed at the patient's bedside and with a real-time connection to an observation centre (called the cockpit) to enable doctors to observe the patient's development from a distance, and still make informed decisions. This model reduces the face-to-face contact between medical personnel and patients but decreases the risk of infections, thus helping maintain the performance of medical staff during pandemics [15].

One of the main objectives of the ICU4Covid project is to develop an Internet of Things (IoT) Hub that provides constant monitoring of a patient in the ICU. This dissertation aims to contribute to this objective by establishing a connection between the IoT Hub and medical devices that collect real-time patient data. By enabling remote monitoring, healthcare professionals can access and analyse vital patient data, including bio-signals such as vital signs, heartbeat, and blood pressure. If the Hub detects any anomalies or irregularities in the bio-signals detected, it uses sophisticated and predefined AI algorithms to provide some feedback to doctors and nurses, and even automate some actions. Leveraging AI technology, the system can recognise patterns, cross-reference with similar cases, and identify potential health issues affecting the patient. This valuable information enhances the reliability of reports provided to the medical staff, assisting them in making well-informed decisions regarding patient care. Furthermore, the IoT Hub can even actuate on its own, keeping medication administration within a limited threshold.

1.1.1 Problem Identification and Integration with ICU4Covid

Currently, the ICU has an absence of tools to support clinicians' decision-making, and medication administration requires the intervention of a healthcare professional to adjust the dosage, which could be adjusted by AI systems that follow a specific set of rules based on patient observations [16]. Moreover, telemedicine is not a common practice, apart from the fact that some hospitals do not have the necessary equipment to adopt it, which is desirable in critical situations.

These issues have been noticed during the COVID-19 pandemic. The potential of AI and the needs for telemedicine were factors that encouraged this dissertation. The proposed solution, and the objective of this dissertation, is to develop a system capable of analysing multiple sensor information and, using AI methods, providing relevant information to healthcare professionals. Monitoring could be done remotely (telemedicine), and decision-making handled by a healthcare professional, only with better and faster information. The system may even automate some actions, but only with the authorisation of a professional could the system act autonomously.

To conclude, I believe that the system to be developed can be applied in other areas and improve healthcare systems.

1.2 Research Methodology

To ensure success, every research work needs a research methodology. The research methodology adopted after the elaboration of the identified problem in this dissertation is presented in Figure 1.3.

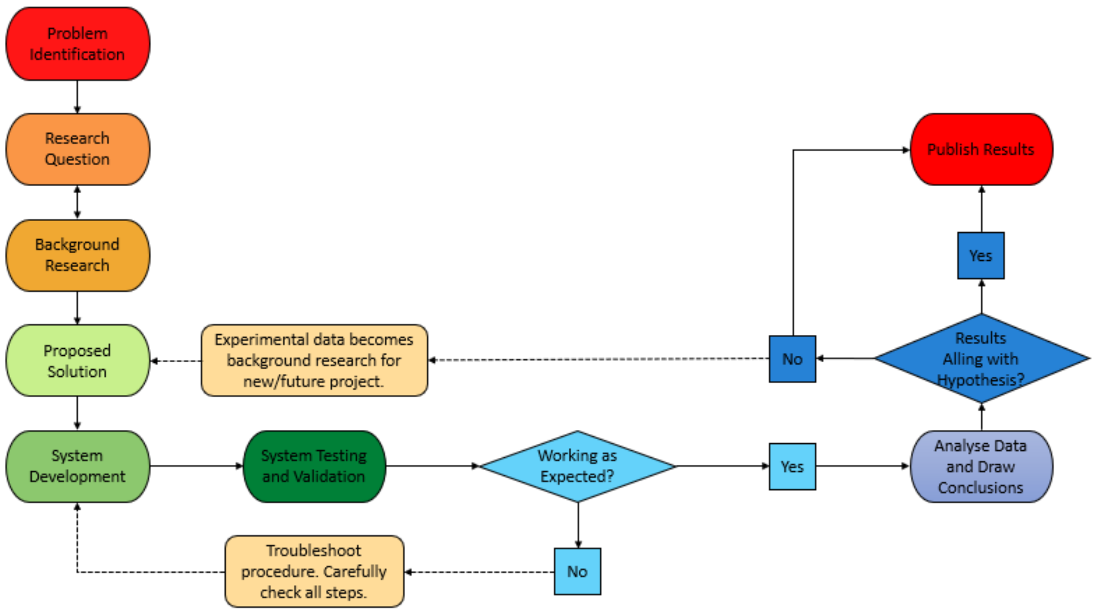

Problem Identification
Research Question
Background Research
Proposed Solution
Experimental data becomes background research for new/future project.
System Development
System Testing and Validation
Working as Expected?
Yes
No
Troubleshoot procedure. Carefully check all steps.
Analyse Data and Draw Conclusions
Results Alling with Hypothesis?
Yes
No
Publish Results

Figure 1.3: Scientific research methodology.

With an eight-step configuration, the research methodology consists in the following steps:

1. **Problem identification:** the first step consists of the identification of a meaningful problem, which led to the start of this research. The problem identification was described in the Section 1.1, Motivation.

2. **Research question:** in this step it is defined a research question. The research question is presented in the next Section (1.2.1 Research Question), and it is important because it helps to guide the project.

3. **Background research:** it is collected important and relevant background information about the topic. The information gathered is in the State of the Art, presented in Chapter 2.

4. **Proposed solution:** through the study of the information collected in the previous step, it is proposed a solution against the problem identified. It is also necessary to study the technologies available to use. This step is presented in Chapter 3

5. **System development:** start developing the proposed solution and carry out the necessary tests for its validation. This is presented in Chapter 4.

6. **System validation:** analysis of the results from the previous phase and the proposed solution is evaluated. If the solution is valid, the results are discussed, and we carry on to the next step. If not, it is necessary to go back a few steps and change the solution. As well as the previous step, this is presented in Chapter 4

7. **Conclusions:** conclusions are made whether the system development was a success or not and ends by stating what future improvements can be done to the developed work. It is presented in Chapter 5.

8. **Publish results:** the final step is to make a final review of the project and prepare the dissertation for publication. Additionally, a paper related to the dissertation has been accepted for presentation at the 29th ICE IEEE/ITMC Conference in Edinburgh. This recognition provides an opportunity to share the research findings and engage in scholarly discussions with fellow researchers and experts in the field [17].

1.2.1 Research Question and Hypothesis

This dissertation aims to contribute to the development of an IoT Hub that through the application of AI and Big Data techniques can help manage an ICU environment. So, the question that the dissertation proposes to answer is:

By applying AI techniques in ICU using patient's clinical data, is it possible improve intensive care and minimise the adversities exposed by COVID-19?

Initially, this dissertation had a very broad vision about what was the aim of this study, however, during the development of it, the focus started to be on the development of an AI algorithm to reliably predict future values of blood pressure, specifically MAP values, based on information generated by ICU sensors. Based on that a more precise research question was made, which is:

Collecting data generated by ICU sensors and feeding them to an AI model, can reliably predict MAP values, helping healthcare professionals in the decision-making process?

Based on the research question, this dissertation is guided by the following hypothesis:

The analysis of data generated in ICU by an AI model can reliably predict future MAP values, helping healthcare professionals' decision-making

This statement will undergo testing and validation throughout the course of this dissertation.

1.3 Document Overview

The presented document has the following structure:

1. **Introduction:** it is where the topic, motivation and research question are presented.

2. **State of the Art:** based on previously done studies, published articles or books and some websites, it is provided an overview about some relevant topics related to the development of the dissertation.

3. **Proposed Solution:** based on the reviewed technologies, it aims to present the proposed solution for the ongoing work.

4. **Implementation:** it explains the developed worked and the obtained results. Implementation details, encountered problems and final results are all presented in this section.

5. **Conclusion:** the final conclusions are presented in this chapter as well as future improvements of the developed work.

2 State of the Art

As stated before, AI is currently being implemented in the medical field. Some AI-related areas are ML and Big Data. These concepts are interconnected and rely on information collected from multiple medical devices. By combining several techniques related to ML and Big Data, such as Deep Learning and Big Data Analytics, it is possible to create a Clinical Decision Support Systems (CDSS). A CDSS can deliver reliable feedback to medical staff and become a solution for real-time monitoring in ICU [7].

In this chapter, several concepts related to the applications of AI in medicine are presented. They will provide knowledge to develop a solution to the research question.

2.1 Machine Learning

According to Bartneck et al. [18], the first-ever ML program was designed in 1950 by Alan Turing, who developed a program, called the "Turing Test", intending to to speak to a person and see if it could fool the person into thinking they were talking to another person.

Nowadays, with the constant increase in computational power, more sophisticated programs can be designed and so, more companies are investing in ML. According to MIT [19], "a 2020 Deloitte survey found that 67% of companies are using machine learning, and 97% are using or planning to use it in the next year". ML provides immeasurable tools for a company to develop and is now spreading across many industries.

ML is considered a sub-category of AI, where algorithms can autonomously learn through the data and information acquired without needing human intervention. Some algorithms can even change and improving on its own [18].

Over the years, many real-world problems have been solved using ML methods. However, for a program to be independent and provide reliable and accurate feedback, it is necessary to train and come up with a model: two concepts that are going to be explained in this section.

2.1.1 Models

A ML model is a trained file (the next section, Section 2.1.2, explains the training process) that analyses large amounts of data to recognise patterns. A model is trained over a set of data, providing it with an algorithm that can be used to reason over and learn from data. The result of training is a mathematical representation of patterns hidden in data. That mathematical expression can be applied to new unseen data and make predictions about it [20].

The idea of a model-based approach in machine learning is to create a model specific to each new application. The hypothesis may come in varieties of knowledge representation forms, such as decision trees, equations, probabilistic models, graphical models, etc. Normally, a division is made between symbolic and sub-symbolic forms however, despite this division, nowadays there are symbolic applications with sub-symbolic characteristics and vice-versa [21].

Symbolic methods are easily readable by a human. A characteristic of symbolic methods is that they can clarify how they got to a conclusion. They provide a human-understandable computational flow that makes them easier to understand, explain and control. In particular, it is easy to alter the knowledge base by inserting or removing some knowledge units. Moreover, it is possible to transfer knowledge to closely related applications [22].

Contrary to symbolic methods, sub-symbolic methods do not have a clear interpretation. They establish relations between input and output variables, and, in this approach, knowledge is represented by numerical patterns. Although they are highly dependent on training data, sub-symbolic methods are more robust against noisy and missing data, and generally have better performances than symbolic methods. However, one problem with sub-symbolic methods is the lack of interpretability. This represents a significant barrier in fields where justifications and interpretations are crucial [22], such as medicine.

2.1.2 Training

A ML training process involves providing a machine learning algorithm with some training data. Then, the algorithm must find patterns in the training data and map the input attributes to the output. One thing to keep in mind is that the input-output answer must be correct in the training data [23].

It is possible to divide the training data into three sets. The first set is the training set, the second is the validation or development set, and the last is the testing set [24].

As the name implies, the training set is used to train the model and the test set is used to verify how well the trained model performs when provided with new unseen data. By evaluating the model using the test set, it is possible to get an estimate of its error and compare it with the training error. If it so happens that the training error is low, but the testing error is high, it means the model is overfitting the training data (too adapted to the training set). One solution is to create the mentioned validation set. To use it, multiple models are trained with multiple hyperparameters in a reduced training set and the model that best performs on the validation set is selected and trained again using the whole training set. Lastly, the chosen model's predictive power is verified using the testing set. This can be useful, for example, when working with neural networks since it can be used to find the optimal size of the network, the number of hidden layers, and other characteristics [24].

TensorFlow [25] is an example of an open-source powerful, developed by Google tool used to train models. TensorFlow provides a collection of workflows to develop and train machine learning algorithms, using Python or JavaScript. TensorFlow runs in a web server or a Node.js environment.

TensorFlow APIs provide methods to create multi-dimensional arrays (tensors) and implement standard mathematical operations on them. Each tensor represents a data input, and the goal is to obtain the mathematical expression that best relates the inputs to the outputs. To obtain the most appropriate expression, TensorFlow will perform multiple iterations (with new inputs and respective outputs) where it will adjust the value of the input weights and the bias value. As stated in Section 2.1.1, the weight of an input is related to the importance of the inputs. The more important an input is, the greater its weight. The bias value is used to make little adjustments and helps the models to be more efficient. Both values only change during the training process and at the end of the training, the model should be acquired with the best weight values.

One more important tool is Keras [26]. Keras is another open-source tool, written in Python, that offers simple and consistent APIs to provide the user with a more comfortable neural network implementation. Keras was recently integrated into TensorFlow so, users can now access it via the tf.keras module.

2.1.2.1 Data Labelling

Data labelling is part of the preprocessing stage of developing a model. It is the process of identifying meaningful and informative raw data and labelling it to give context so that the machine learning models can make accurate predictions. To label data, a human has to make a judgement about a given piece of unlabelled data and tag the necessary aspects. Labelled data makes the training process more efficient and simple, and it is used in supervised learning [27].

On the other hand, unlabelled data is used in unsupervised learning and can discover new patterns in data, allowing for new categories when labelling.

Supervised and unsupervised learning are explained in the next section (Section 2.1.3).

2.1.3 Types of Learning

Typically, ML is categorised into three main types of learning: supervised learning, unsupervised learning, and reinforcement learning [18]. This section makes an overview about the different types of learning.

2.1.3.1 Supervised Learning

Supervised learning is applied when the data comes in the form of inputs (x) and the desired outputs (f(x)). The algorithm is required to learn and needs to generate a new function (h(x)) that best approximates the original function (f(x)) by analysing several input-output examples of the original (f(x)) [21]. This approach can be divided into two main categories: classification and regression learning.

Classification and Regression are two different predictive models. Predictive modelling "is the problem of developing a model using historical data to make a prediction on new data" [28] where the answer is not known.

In classification, during the training process, a computer program learns to categorise data into different data classes while considering several factors. The task of a classification algorithm is to find the appropriate function that relates inputs (x) to discrete outputs (y) [21]. In other words, classification aims to classify discrete values such as male and female, infected or not infected, or, as shown in Figure 2.1, distinguish circles, squares, and triangles.

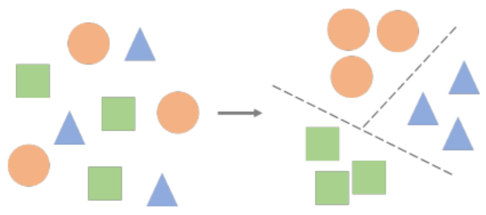


Figure 2.1: Classification model [29].

Regression is a technique that discovers correlations between dependent and independent variables, somewhat like a mathematical expression. The objective of a regression algorithm is to find the function that relates the inputs (x) to the continuous outputs (y) [21]. It helps to predict values such as price or salary, as shown in Figure 2.2.

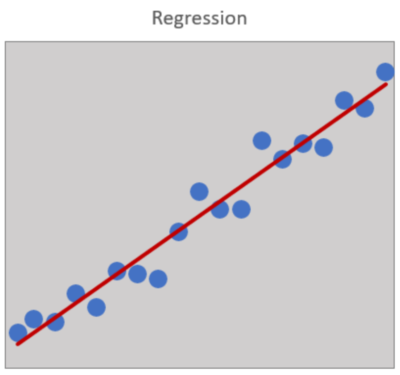


Figure 2.2: Regression model.

Both are supervised procedures and, based on previously trained data, they learn how to label and interpret new unseen data [21].

A study made by Hashi, Uz Zaman, and Hasan [30] compares the performance of two data mining classification predictive models: C4.5, a decision tree technique, and K-Nearest Neighbor (KNN) (these algorithms are further explained in Section 2.1.5). The objective of both models is to correctly identify patients with diabetes. For the training and testing phase, a dataset with 768 entries, where 500 are negative results and 268 positives, was used. A 70:30 percentage split was made to evaluate the classification result. Of the 768 inputs, 538 were randomly chosen to train the model and the rest were used for the testing phase. The final analysis showed promising results. This system gives benefits to healthcare providers, regarding disease diagnosis. The system achieves 100% accuracy in the training phase using the decision tree algorithm (the KNN was not specified) and 90.43% and 76.96% accuracy in the testing phase using C4.5 and KNN, respectively.

2.1.3.2 Unsupervised Learning

Unsupervised learning is applied when an algorithm looks for patterns in unlabelled data because there is no labelled data available. Data is available only in the form of input and there is no corresponding output. Such algorithms model non-implicit patterns so they can learn about their characteristics. One of its main examples is clustering [29].

Clustering is an unsupervised task, whose aim is to divide the population or data points into different groups called clusters where all data points within a group are similar. Data points should be internally similar and externally different. This method is based on an important notion which is similarity [21], and a cluster analysis allows one to discover hidden structures and patterns in data. Also, there are multiple approaches to clustering. Figure 2.3 exemplifies a prototype-based clustering method, where identical elements are placed in the same group.

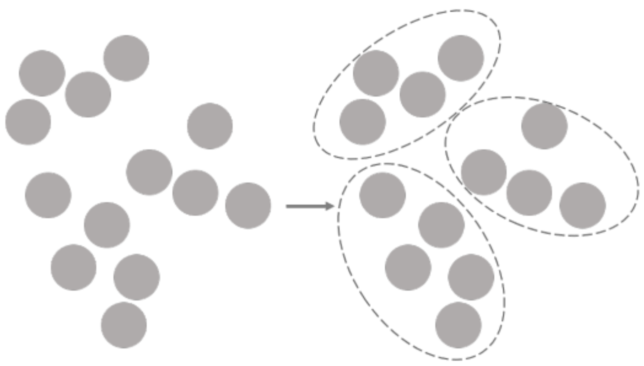


Figure 2.3: Prototype-based clustering method [29].

Ribeiro et al. [31] noticed that one main problem found in ICU is unplanned admissions. There is a lack of technologies to identify who needs to enter. Therefore, they applied clustering techniques to data extracted from INTCare, a Clinical Decision Support System used in the ICU of Centro Hospitalar do Porto in Portugal. They identified several useful factors which helped to generate patterns that would allow for a better understanding of a patient's state during admission. This enabled clinicians to quickly determine whether or not a patient had similar values to those reported by the clusters. Another aim of the project was to alert clinicians if a patient has an important detail to consider, for example, surgery or transplant.

During the modelling phase, they used a dataset composed of thirty-three attributes yet, for the demonstration part, they reduced the attributes to four: TYPEOFADMISSION, TYPEOFADMISIONSURGERY, RISK, and TRANSPLANTED. They ended up concluding that RISK was the only feature that had a negative impact on the results.

According to the paper, the cluster cannot ensure who is admitted to the ICU but they provide important information about who needs to be. Finally, the model and results obtained were implemented to strengthen INTCare.

2.1.3.3 Reinforcement Learning

Reinforcement learning involves exploring an unknown environment to attain an objective. Reinforcement learning is based on the hypothesis that every decision has either a penalty or a reward and, to achieve a goal, a system must maximise the cumulative reward. These are closed-loop problems because every action taken influences future inputs. In some cases, actions not only affect immediate rewards but can also affect future situations, and consequently all those that follow. Furthermore, unlike many forms of machine learning, the system is not taught the actions to do but must find the decisions that offer the best outcome by trying them out. These characteristics - being a closed-loop, actions having future repercussions, and not having direct instructions - are unique features of reinforcement learning [32]. Figure 2.4 represents the reinforcement learning logic.

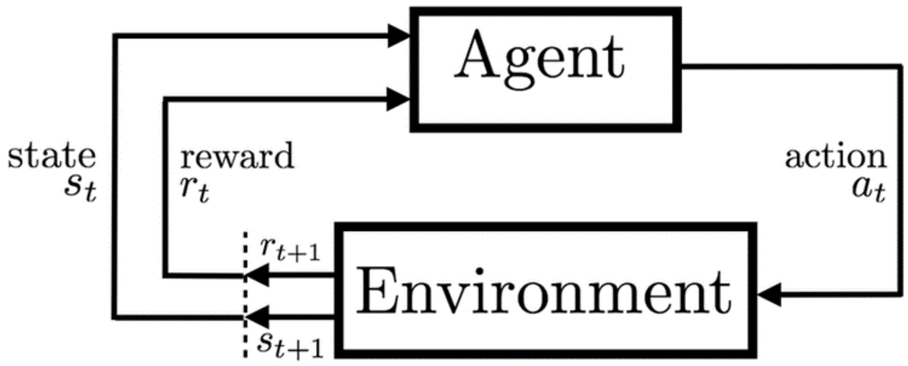


Figure 2.4: Reinforcement Learning [33].

2.1.4 Deep Learning

As stated before, ML is a subcategory of AI. Similarly, Deep Learning can be considered a subcategory of ML. Deep Learning is a technique used to mimic the way of learning and thinking by humans. To do that, Deep Learning makes use of Artificial Neural Network (ANN), more precisely deep neural networks [34].

An ANN is a computational model inspired by the structure and function of biological neural networks, particularly the human brain. ANNs are composed of a large number of interconnected processing elements, known as neurons or nodes, that work in parallel to process information [34]. ANNs are algorithms that process information by mimicking the behaviour of the human brain. The network comprises multiple layers, with the first layer receiving the input variables. It is followed by one or more hidden layers, where the network performs calculations, and it ends with the output layer [34]. Figure 2.5 illustrates a representation of a simple neural network.

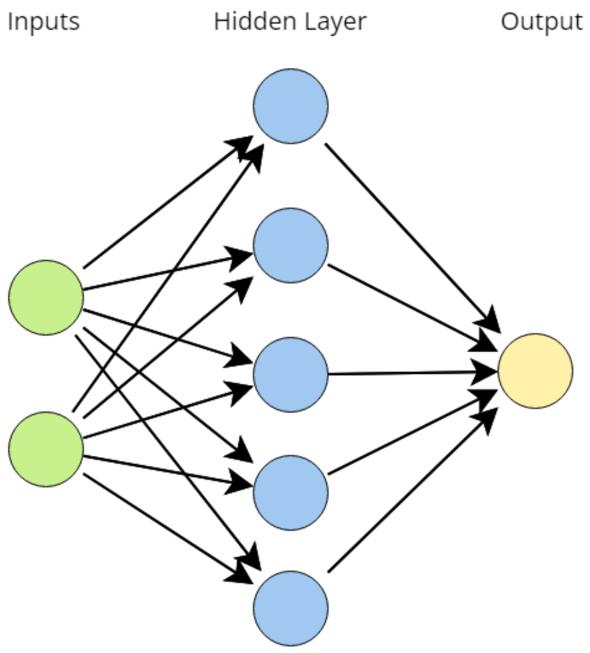


Figure 2.5: Simple neural network.

The input layer is the first layer of the ANN, where the input data is introduced into the network. The input data is then passed through the hidden layers, which perform calculations on the input data. Each connection between neurons is associated with a weight value, which reflects the importance of that connection. The more important an input is, the greater its weight. During the training process, these weight values are adjusted to optimise the network's performance [34].

In addition to the weight values, a bias value can be added to adjust the output of the neurons and help the network to be more efficient. After the calculations, the network applies an activation function to the resulting values. The activation function transforms the previously calculated value into another value normally comprehended between 0 and 1. The activation function is critical as it introduces non-linearity to the network and allows it to model complex relationships between inputs and outputs [34]. An example of an activation function is a sigmoid function, shown in Figure 2.6, where x is equal to the sum of all the inputs (multiplied by their weights) plus the bias value.

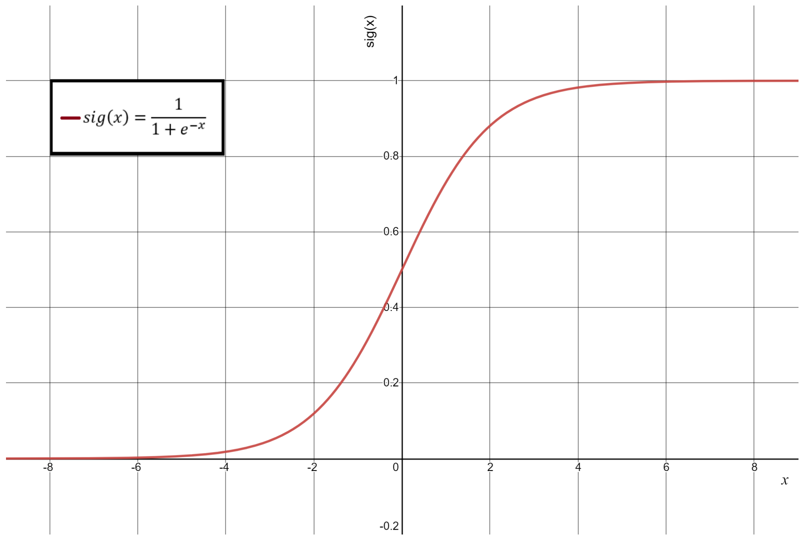


Figure 2.6: Sigmoid activation function.

The final layer of the ANN is the output layer, which produces the desired output based on the input data and the learned relationships between the input and output data [34]. The described operations are represented in Figure 2.7.

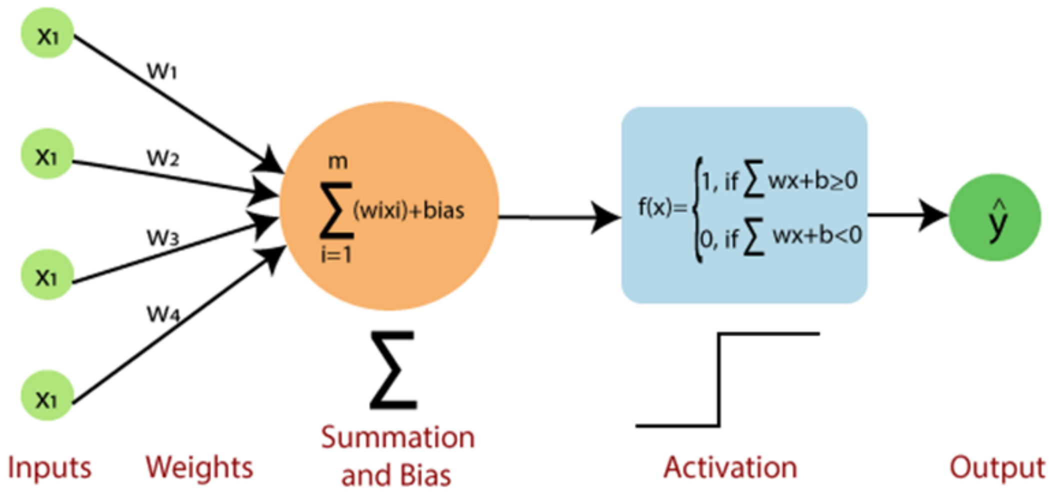

X1
X1
X1
X1
W1
W2
W3
W4
$\sum_{i=1}^{m}$(wixi)+bias
f(x)= {1, if ∑ wx+b≥0 ; 0, if ∑ wx+b<0
ŷ
Σ
Inputs
Weights
Summation and Bias
Activation
Output

Figure 2.7: Operation of a neural network [35].

ANNs are the core of Deep Learning and the process of a Deep Neural Network (DNN) is identical. The difference between a simple neural network and a DNN is the number of hidden layers. The number of hidden layers of a DNN is bigger, which makes the complexity of a neural network increase, improves the expression power, and the performance of these methods [34]. Figure 2.8 shows an example of a DNN.

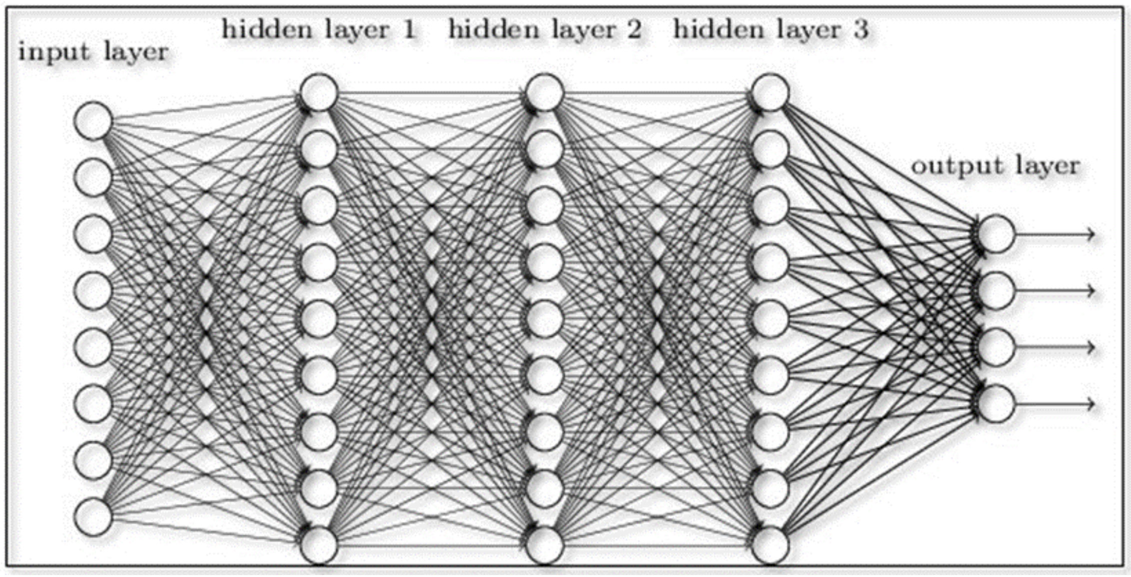


Figure 2.8: Deep neural network [36].

DNN are very promising and there have been a development of new technologies that use this technique to improve the ICU. Critically ill patients in the ICU suffer, or may develop, life-threatening conditions. Early recognition is crucial and may identify patients in need of life-saving interventions as well as help in the decision-making process [37].

Since 1990, Sequential Organ Failure Assessment (SOFA) has been implemented in the daily monitoring of the ICU. It was firstly designed to provide population insights, however, its scope has expanded significantly and is currently utilised as a key criterion of sepsis on an individual level [38].

SOFA is based on a scoring system. It gives a score value, between 0 and 4, for each of the respiratory, cardiovascular, hepatic, coagulation, renal and neurological systems. The higher the score is, the worse is the organ dysfunction. The sum of these scores yields the overall score, which can be used to determine the illness severity and prevent mortality [38]. Even though SOFA gives a reasonable evaluation of a patient, it lacks flexibility for each component score and many of its variables are frequently absent in Electronic Health Records (EHR). Furthermore, as time passes, the SOFA accuracy worsens since the scores are given at the time of entrance and it does not consider the patient evolution.

Shickel et al. [37] propose an innovative solution to face SOFA faults. The solution proposed is called DeepSOFA and uses deep learning to analyse clinical data in the ICU. To compare the accuracy of the new model, two more models were used: the traditional SOFA and the Bedside SOFA. By observing the results, it is possible to conclude that DeepSOFA performed better than the other models. After patients were admitted to ICU, DeepSOFA was able to follow the patients' development better and have a better accuracy. The following figure (2.9) shows the "Model accuracy for prediction windows of increasing time from hospital discharge or death. (A,B) Externally validated DeepSOFA, Bedside SOFA, and Traditional SOFA score accuracy in predicting in-hospital mortality, expressed as area under the receiver operating characteristic curve (AUC) for 100hours preceding death or hospital discharge. (C,D) Externally validated DeepSOFA accuracy for individual models corresponding to variable sets derived from SOFA organ system classification for the 100hours preceding death or hospital discharge. Shaded regions represent 95% confidence intervals based on 100 bootstrapped iterations. Columns specify the validation cohort. DeepSOFA model validated in UFHealth (A,C) was trained using MIMIC, and DeepSOFA model validated in MIMIC (B,D) was trained using UFHealth. SOFA: Sequential Organ Failure Assessment".

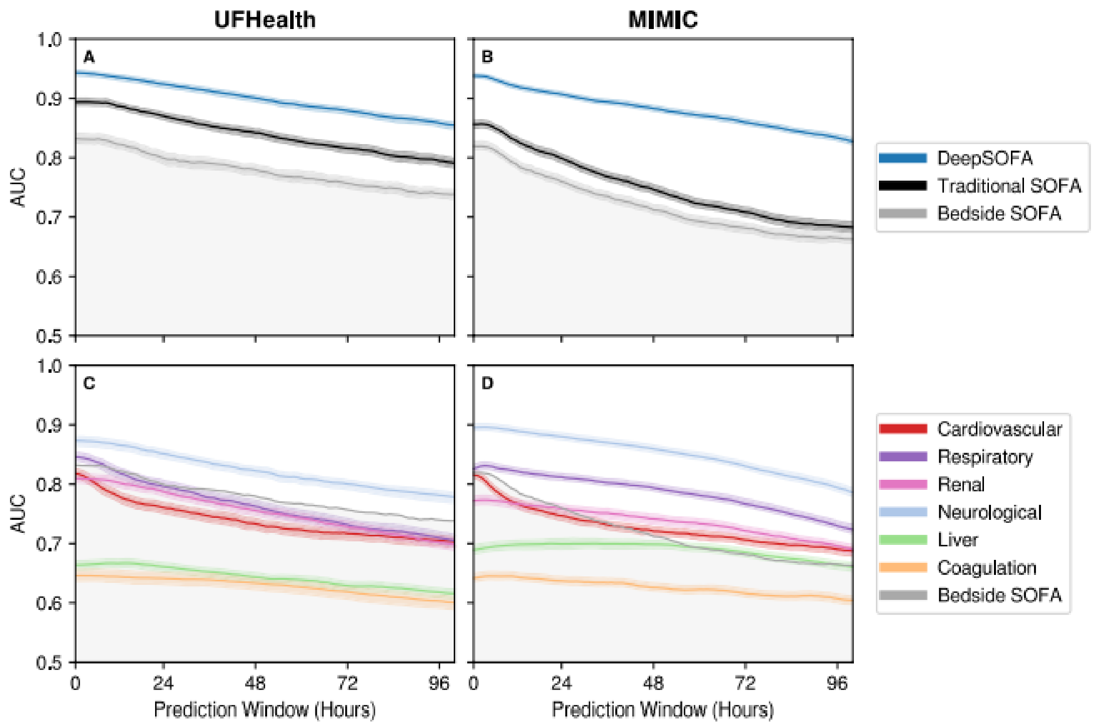


Figure 2.9: DeepSOFA results [37].

The superior accuracy of deep learning models can be attributed to their capacity for learning and associating multiple types of data. With that said, deep learning techniques may improve decision-making processes by analysing large volumes of data (a difficult and time-consuming task) and serving as an early warning system to notify healthcare providers about patients who need clinical interventions.

2.1.5 Machine Learning for Blood Pressure Prediction

The ML models used in the medical field varies from more conventional approaches, such as Logistic Regression (LR), to more sophisticated ones, like ANN. They are intended to give medical professionals a tool to assist in clinical decision-making [39]. In this section, a review is made of several machine learning techniques applied in the medical field to observe blood pressure values, as well as a brief description of these methods, in particular those that will be further compared in the next section.

Linear Regression defines a linear relationship between independent and dependent variables. It is a relatively simple algorithm that generates a linear equation that best fits a given problem. This algorithm relies on continuous output [24].

Support Vector Machine is an algorithm, capable of performing regression, linear and non-linear classification, and even outlier detection. It is one of the most widely used models because it can categorise complex but small to medium size datasets. SVM works by creating boundaries between distinct classes and when prompted with new instances categorises them as belonging to one of the classes [24].

Logistic Regression is commonly use to determine if an instance belongs to a given class. Its output ranges from 0 to 1 and it is designed to predict a binary outcome. It gives an estimated probability and, if it is above 50%, the model predicts that the instance belongs to a given class or else not [24].

Naive Bayes classifier is simple probabilistic classifier that uses Bayes' theorem for classification tasks. In the Naive Bayes approach, it is assumed that all features into consideration are independent from one another. The independence assumption is a naive assumption made in the classifier, but it works well in practice for many datasets. This helps to simplify the optimal classifier making it easy to write and run more efficiently than more complex Bayes algorithms [40].

Linear Discriminant Analysis is used for classification problems. It is a statistical technique, and it aims to find a linear combination of features that separate different classes as well as possible. It does this by maximising the separation between the classes while minimising the variance within each class. It is a linear method, which means that it tries to find a linear decision boundary that separates the different classes. The resulting function is linear with the form of

$$f(x) = w * x + b,$$

where w represents the weight vector, x the characteristic vector, and b is simple a threshold [41].

Random Forest Classifier is an ensemble machine learning algorithm that is used for classification tasks. It is called "random forest" because it is made up of a large number of decision trees, where each tree is "trained" or created using a random sample of the data. Each decision tree in the forest independently makes a prediction about the class of a given data point, and the final prediction of the random forest is the majority class predicted by the individual decision trees. The use of multiple trees helps to reduce overfitting and improve the overall performance of the model. In simple terms, a random forest classifier is a collection of decision trees that work together to make a prediction about the class of an input sample [42].

C4.5 decision tree is a specific type of decision tree algorithm used for classification tasks. It is an extension of the ID3 algorithm and is known for its ability to handle both continuous and categorical input variables. The main difference between a C4.5 decision tree and a random forest classifier is that a C4.5 decision tree is a single decision tree, while a random forest classifier is an ensemble of multiple decision trees. A C4.5 decision tree generates a tree by recursively splitting the data based on the feature that maximises the information gain at each split, while a random forest classifier generates multiple decision trees by using random subsets of the features at each split [42].

Multi Layer Perceptron is a type of feed forward ANN, which means that the information flows only in one direction, from the input layer to the output layer, without feedback loops. The Multi Layer Perceptron consists of multiple layers of neurons, including an input layer, one or more hidden layers, and an output layer [43]. Its way of operating is much alike the one described in Section 2.1.4.

Backpropagation Neural Networks represent a kind of ANNs that uses backpropagation techniques during its training process. During training, the input data is propagated through the network, and the output generated by the model is compared to the desired output. The difference between the two is calculated, and the weights of the neurons are adjusted using a Deepest-Descent algorithm to minimise the error. This process is repeated over many iterations until the model achieves the desired level of accuracy. This allows the model to learn and adjust its weights by propagating the error information backwards from the output layer to the input layer, which helps the model make better predictions over time [44].

Radial Basis Function Neural Network is a type of ANN that uses radial basis functions as activation functions and is usually divided into three layers: input, hidden, and output layers. Traditionally, its basis functions are Gaussian functions, and it aims to minimise the least-squares criterion resulting in better performance [45].

Convolutional Neural Networks are a deep learning technique inspired by the human's brain visual cortex, used in image recognition and processing. A CNN architecture consists of multiple layers, including convolutional layers, pooling layers, and fully connected layers. Each layer has its functionality and, by working together, they provide the final output [24].

K-Nearest Neighbour aims to classify new data entries into the most similar labelled entries. It assumes that similar things are near each other using distance metrics such as Euclidean or Manhattan distance to find the closest points. To make a prediction, the KNN algorithm calculates the distance between the new data entry and the training set data points. It then selects the k closest data points based on the calculated distance, and the prediction is made based on the class or value of those nearest neighbours. The variable k decides how many neighbours will be chosen for the KNN algorithm and has a significant impact on the model's performance.

Sequential Minimal Optimisation is an algorithm used to train Support Vector Machine (SVM)s. The algorithm is designed to efficiently solve the quadratic programming problems that arise during the training of SVMs, which involves finding the hyperplane that maximally separates two classes of data in a high-dimensional feature space. It works by breaking down the large quadratic optimisation problems into a series of smaller sub-problems that can be solved analytically. Sequential Minimal Optimisation works by iteratively selecting two Lagrange multipliers to optimise and then updating them analytically. This process continues until the algorithm converges to a solution that satisfies the Karush-Kuhn-Tucker conditions, which are necessary for an optimal solution to the SVM problem [46].

2.1.5.1 Analyses of Machine Learning Approaches

Moghadam et al. [47], developed a LR model to predict hypotension events. Hypotension events are events defined by abnormally low values of blood pressure. The authors used the Medical Information Mart for Intensive Care III (MIMIC-III) database [48] to randomly select 1000 records which had physiological signals of interest. Among those signals were values of arterial blood pressure, heart rate, systolic blood pressure, diastolic blood pressure, respiration rate and oxygen saturation, all collected from multiple sensors present in ICU. Throughout the study multiple ML classification techniques were trained, tested, and compared. The authors stated that, a LR algorithm and a SVM had similar performance to each other, and both outperformed the rest of the developed algorithms. However, because of the shorter training time of the LR compared to the SVM, the authors solely focused on the former and did specifying the accuracy of the SVM. It is stated that the LR can predict hypotension events within 30 minutes with an accuracy of 95%, a sensitivity of 85% and a specificity of 96%.

In a recent study, Cherifa et al. [49] developed an ensemble ML method that was capable of predicting hypotensive events in the ICU 10 min in advance. The authors used patient baseline data, physiological signals (heart rate, systolic and diastolic blood pressure, oxygen saturation and mean arterial pressure), as well as given treatment variables like the administration of vasopressors, during the 60 min prior observation window, to establish their feature set. The model was tested using the Multiparameter Intelligent Monitoring in Intensive Care II (MIMIC-II) database [50] and achieved an accuracy of 87%, a sensitivity of 70% and a specificity of 92%.

In another study from Moghadam et al. [51], the authors stated that invasive measurements of blood pressure signals are neither comfortable nor feasible for all patients, so the authors proposed a machine learning algorithm that can predict hypotensive events up to 30 minutes in advance using noninvasive physiological signals and compared it with an invasive method. The features used to train the model were comprised of several physiological signals of interest which included blood pressure, heart rate, breathing rate and oxygen saturation. Those features were calculated using the MIMIC-III database. The MIMIC-III database does not have a large enough dataset that contains non-invasive values so, to overcome this challenge, the authors simulated the non-invasive values of the different features by down-sampling their corresponding invasive measured signals. This also helped to make a better comparison by using the same dataset for the invasive model. The final results showed that even though the invasive algorithm achieved better performance, the non-invasive one was not far behind, achieving an accuracy of 93%, a sensitivity of 84%, and a specificity of 94%, against 95% accuracy, 85% sensitivity, and 97% specificity of the invasive method.

Additionally, more complex models of ML in the field of deep learning are used to perform hypertension classification tasks. For instance, Lee and Mark [52] trained an ANN to find patterns in ICU patients' physiological signals that could distinguish impending hypotensive events. The authors used minute-by-minute generated data from 6 different time series (heart rate, systolic blood pressure, diastolic blood pressure, mean arterial pressure, pulse pressure and cardiac output) to predict hypotension 1 hour beforehand. The trained algorithm was tested using the MIMIC-II database and achieved an accuracy of 85.8%, a sensitivity of 82.6% and a specificity of 85.9%.

Also, several studies discussed the classification and prediction of hypertensive events. Hypertension events, commonly known as high blood pressure, are the opposite of hypotension events. Hypertension events are characterised by abnormally high values of blood pressure. Patnaik et al. [53] compared the performance of multiple ML classification techniques. The classification techniques used in this study are Naive Bayes Classifier, SVM, LR, Random Forest and Multi Layer Perceptron. The data used to develop the different models was taken from the Korean National Health Insurance Corporation (NHIC) which contains EHRs of multiple patients who had their annual checkups between 2002 and 2013. The data considered to train the models were related to the population density, health checkup results, demographic details, and income-quantile details. The best accuracy obtained by the different models did not vary much however the SVM stands out with an accuracy of 80.23%.

In another paper, Nour and Polat [54] proposed a ML-based method for the classification of hypertension types based on personal data. They used the photoplethysmography-blood pressure (PPG-BP) dataset [55], which contains eight features that define the hypertension types: sex, age, height, weight, systolic blood pressure, diastolic blood pressure, heart rate, and body mass index. The models developed for the classification of hypertensive patients were a Random Forest Classifier, a C4.5 Decision Tree, a Linear Discriminant Analysis, and a linear SVM. The models achieved an accuracy of 99.5%, 99.5%, 96.3%, and 92.7%, respectively.

Additionally, more complex models of ML in the field of deep learning are used to perform hypertension classification tasks. For instance, Kwong, Wu, and Pang [56] generated two models to classify high blood pressure, using factors related to a person's age and lifestyle. The models raise alerts when the predicted systolic blood pressure is above the hypertension alarm value. The authors proposed two models: a Back-propagation Neural Network and a Radial Basis Function Neural Network. The dataset used was composed of 498 samples and included 9 different features (systolic blood pressure, age, body mass index, stress level, exercise level, alcohol consumption level, smoking status, and salt intake level). According to the findings, the best accuracy achieved by the Back-propagation Neural Network was 94.28%, with four hidden nodes, and by the Radial Basis Function Neural Network 91.06%, with five hidden nodes. Both models had one hidden layer. Other works have used more complex models, such as Convolutional Neural Network (CNN)s. For example, Luo et al. [57] implemented a CNN to do the same task. The study data were obtained from the MIMIC-II Waveform Database Matched Subset, which contains heart rate, systolic blood pressure, diastolic blood pressure, mean blood pressure, central venous pressure, pulse rate, respiration, and oxygen levels. Moreover, the authors present a comparison of their model to six other ML techniques: a KNN, a C4.5 Decision Tree, a Random Forest, a Sequential Minimal Optimisation, a Native Bayes, and a LR. The final accuracy reached by the CNN was 89.95%, having surpassed all the mentioned techniques, which reached an accuracy of 79.84%, 78.26%, 79.28%, 81.25%, 49.44%, and 76.58%, respectively.

Table 2.1 presents a summary of the discussed studies and their respective methods.

Table 2.1: Summary of machine learning algorithms to predict and classify blood pressure values.

Study	Approach	Accuracy
Moghadam et al. [47]	LR	95%
Cherifa et al. [49]	Ensembled ML Algorithm	87%
Moghadam et al. [51]	Non-invasive ML Algorithm	93%
Lee and Mark [52]	ANN	85.8%
Patnaik et al. [53]	Naive Bayes Classifier	77.71%
	SVM	80.23%
	LR	79.55%
	Random Forest Classifier	79.58%
	Multi-Layer Perceptron	73.85%
Nour and Polat [54]	Random Forest Classifier	99.5%
	C4.5 Decision Tree	99.5%
	Linear Discriminant Analyses	96.3%
	SVM	92.7%
Kwong, Wu, and Pang [56]	Back-propagation Neural Network	94.28%
	Radial Basis Function Neural Network	91.06%
Luo et al. [57]	CNN	89.95%
	KNN	79.84%
	C4.5 Decision Tree	78.26%
	Random Forest	79.28%
	Sequential Minimal Optimisation	81.25%
	Native Bayes	49.44%
	LR	76.58%

2.2 Big Data

The definition of Big Data has evolved rapidly, which has raised some confusion. It is a concept that became widespread in more recent years and can be defined as a set of extremely large, complex, and fast data which traditional data processing methods are inadequate and special tools are needed for its processing. This concept is often simply referred to as the use of different algorithms to extract as much information as possible from different data sets. However, the standard definition was made by Doug Laney in 2001 [58]. It has as his reference the three V's as a common framework to describe big data: velocity, volume, and variety. Nowadays, this definition has been updated and other V's are added to better describe this term. One problem, which has caused some controversy, is the number of V's used to describe Big Data. There are many websites and papers published that differ from each other, however, the most accepted V's, in addition to those already mentioned, are value and veracity [58].

1. **Volume:** Volume is connected to the very concept of Big Data. Without it, Big Data will never be that big. Nowadays, there are a lot of data to be stored and the amount of it, tends to grow exponentially, and due to the improvement of technology, it is possible to store these vast amounts of data in different locations (or even clouds) for more affordable prices, but it must be ensured that new technologies can cope with the growing size of data [58]. Figure 2.10 indicates the volume of data stored around the world in zettabytes (10^{21}), during the years.

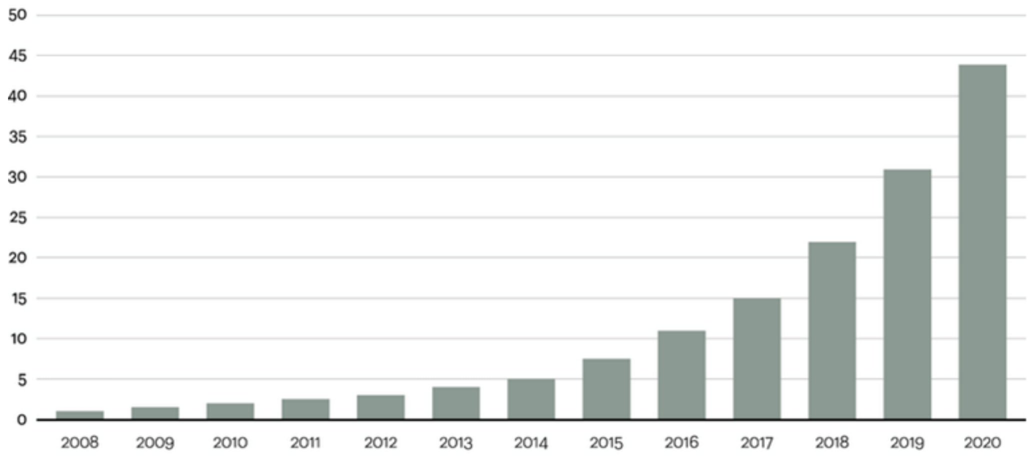


Figure 2.10: Data growth over the years [59].

2. **Velocity:** Big Data velocity concerns mainly two aspects: the increasing speed at which data is created and the speed at which data is analysed and stored.

   As seen before, data volume is exponentially growing, and this velocity contributes to bigger and bigger databases. It can even be said that data generates more data, and this only contributes to the velocity at which data is created [58]. According to an article published in Medium in September 2020 [60], "Facebook generates 4 petabytes of data per day – that's a million gigabytes". This shows how fast data volume is increasing.

   In addition, data acquired has to be processed in an adequate time. It has become more frequent for consumers and companies to demand data to be received and processed in real-time, with minimum latency [58].

3. **Variety:** It regards the wide variety of existing data and its corresponding sources. With the arrival of Big Data, data begins to arrive in many different formats, from structured to unstructured. Besides, there are a lot of varieties of structured, semi-structured, and unstructured data [58]. For example, in ICU, multiple sensors, such as temperature, oxygen, pressure, humidity, etc, collect different types of values or information. One of the challenges of Big Data is to be able to deal with this diversity.

4. **Veracity:** It refers to data quality. Because there are many different sources, it becomes difficult to know whether they are reliable or not. They can provide false information, and, in Big Data, this becomes a major problem as it is impossible to check these large quantities of information [58].

   It is also necessary to know that systems are imperfect and may produce false data. Every inaccurate data can lead to the wrong path/decision. This is important in a telemedicine environment because there is no mechanism for knowing whether the values obtained by the ICU sensors were correct or if a problem had occurred. Furthermore, the patient's data must be protected as well as his or her identity [58].

5. **Value:** Value is referent to the usefulness of the gathered data for specific tasks. Some data may be of use for some assignments and not for others. Big Data extracts values from data and reveals hidden truths, creating value. Data by itself, regardless of its volume, has no value [58].

2.2.1 Big Data Lifecycle

Arass, Tikito, and Souissi [61] provide a definition of the data lifecycle. It states that a data lifecycle is a set of steps (or phases) through which data passes from the point it enters a system until it leaves it.

In the same article, Arass, Tikito, and Souissi [61] analyse a series of different lifecycles and, through their analysis, they found that certain phases are mandatory whatever the field of application of Big Data, namely: Creation, Analysis, Storage, and Archiving.

Creation involves the creation and consequent capture of data generated from various sources. In this dissertation, this part could be associated with the data generated by the sensors presented in ICU. It is the followed by the Analysis phase. It is where data is analysed to provide accurate future observations. This phase also involves the selection of appropriate tools and methods for data processing. This phase is represented by the model and the application responsible for classifying the data and cleaning it. Next comes the Storage phase. As the name implies, is the storage of data in the system. In this step, information security is also present. Security becomes more relevant in this dissertation context, since patients information must remain anonymous. It is represented by the application responsible for storing the received data for future improvements of the model. The final phase is the Archiving one. It allows to archive data that is no longer needed [61].

2.2.2 Big Data Analytics

Gather meaningful information from massive amount of data has become increasingly important. With the existing variety of data in big data, extracting meaningful insights from different data sources is challenging. Big data analytics is responsible for processing and analysing the necessary data, discovering hidden patterns, and finding meaningful information. The tools available to handle all this data have significantly increased and today it is possible to examine data and obtain almost real-time responses. In general, these technologies are not expensive, and most software is open source [62].

Once data can be analysed, there are some methods to turn Big Data into valuable insights. ML is one example of a Big Data Analytics technique. It is a type of AI that enables software applications to predict outcomes more accurately without being expressly programmed to do so. ML methods were mentioned in more detail in Section 2.1 and its methods are at the base of some other Big Data Analytics techniques. Further Big Data Analytics techniques are:

- **Descriptive Analytics:** It helps to describe or summarise the historical data and provides insights into what has already happened. It involves mining large datasets to identify patterns and relationships, and then presenting the results in a way that is easy to understand [63].

- **Diagnostic Analytics:** This technique helps to diagnose the reasons behind an past event. It involves analysing the data to identify the factors that contributed to the event or problem [63].

- **Predictive Analytics:** This technique helps to predict future events or trends based on historical data. It understands the insights of data and identifies patterns that can be used to predict future outcomes [63].

- **Prescriptive Analytics:** This technique helps to prescribe or recommend the best course of action to achieve a desired outcome. It involves using the insights generated from descriptive, diagnostic, and predictive analytics to make recommendations and optimise decision-making processes [63].

In healthcare organisations, there is a wide variety of data, and Big Data technologies are essential to capture all available data and improve the quality of healthcare. Ristevski and Chen [64] stated that "applications of big data analytics can improve the patient-based service, to detect spreading diseases earlier, generate new insights into disease mechanisms, monitor the quality of the medical and healthcare institutions as well as provide better treatment methods".

Big Data Analytics techniques employed on clinical data available can do early detections of multiple diseases and identify the optimal practical guidelines for their treatment. Moreover, it can provide customised patient treatment by constantly monitoring patients' development. Analysing the patient data and data generated by other patients, who had similar symptoms, enables the doctors to make more informed decisions and provide better treatments. Furthermore, doing a live analysis helps predict the spreading of viral diseases [65].

Big Data Analytics software is either used for batch processing or real-time processing. Batch processing is an efficient way of processing high volumes of data, where the data collected is grouped or batched over a while for further processing. Apache Hadoop is one of the most used open-source tools for batch processing. However, when it comes to real-time processing, or stream processing, which involves a continual input, processing, and output, Apache Hadoop's performance degrades and that is where Apache Spark appears. Apache Spark is another open source tool mostly used for real-time processing even though it can do batch processing [66]. These technologies are analysed in the following sections.

2.2.2.1 Hadoop

Apache Hadoop [67] is one of the most used open-source tools. It is a framework that allows the distribution of large volumes of data across multiple clusters of computers. Each machine offers local storage and computational power. This method can do parallel processing and is designed to detect and handle failures, delivering a high-available service on top of clusters of computers, each one prone to errors. Hadoop is composed by four main modules:

- **Hadoop Common -** refers to a collection of common utilities that support the following modules.
- **Hadoop Distributed File System (HDFS) -** HDFS is Hadoop's storage unit. It is a distributed file system designed to run on commodity machines [1], splitting data into multiple blocks and saving it into various data nodes. HDFS makes copies and stores data across multiple computers, so if one data node crashes, that data is not lost, making HDFS highly fault-tolerant.
- **YARN -** Separates the resource management and job scheduling / monitoring functions into distinct daemons[2].
- **Map Reduce -** A software framework used for applications that process vast amounts of data in parallel on various commodity hardware clusters. Map Reduce usually divides the input data into small independent blocks, which are then processed in

[1]A term used to describe affordable devices.
[2]Programs that run as background processes.

parallel by map tasks. The outputs of the map tasks are then inputs of the reduced tasks, which will aggregate the results that will lead to the final output. Tasks are scheduled, monitored, and failed tasks are re-executed by the framework.

Hadoop is currently utilised by a wide variety of companies and organisations for research and production.

2.2.2.2 Apache Spark

Apache Spark [68] is a tool that allows combining Big Data analysis and AI over a large set of data. It provides high-level Application Programming Interfaces (API) and multiple components, each designed for a specific purpose.

The foundation of the platform is Spark Core. Spark Core includes components responsible for memory management, fault recovery, scheduling, monitoring and distributing tasks, and interacting with external systems. Spark Core is the basis of all Spark components.

One of its components is Spark Streaming. Spark Streaming is a Complex Event Processing (CEP). A CEP is a set of technologies used for querying continuous data inputs to obtain timely responses and detect opportunities and information before storing the data or, in some cases, discarding it. Spark Streaming is a solution that leverages Spark Core's fast scheduling capabilities to do real-time processing.

Spark Streaming has a Receiver that gets data streams coming from one or multiple sources. Those data streams are represented as Discretized Streams or simply DStreams, and they are the basic abstraction provided by Spark Streaming. Internally, a DStream is represented by a series of RDDs (Resilient Distributed Datasets). which contains batches of data from an interval of time. RDDs are a fault-tolerant collection of elements that can be processed in parallel. Any operation applied on the DStreams translates to operations on the RDDs. In the end, the processed data is ready to be outputted to other systems.

2.2.2.3 Apache Storm

One last technology used either for batch processing or real-time processing is Apache Storm [69]. Just like Spark Streaming, Apache Storm is also a CEP.

The Storm architecture is composed of two parts: spouts and bolts. Spouts are the parts where the data streams enter the processing unit, and they are connected to bolts and are responsible for passing the continuous streams to them. There can be multiple layers of bolts and a bolt possibly emits new streams and sends them to another bolt. Spouts and bolts are both programmed by users. Figure 2.11 shows an example of an Apache Storm topology. A topology is a computational graph that shows how information is transmitted.

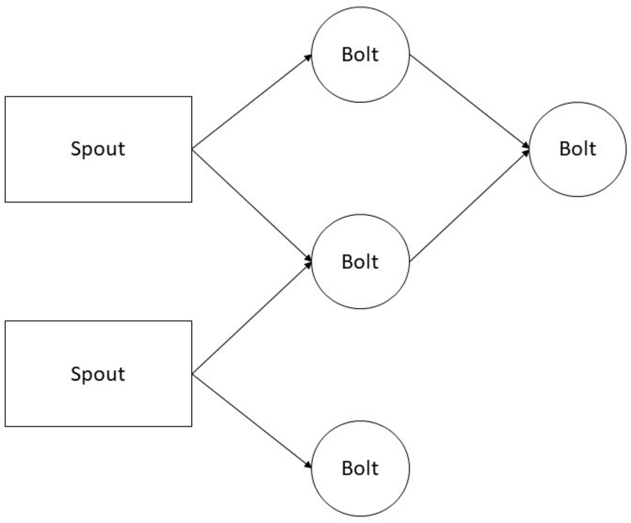


Figure 2.11: Apache Storm topology.

Each node runs in parallel, and the topology will continue its work until it is shut down.

2.2.3 Unstructured Data

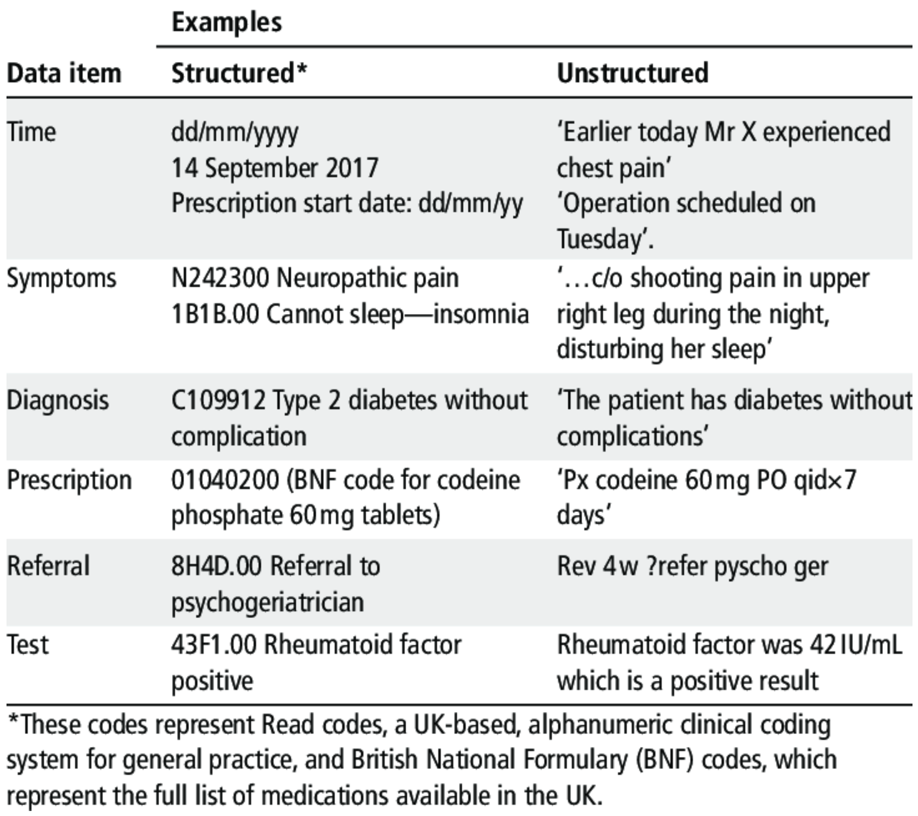
Unstructured data is information, stored in many different forms, which do not have an identifiable structure. Due to its lack of structure, unstructured data is difficult to process and analyse using traditional data analysis techniques. Also, unstructured data represents most of the generated data [70]. According to IBM [71], "projections indicate that unstructured data is over 80% of all enterprise data, while 95% of businesses prioritise unstructured data management". Examples of unstructured are the values collected by sensors or handwritten clinical information. Storing this types of data, requires different techniques such as Not Only SQL (NoSQL)[3] databases, which are used instead of traditional relational databases. NoSQL databases offer more flexible and scalable storage solutions, with the ability to handle a wide range of data types, including unstructured data.

2.2.4 Structured Data

Opposite of unstructured data, structured data is highly organised, follows a predefined schema, and is easily understandable. A good example of structured data is a relational

[3]Generic term used to represent non-relational databases.

database system, such as Structured Query Language (SQL) or a JavaScript Object Notation (JSON) file. Both provide enough details to be easily understandable. However, designing a schema is an elaborate process because it must be defined before inserting any content [70]. Figure 2.12 shows examples of structured and unstructured data in the medical field.



	Examples	
Data item	**Structured***	**Unstructured**
Time	dd/mm/yyyy 14 September 2017 Prescription start date: dd/mm/yy	'Earlier today Mr X experienced chest pain' 'Operation scheduled on Tuesday'.
Symptoms	N242300 Neuropathic pain 1B1B.00 Cannot sleep—insomnia	'...c/o shooting pain in upper right leg during the night, disturbing her sleep'
Diagnosis	C109912 Type 2 diabetes without complication	'The patient has diabetes without complications'
Prescription	01040200 (BNF code for codeine phosphate 60 mg tablets)	'Px codeine 60 mg PO qid×7 days'
Referral	8H4D.00 Referral to psychogeriatrician	Rev 4 w ?refer pyscho ger
Test	43F1.00 Rheumatoid factor positive	Rheumatoid factor was 42 IU/mL which is a positive result

*These codes represent Read codes, a UK-based, alphanumeric clinical coding system for general practice, and British National Formulary (BNF) codes, which represent the full list of medications available in the UK.

Figure 2.12: Example of structured and unstructured data in the medical field [72].

Unstructured data can account for up to 80% of data in a healthcare institution [73]. Unstructured clinical data contain valuable information, so it is necessary to be aware of its importance and existence. However, it is easier to interpret structured and standardised data. In a study made by Karim et al. [74], an application for structured reporting was created. This application was aimed at the radiology area, where most of the reports were done by hand, and many times two reports on the same image, by two radiologists, did not match. The objective of this application was to achieve a structured and standardised model and verify the impacts.

The built application aided the radiologist's workflow and data extraction, according to the findings. It is also mentioned that "structured standardised reporting has the potential to improve the patient care process and expand in other clinical realms within vascular imaging and other diseases".

An example of a standard structured model is Fast Healthcare Interoperability Resources (FHIR). FHIR is a standard model, developed by Health Level 7 (HL7), for healthcare data exchange. It allows healthcare clinical data to be exchanged and shared, with security, with those who need to access it. Moreover, FHIR can normalise and integrate unstructured data and then be shared among systems.

Since FHIR is a standard model in a telemedicine context, where there is cross-system communication, it also boosts interoperability.

2.2.4.1 FHIR

According to the Office of the National Coordinator for Health Information Technology [75], FHIR's development began in 2012. The purpose of FHIR was to combat the difficulties of interoperability faced in the medical area. One of the encountered problems was that healthcare facilities had their own database, which created conflict when sharing data with two different facilities. The growth in the availability of health data created the need for clinicians and patients to share data in an easy, reliable, and fast way.

FHIR is a set of "Resources" that describe clinical concepts in healthcare. The resources can be coded and handled separately, or they can be combined into sophisticated structures and documents that can be preserved and accessed when needed. FHIR resources are based on basic Extensible Markup Language (XML) or JSON representations, with an Hypertext Transfer Protocol (HTTP)-based RESTful protocol, and each resource is identified by a Uniform Resource Locators (URL) [76].

FHIR can be used for data analysis and AI by providing a consistent way to access and retrieve healthcare data from disparate sources. By using FHIR, healthcare data can be easily integrated into AI applications for analysis, prediction, and decision-making [77].

FHIR supports a wide range of data types and structures, making it ideal for capturing and exchanging complex healthcare data. It also includes features such as versioning and auditing, which can help ensure the accuracy and integrity of the data. Overall, FHIR provides a standardised, interoperable framework for accessing and analysing healthcare data, which can help facilitate the development of innovative AI applications in healthcare [77].

In the paper, "A FHIR-Enabled Streaming Sepsis Prediction System for ICUs", Henry et al. [78] present an application developed by Georgia Tech and Emory School to detect the risk of Sepsis using a ML framework. Sepsis is a life-threatening illness caused by a human's immune system's extreme response to an infection, which causes extreme inflammations and may damage oneself tissues and organs [79]. It is a well-known disease by the ICUs and the survival outcome is greatly influenced by the speed and suitability of the chosen treatment. The approach presented in the paper is "a cross-platform Sepsis Prediction and alert system which combines real-time data acquired from an FHIR server, a machine learning sepsis prediction tool and a front-end user interface intended for use by an ICU clinician" [78]. The system gathers information about a patient and, using ML algorithms, calculates a score that translates into a patient's risk of sepsis. The system demonstrated an accuracy of nearly 85% in detecting sepsis in a 4-hour gap, which was possible due to advancements in ML and the availability of real-time vital signs, both stated to be capable of improving healthcare systems. Additionally, systems that are compatible with FHIR standards, are also compatible with this system, which brings an improvement in interoperability.

2.2.5 Some Considerations on Medical Field Big Data

Some issues should be considered regarding Big Data. First off, there is a lot of heterogeneity of data sources (for example, the multiple sensors present in the ICU), which may lead to interoperability challenges. Multiple Big Data Analytics methods can be applied to better handle the heterogeneity of data.

Secondly, clinical data may be influenced by the staff who entered the data, which can lead to a variety of problems such as entering missing values or incorrect data. These problems may also lead to other problems for instance data quality, integration, and inconsistency.

Lastly, Big Data privacy and security are two important issues. All medical data is sensitive, and this data is legally considered to be possessed by the patient. So, it is necessary to ensure that when data is shared, patients' identities must be anonymous while also having good encryption algorithms [64].

2.3 Explainable AI and Clinical Decision Support

In the medical field, with the implementation of EHRs, the creation of datasets has been getting bigger, to the point that it can be referred to as "Big Data". Clinical "Big Data" can be used to revolutionise several aspects of healthcare, such as the ones mentioned in Section 2.2.2, and can also help the development of new decision support systems. The ICU is the one that can take the most advantage of Big Data. It is possible to monitor patients with life-threatening conditions while recording all the data available, producing large amounts of data, and making discoveries. In addition, it is possible to integrate data from previous similar cases, enabling clinicians to choose the appropriate approach for each individual [80].

With the rapid increase in data, there is a need to use new technologies to help clinicians to make better-informed decisions. CDSS are systems designed to help clinicians make these informed decisions for everyone.

CDSS help to decrease human error in the medical field. It is estimated that human error is responsible for the deaths of 200.000 people per year, and it also has a cost of 1.9billion$ annually [8]. If used properly, CDSS can change how medicine is delivered. According to Payne [81], there are several ways that computers can help clinicians, which are the following (quoting):

- Simplify Access to Data Needed to Make Decisions - Reporting results and creating standard and structured reports can help reveal important details and help in speeding decision making.
- Provide Reminders and Prompts - Reminders and prompts influence physician behaviour, resulting in better chronic, critical, and preventative medical care delivery.
- Assist in Order Entry - When preparing a prescription for a patient is necessary to take into account some factors such as allergies or previous prescriptions, and assisting clinicians to order medication can help reduce medication prescription errors.
- Assist in Diagnosis - There have been multiple programs that consider various factors to help determine the presence or absence of diseases.
- Review New Clinical Data; Alert When Important Patterns Are Recognised - When new events occur or new data is found or altered, sometimes bringing that information to the clinicians' attention is necessary.

Moreover, sometimes multiple resources need to be handled in an ICU environment and sometimes this handling proves to be a challenging task. All medical resources need to be dispatched at an adequate time when needed but with the increasing size and complexity of incidents, the difficulties of resource allocation also increase. Besides, in emergencies, medical information may be uncertain and incomplete.

Multi-criteria decision making (MCDM) methods can be a solution to these adversities. They provide tools that enable the management of the different resources whilst prioritising the needs and coordinating the resource distribution [82].

Multiple pieces of evidence show that CDSS helps clinicians carry for patients, and ML techniques are capable of improving these systems. However, ML techniques normally work as a black box, making it almost impossible to follow the logic of a system / machine while making important predictions in critical contexts. For this reason, the demand for transparency has been increasing. Explanations supporting an output are crucial, especially in situations where professionals need more information than just a simple binary prediction.

Improving the interpretability of a machine could also improve the human-machine relationship and trustworthiness. In addition, it is possible to correct the system deficiencies. This is where Explainable Artificial Intelligence (XAI) appears. XAI focuses on the challenge of demystifying the ML black box. It is a set of processes and methods that proposes developing a set of ML techniques to produce more explainable models while maintaining the performance level and allowing humans to comprehend, trust and manage systems [83].

One approach for AI transparency is explainability by design. Explainability by design involves developing models that are inherently interpretable so that they can be easily understood and trusted by users. This may involve using simpler, more transparent models [83], such as decision trees or linear regression, rather than complex models such as ANN.

Another XAI approach is post-hoc explainability. Post-hoc explainability techniques are methods used to explain the decision-making process of a ML model after it has made predictions or classifications. These techniques do not require any modification of the model during training or testing, but instead, analyse the model's outputs and behaviour to provide explanations for the decisions made. Post-hoc operations can either be model-agnostic or model-specific [83].

Model-agnostic techniques are methods that can be applied to any type of ML model, regardless of the algorithm or architecture used to train the model. These techniques are not specific to a particular model and can be used to analyse, interpret, or improve the performance of any ML model. In general, there are three model-agnostic methods: explanation by simplification, feature relevance estimation, and visualisation explanation [83].

Explanation by simplification refers to the process of generating an interpretable, simplified version of a complex black-box model that can help provide insights into how the model works and why it makes certain predictions. The goal of explanation by simplification is to create a simpler, more transparent model that approximates the behaviour of the original model, while also being easier for humans to understand. This simplified model can then be used to identify important features or variables that contribute to the model's predictions and to gain insights into how the model operates [83].

Feature relevance estimation is a process of determining the influence of different features or variables in a predictive model, without assuming any particular model structure or form. The goal of feature relevance estimation is to identify the features that have the greatest impact on the model's predictions and to understand how changes in these features affect the model's output. It is particularly useful in situations where the model is complex and difficult to interpret, such as in deep learning models. One fruitful contribution to this path is SHapley Additive exPlanations (SHAP) [84] [83].

SHAP is based on the concept of Shapley values from cooperative game theory, which quantifies the contribution of each player in a coalition. In the AI context, the players are the features and the "coalition" is the set of features used for the prediction. To compute the SHAP values for a given instance, SHAP explores combinations of features and calculates the contribution of each feature by comparing the predictions with and without the feature. As such, SHAP can estimate the impact of each feature towards a final output [85].

SHAP also provides a tool called the SHAP summary tool, which helps visualise the contribution of each feature to the final model output. The tool presents the variables on the left side of the plot, sorted in descending order based on their impact on the model's output. Features at the top had a greater impact than those at the bottom. The plot includes a horizontal axis that indicates whether a feature had a positive or negative impact on the model's output. Features on the right side of the axis had a positive impact, while those on the left side had a negative impact. Each feature is represented by a coloured ball on the plot, with the colour relative to the value of multiple features. Features with higher values appear redder, while those with lower values appear bluer. Overall, the SHAP summary plot provides a quick and intuitive way to interpret the importance of features in a ML model. In Section 4.6 an illustrative example of a chart generated by the SHAP summary tool can be found. This chart serves the purpose of enhancing the comprehension of the provided explanation.

Visualisation explanation, as the name implies, refers to methods of generating visualisations to help explain the behaviour of a predictive model. Visual explanations can help provide insights into how the model is making its predictions, which features are most important, and how changes in input data affect the model's output. These techniques are often used in combination with feature relevance techniques, which help identify the most important features in the model and provide the information that is eventually displayed to the end user. However, visual explanations are less common in the field of XAI as these methods must be able to work with any type of ML model. This can be a complex task, as it often requires the development of new algorithms or visualisation methods that are tailored to specific types of models [83].

On the other hand, model specific techniques are methods that are designed to work with a specific type of ML model or algorithm. These techniques are tailored to the characteristics and assumptions of a particular model and can provide more specific insights into its performance or behaviour [83].

Overall, the choice between model agnostic and model specific techniques depends on the goal of the analysis and the characteristics of the ML model. Model-agnostic techniques are more general and can provide insights into any model, while model specific techniques can provide more detailed and tailored insights into specific types of models. These techniques can help identify biases and errors in the model and increase trust in its predictions.

In the mid-1990s, an ANN was trained to determine which pneumonia patients should be admitted to hospitals and treated as outpatients[4]. Initial findings indicated that the network would improve accuracy however, it inferred that pneumonia patients with asthma had a lower risk of dying. Medically, this is counter-intuitive, yet, because patients with asthma were directly admitted to the ICU and had proper care, their chances of survival were in line with what was supposed. Since this factor was unknown, it was impossible to see why the network had come to such a conclusion, and the application of AI, for clinical purposes, was stopped. Only by interpreting the model would it be possible to discover such a problem and correct it [86].

[4] a person who goes to a hospital for treatment, but does not need to stay a night there

3 Proposed Solution

The proposed solution relies on the improvement of healthcare conditions in the ICU environment by validating an AI model in the context of the ICU4Covid European project. To contribute to the project, it is intended to validate a ML algorithm capable of providing feedback to ease and facilitate healthcare professionals' daily work. The proposed approach involves collecting data from sensors at the disposal of an ICU environment and using the trained model to make predictions that can assist clinicians in making more informed and prompt decisions. This would be a valuable asset as the ICU is a high-pressure environment where quick responses are often required, and practitioners could take action before something happens. Not only that, but also, patients' supervision could be done remotely, as the data collecting would be done by an intelligent system and practitioners would only need to take action if, based on sensors' data, irregular patterns start to appear.

As patient data is involved, trust and reliability are critical factors to consider when developing a system for the medical field. When dealing with human well-being and sensitive data, practitioners must have a clear understanding of the ML model's process and how it arrived at a purported solution. As machines are not infallible and several errors can occur, an XAI algorithm capable of providing insights into how a ML algorithm arrived at a solution is another essential aspect of development, enhancing human-machine interaction. The proposed solution comprises two main phases: training methodology and prediction approach (including its following explanation), which are explained in the following sections. To provide a more comprehensive context, a chapter on clinical relevance is also provided.

In conclusion, the proposed solution presents a significant opportunity to improve healthcare conditions in the ICU environment. However, it is crucial to develop a reliable and trustworthy system, given the sensitive nature of patient data and human well-being. The proposed approach's effectiveness will depend on its ability to predict, give interpretable insights and effective communication between healthcare professionals and the AI model. Consequently, the focus of this study is to give insights to healthcare professionals on MAP values. The idea is to develop an AI algorithm that, based on the data generated by the ICU sensors, can predict future MAP values, making it possible for healthcare professionals to actuate before something happens, allowing to dosage medication at a safe distance.

3.1 Clinical Relevance

Among several parameters for ICU patients, the measurement of blood pressure has been considered an appropriate way to obtain important clinical information. By measuring the systolic, diastolic, and mean arterial pressure it is possible to identify several problems that arise. Beyond that, many patients who enter the ICU are suffering from sepsis or septic shock. In many situations, a patient may suffer from one of those if the MAP is below 65 mm Hg [87].

Sepsis is a life-threatening illness caused by a human's immune system's extreme response to an infection, which causes extreme inflammations and may damage tissues and organs. Septic shock is defined as a subset of sepsis where severe cellular, metabolic, and circulatory anomalies are linked to a higher risk of death than sepsis alone. Clinically, septic shock patients can be distinguished by the need for vasopressors to keep their MAP at 65 mm Hg or higher [79].

One of the most commonly used vasopressors is noradrenaline (or norepinephrine) which aims to maintain the MAP values above the desired threshold (normally the mentioned 65 mm Hg) [88]. This chapter aims to propose a solution that can help manage the administration of noradrenaline through the analyses of bio-signals to predict unusual MAP values, allowing for remote medication management, using an AI algorithm.

3.2 Training Methodology

As observed in the Section 2.1.5, different ML models obtain distinct performances. Thus, it is believed that it is meaningful to train several models, using the same dataset (presented in Section 3.2.1), whose characteristics are distinguishable from one another. In this case, the proposed models are:

- LR.
- SVM.
- ANN.

These models are chosen for their different characteristics, and for the fact that they are suitable for different types of problems. LR and SVM are algorithms that aim to classify data entries, whereas the ANN can output continuous values.

After the models are trained, it is necessary to evaluate their performance using appropriate evaluation metrics such as accuracy, precision, recall (also called sensitivity or true positive rate) and f1-score. Accuracy represents the ratio of correctly predicted instances (both True Positives (TP) and True Negatives (TN)) to the total number of instances (TP, TN, False Positives (FP), and False Negatives (FN)). In other words, accuracy assesses the proportion of correct predictions made by the model out of all predictions. Precision is a metric that focuses on the correctness of positive predictions made by the model. It calculates the ratio of TP (correctly predicted positive instances) to the sum of TP and FP (incorrectly predicted positive instances). Precision in typically used along with the recall metric. Recall measures the model's ability to correctly identify positive instances from the entire population of actual positive instances. It calculates the ratio of TP to the sum of TP and FN (positive instances incorrectly classified as negatives). F1-score is a metric that combines precision and recall into a single value, providing a balanced measure of a model's performance. The F1-score provides an overall assessment of a model's effectiveness in both precision and recall, making it useful when a balanced performance between the two is desired [24]. For better understanding, the equation from the presented methods are the following:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \cdot 100 \tag{3.1}$$

$$Precision = \frac{TP}{TP+FP} \tag{3.2}$$

$$Recall = \frac{TP}{TP+FN} \tag{3.3}$$

$$F1-score = \frac{2}{\frac{1}{Precision}+\frac{1}{Recall}} = 2 \cdot \frac{Precision \cdot Recall}{Precision+Recall} = \frac{TP}{TP+\frac{FN+FP}{2}} \tag{3.4}$$

It is also important to use techniques such as cross-validation and grid search to tune the hyperparameters of the models, which can improve their performance. The model that obtains the best performance is then chosen for further testing in a real-world simulation [24].

For the sake of simplicity, it is proposed to start by training the LR to predict if the following measure of MAP will be below 65 mm Hg, specified in the above section (Section 3.1). Then it is proceeded to the SVM and concluded with the ANN.

Finally, it is important to note that the final model can be retrained over time with new data to improve its performance and adaptability. Keeping records of all the data processed and repeating the whole training process using the stored data, enables the model to be more efficient and know how to adapt to new situations. Additionally, the stored data can be added to the previously used dataset for the future development of a new model, improving the overall quality of the dataset.

Figure 3.1 shows the proposed training process.

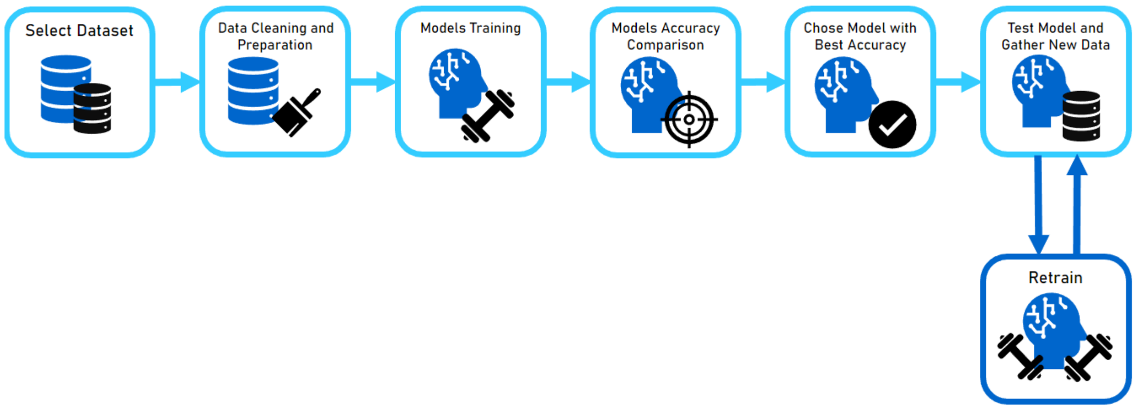

Select Dataset
Data Cleaning and Preparation
Models Training
Models Accuracy Comparison
Chose Model with Best Accuracy
Test Model and Gather New Data
Retrain

Figure 3.1: Training methodology.

The first step in this process is to obtain a large and representative dataset that is suitable for the task at hand.

3.2.1 Training Dataset

The development and comparison of multiple ML models using a single dataset is an important aspect in the field of ML, since training all the models with the same data, offers a better comparison method. The dataset must be comprehensive, diversified, and large enough to reflect the population under consideration. Internal data sources, public data repositories, and data given by third-party providers may all be included.

A rigorous search was conducted to locate datasets that contained the requisite variables and features necessary to model and address the identified problem accurately. Multiple potential datasets were identified, and each dataset's quality and relevance were evaluated. Specifically, the accuracy and completeness of the data were checked, and any missing or incomplete records were identified, in addition to any potential biases or errors that may affect the model's accuracy. The datasets identified as potential targets were:

- Cardiovascular Disease Dataset [89].
- Heart Attack Analysis & Prediction Dataset [90].
- Multi-parameter Intelligent Monitoring in Intensive Care (MIMIC) [91].
- MIMIC-II.
- MIMIC-III.
- PPG-BP.

MIMIC-II, MIMIC-III and PPG-BP were mentioned in Section 2.1.5.1, however, these are not publicly available datasets and so they were excluded from the potential datasets to use. The Cardiovascular Disease Dataset and the Heart Attack Analysis & Prediction Dataset did not have as much variables of interest as MIMIC, and were not as extensive. So, through much deliberation, the MIMIC Database was chosen as the most suitable dataset to train the ML model because it is a publicly available critical care dataset that contains de-identified health-related data of over 90 patients who were admitted to the ICU. The data includes the patient's vital signs, laboratory test results, medications, and fluid balance, as well as detailed information about the patients' ICU stays. The database contains nearly 200 days of real-time signals and accompanying data, each record typically consists of several hundred individual files. The recordings length varies; some are at least 20 hours, and others may exceed 40 hours. The database contains electrocardiograms, alarms related to changes in the patient's status, annotations related to changes in the functioning of the monitor, and values for arterial blood pressure, pulmonary arterial pressure, central venous pressure, and fingertip plethysmograph for records that have the corresponding signals.

The database information for use is comprised of six signals, these being the respective values of blood pressure (systolic, diastolic, and MAP), heart rate, breathing rate (or respiration rate) and oxygen saturation. All these signals were obtained from bedside monitors, and it is important to note that the dataset was pre-labelled.

To have a more robust model, information related to the person's age, gender and condition have also been considered. Notice that when dealing with medical data an important thing to take into consideration is the privacy of the patients. Their personal data, such as name, identification, etc, should not be available for public disclosure and are not present in this dataset.

Table 3.1 showcases a subset of the dataset used in the analysis. The table consists of three rows, with each row, representing a sensor reading. The readings are listed in consecutive order, from top to bottom. The objective is to develop three separate models that accurately predict future MAP values based only on the preceding sensor reading.

To develop a LR, a SVM, and an ANN, the dataset includes two distinct target columns: a binary target and a target column. Each model will utilise the appropriate column based on its specific characteristics.

For the LR and the SVM models, the binary target column will be used to classify whether the next reading will be below 65 mm Hg or not. The binary target column contains a value of 1 if the subsequent MAP reading is greater than 65 mm Hg, and 0 if it is not. These models will aim to predict the binary outcome.

On the other hand, the ANN model will utilise the target column to predict the actual numerical value of the next MAP reading. The target column directly represents the numerical value of the subsequent reading.

To summarise, the binary target column is used for classification by the LR and SVM models, while the target column is used the prediction by the ANN model. Each model solely utilises one of these target columns during its development process.

Table 3.1: Dataset sample.

Reference	Age	Sex	Condition	MAP	SBP	DBP	HR	PR	BR	O2	Target	Binary Target
230	75	1	CHF	65	104	46	72	72	15	97	65	0
230	75	1	CHF	65	105	46	72	72	15	97	66	1
230	75	1	CHF	66	105	46	72	72	15	97	66	1

MAP: Mean Arterial Pressure.
SBP: Systolic Blood Pressure.
DBP: Diastolic Blood Pressure.
HR: Heart Rate.
PR: Pulse Rate.
BR: Breathing Rate.
O2: Oxygen Saturation.

3.3 Prediction Approach

The model is trained with a dataset of variables, and it can be used to make predictions by feeding it with the same variables used during the training process. The sensors are connected to a program that loads the stored trained model and listens to incoming sensor information. The program keeps doing constant predictions, compares the predicted values with a defined threshold and raises an alarm if multiple predicted values are lower than the defined one so that healthcare professionals can act to prevent the MAP from dropping to values below the set value or authorise the administration of noradrenaline. Whenever sensors read values, the model makes a blood pressure prediction.

It is important to note that regular monitoring and maintenance of the system are necessary. The sensors and the program need to be calibrated and checked regularly to ensure that they are working correctly and that the predictions are accurate. For instance, the first time the program is launched, certain information must be provided. As stated before, the normal MAP threshold is 65 mm Hg still, this value does not apply to all patients. Depending on the patient's medical condition or some health problems (e.g., hypertension, diabetes, etc.), the MAP threshold can vary to higher or lower values, and the doctor is responsible to provide the best-suited threshold value for each patient. Additionally, a plan for handling errors or malfunctions should be in place.

Furthermore, since the sensors are constantly reading values, it is necessary to have a receiver that can get data streams coming from one or multiple sources. Based on the technologies analysed in the Section 2.2.2 it is proposed on using Spark Streaming since it is stated to be an easy-to-implement technology and provides tools to store data and discard it if needed. It comes in handy since one of the objectives is to save the processed data for future improvements of the model, as indicated in Section 3.2.

It is also important to have a robust cybersecurity plan in place to protect sensitive patient data from being accessed by unauthorised parties and ensure compliance with relevant regulations.

3.3.1 Prediction Architecture

In computer science, an architecture is the entire design and structure of a system, application, or program. An architecture defines the fundamental components, relationships, and principles that control the behaviour and functionality of the system.

Defining an architecture entails identifying the problem or goal that the system must solve or achieve. It comprises establishing the system's requirements, restrictions, and desired outcomes. Then, the components that will compose the architecture must be designed as well as the way they will interact with each other, all of this while considering multiple factors that will affect the overall system such as performance, scalability, maintainability, and security.

Once the architecture has been specified, it may be developed, tested, and modified iteratively as needed. The architecture serves as a blueprint for the system, helping to ensure that the final product meets the desired objectives and requirements.

For easier understanding Figure 3.2 represents the architecture of the proposed solution.

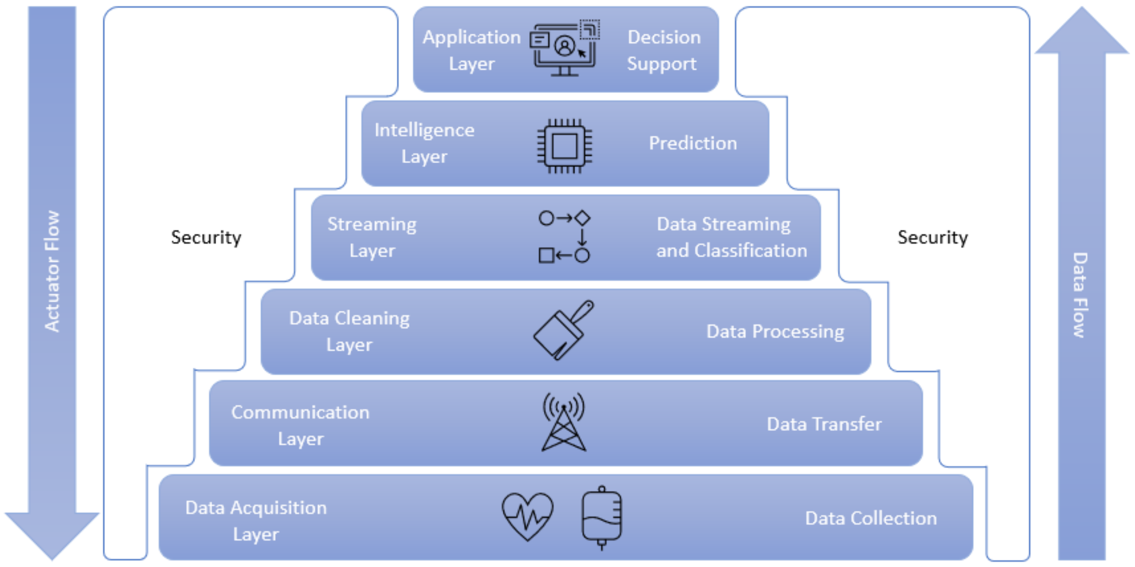


Figure 3.2: Prediction high level architecture.

It can be categorised into six different layers: data acquisition layer, communication layer, data cleaning layer, streaming layer, intelligence layer, and application layer, described as follows:

- **Data Acquisition Layer** is responsible for gathering data from various sensors and other data sources. It includes hardware components such as sensors, data loggers, and other devices that are used to collect data, as well as the software needed to manage and process the data. In this study, it would be composed of the multiple sensors present in the ICU.
- **Communication Layer** is responsible for transmitting data between different components of the system. It handles the communication between the physical devices and the component responsible for processing the information received (the processing unit, which, in this study, will be the B-Health IoT Box). It may include network hardware such as routers, switches, and hubs, as well as the protocols and software needed to ensure reliable, secure and efficient communication between different devices. In this study, it is mandatory to be in constant communication between the multiple sensors of the ICU and the processing unit.
- **Data Cleaning Layer** is responsible for performing various data cleaning processes to ensure the quality, the consistency, and the reliability of the data being used by the system. By eliminating excess or irrelevant data, the layer reduces the data size that needs to be transmitted, thereby improving the efficiency and speed of the overall process. This optimisation enables faster and more streamlined data transfer from the communication layer to the streaming layer. In this study, it collaborates closely with the communication layer. The Data Cleaning Layer ensures that the transmitted data is appropriately cleansed, standardised, and aligned with the expected format, facilitating seamless integration with subsequent layers.
- **Streaming Layer** is responsible for processing and analysing data streams in real-time as they are generated, rather than processing stored data. It allows continuous processing of data and enables immediate feedback, making it particularly useful for applications that require real-time analytics or quick responses. It is required in this system since data is continuously being generated and it is useful to be classified and processed before a prediction is given.
- **Intelligence Layer** is responsible for analysing data and generating insights using machine learning, artificial intelligence, or other advanced analytics techniques. It is where the future chosen model is loaded and will play its part.
- **Application Layer** includes the user-facing components of the system, such as the feedback provided to medical staff. It is where most user interaction happens, and that is why it requires it to be simple and easy to understand. In this study, this layer would be a simple user interface where the ML model reasoning would be

explained, and the initial settings would be entered, or the communication with the actuator part of the system enabled.

3.3.2 Workflow for Blood Pressure Prediction

The architecture previously presented represents the data flow through the system. This section aims to show how each layer enters the workflow for the prediction of blood pressure values. The process initiates with the collection of data from diverse sensors in the ICU. These data are then subjected to a cleaning process and forwarded to the system accountable for their classification. The purpose of the cleaning process is to remove unnecessary data generated by the ICU sensors, making the transmission process smoother and start preparing the data for prediction.

Upon receiving the data, the system classifies the input and initiates the prediction process. If the resulting prediction exceeds the multiple times the predefined threshold value, the system triggers an alarm to notify the medical personnel. In contrast, if the prediction does not breach the limit, the system reverts to its original state of data collection. Concurrently, upon making a prediction, the system also presents a model explanation to exhibit the impact of each variable on the resultant output. The diagram in Figure 3.3, represents the system's conduct.

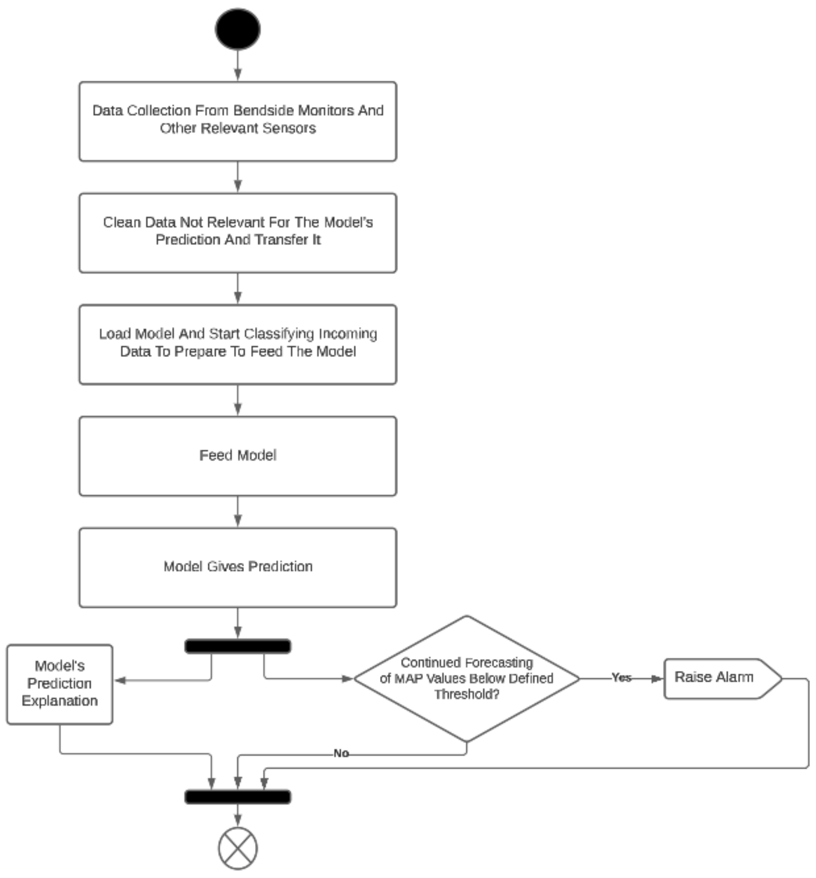

Data Collection From Bendside Monitors And Other Relevant Sensors
Clean Data Not Relevant For The Model's Prediction And Transfer It
Load Model And Start Classifying Incoming Data To Prepare To Feed The Model
Feed Model
Model Gives Prediction
Model's Prediction Explanation
Continued Forecasting of MAP Values Below Defined Threshold?
Yes
Raise Alarm
No

Figure 3.3: Blood pressure prediction workflow.

3.4 Integration with ICU4Covid

As mentioned before, this study is part of the ICU4Covid project. It is pretended to validate the trained model through its integration with the ICU4Covid's Cyber-Physical System for Telemedicine and Intensive Care (CPS4TIC).

CPS4TIC provides means to establish continuous operations between the ICU Hubs in central and remote hospitals to share information and predictions in real-time, reducing the risk of infection for the staff. The system is composed of several components which provide means for telemedicine and remote monitoring and enable prediction models to be stored, adapted, and shared. The system enables the collection of data from patients through its real-time connection to ICU devices and existing Patient Data Management Systems (PDMS), providing means for the development of prediction models for diagnosis, prognostics and treatment schemes based on evidence, data, and the experts' decisions. CPS4TIC focuses on the real-time monitoring of patients, telemedicine consultation, and the development and rapid sharing of models between ICU Hubs [92].

The ongoing study would be deployed on CPS4TIC, and its function would be to receive the data collected in the ICUs and send feedback to the intended destination, becoming an additional resource since the model contributes to the improvement of telemedicine while maintaining the same standards of quality care.

Moreover, the ICU4Covid project uses Federated Learning to develop ML models. Federated Learning is a ML technique that enables multiple devices or systems to train a shared model while keeping their data locally on the device, rather than centralising the data in a single location. This allows for training models on a large, decentralised dataset, while also addressing privacy concerns [93].

The basic idea behind Federated Learning is that each device or system trains a local copy of the model using its data, and then sends updates to a central server. The central server then aggregates these updates to create a shared global model. This process is repeated several times, with each device or system updating its local model using the shared global model, and sending updates back to the central server. It is particularly useful in situations where data is distributed across multiple devices or systems, and it is not practical or possible to centralise it [93].

Federated Learning enables organisations or devices to train ML models without compromising the privacy and security of their data. These features are very useful in the ICU4Covid framework since medical organisations have to work under strict patient data privacy policies, and data is distributed across multiple hospitals and cannot be centralised easily [93]. Figure 3.4 is a visual representation of what federated learning is.

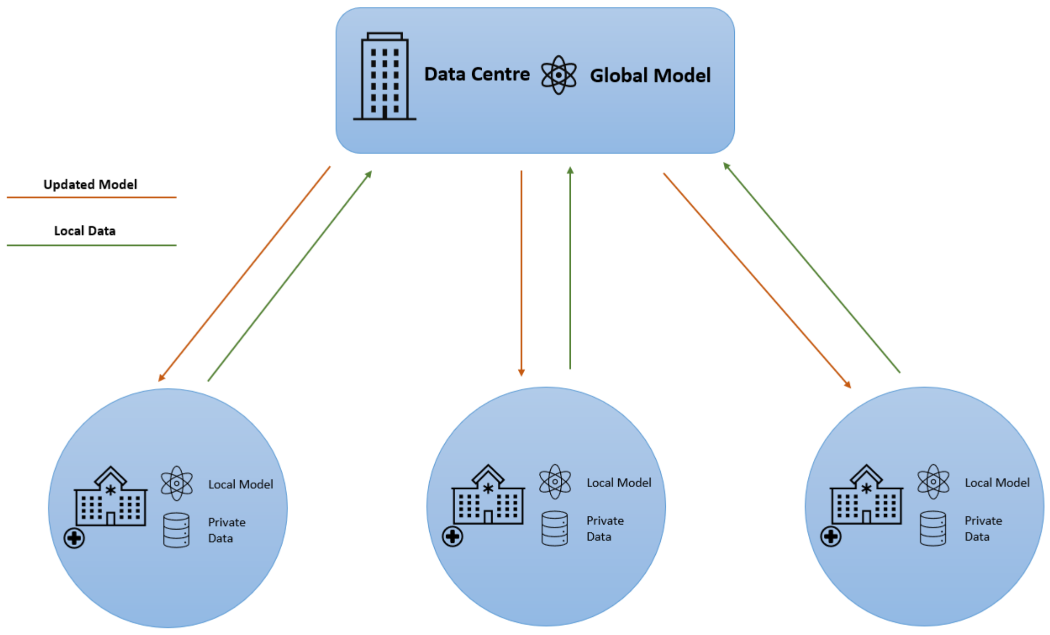


Figure 3.4: Federated Learning.

4 Implementation

This chapter outlines the implementation procedures undertaken for the development of a predictive model for MAP in the ICU4Covid project. It commences with an examination of the training process of multiple models, with a subsequent analysis of their outcomes. The chapter then delves into the various steps taken to rigorously test the model using a simulator and provides detailed explanations of all associated settings. Finally, the integration and validation of the model in the ICU4Covid project are presented, including a discussion of encountered challenges throughout the implementation process.

Figure 4.1 depicts the architecture presented in Section 3.3.1 and associates the components of the final solution to the different layers. It also shows how data flows through the various components and the tools used.

The data acquisition layer is composed solely of the Philips IntelliVue Information Center iX (PIIC iX) component (more details about PIIC iX are presented in Section 4.2), which generates the initial data. The data is then transmitted to the communication layer, which is composed by Mirth [94]. From there, the data is further directed to the data cleaning layer, where using Mirth's framework, various data cleaning and filtering operations are executed, ensuring the quality and integrity of the data. Once the data has undergone the necessary cleaning processes, it returns to the communication layer. The communication layer facilitates the seamless flow of the cleaned data. It leverages the outputs from the data cleaning layer to enhance the data transmission process, ensuring the delivery of high-quality and refined data. Section 4.3 presents these operations in more detail. The communication layer subsequently transmits the cleansed data to the B-Health IoT Box [95], which houses the remaining layers of the system.

The B-Health IoT Box, an integral component of the CPS4TIC, is strategically positioned near the patient's bed in the ICU. The B-Health IoT Box collects and transmits vital patient data to a centralised observation centre. It is also a versatile and customisable device that allows for the installation of various applications. This enables real-time monitoring and analysis of patients' health status, facilitating timely interventions and informed decision-making by medical professionals. The validation of the developed work takes place within this context, and Section 3.4 provides a comprehensive exploration of the B-Health IoT Box and the validation process carried out in the dissertation.

The streaming layer, which is part of Spark Streaming (and the B-Health IoT Box), then loads the model and awaits incoming data to classify it (Section 4.4). The resulting data is passed on to the intelligence layer, also part of Spark Streaming, which uses the trained ANN to provide a prediction. Section 4.5 delves into more details about the intelligence layer and its results.

Upon completion of these operations, the classified data, along with the prediction, is stored by the application layer. The application layer also displays an explanation of the prediction to the user and may raise an alarm depending on the value predicted and the frequency of values equal to or below it. The explanation is analysed in Section 4.6, while the alert and the data storage are explained in Section 4.5.

It should be noted that the streaming layer and the intelligence layer are both components of Spark Streaming, which is also a part of the B-Health IoT Box. In addition, the application layer can be added to the B-Health IoT Box.

Overall, the system consists of various layers and components that work in tandem to ensure the accurate and efficient prediction of medical data, along with appropriate explanations and alerts for healthcare professionals.

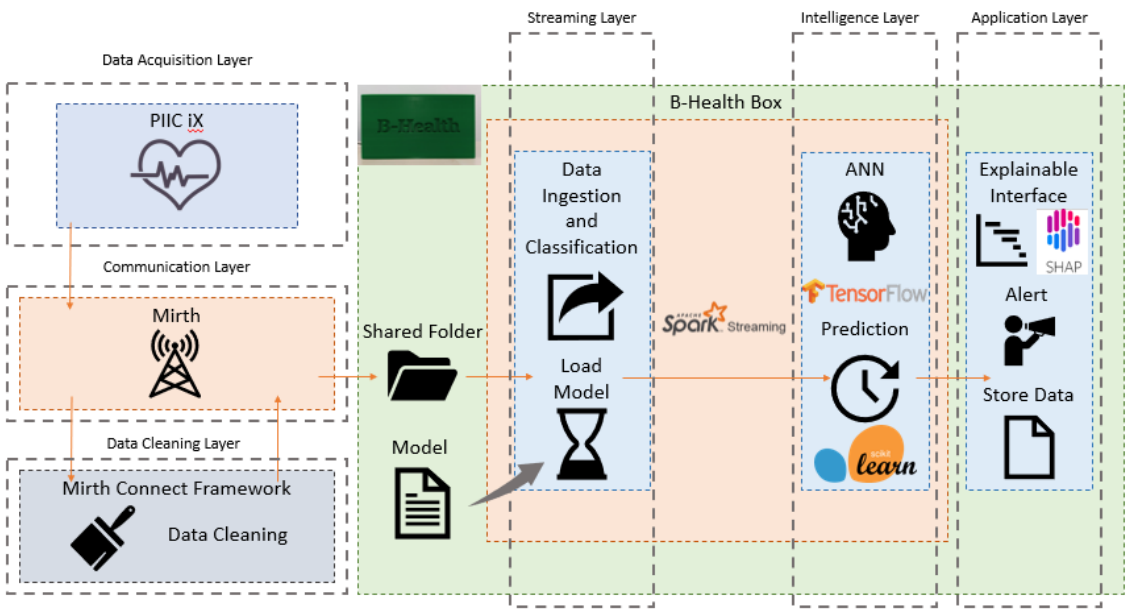


Figure 4.1: Architecture.

4.1 Models Training and Testing

Before starting the models' training, it was necessary to convert the data from the MIMIC dataset into the .csv format as it facilitates the training process since it is presented in a structured format. As the data from this dataset came in multiple .txt files, a small

Python script was developed to aggregate these files into a single .csv file. However, as a consequence of the substantial size of the dataset, certain practical limitations arose when attempting to handle it as a single .csv file. Consequently, it became imperative to partition the dataset into two .csv files to circumvent these constraints. The division of the dataset into two separate files did not present any issues, as it aligns with the standard practice in model training that necessitates both training data and testing / validation data. The first file, encompassing 85% of the dataset, was exclusively employed for the training process. Conversely, the second file, constituting 15% of the dataset, was dedicated to testing purposes. This division allowed for the assessment of model performance with new data inputs.

The training consisted of the development of three different models as stated in Section 3.2. To evaluate the performance of the models trained, an evaluation metric was chosen. Within the existing evaluation metrics, accuracy was the one used. The mathematical expression of the accuracy metric in ML is typically represented as the proportion of correct predictions made by the model out of all predictions made, often represented as (number of correct predictions) / (total number of predictions). It is usually denoted by the symbol "Accuracy" and is often represented in percentage form (multiplied by 100). The equation is represented by equation 3.1:

$$Accuracy = \frac{Correct\ Predictions}{Total\ Number\ of\ Predictions} \cdot 100 \tag{4.1}$$

Since accuracy is the ratio of correctly predicted instances, it can also be represented by the quotient between the sum of TP and TN, divided by the sum of TP, TN, FP, and FN, as shown in Section 3.2.

Accuracy is calculated during the testing process of a ML model after the model has been trained. The purpose of the testing process is to evaluate the performance of the trained model on previously unseen data. During the testing process, the model makes predictions for the target variables of the test data, and the accuracy metric is calculated by comparing these predictions to the actual target values. This gives an estimate of how well the model has learned from the training data and how well it generalises to new data.

To improve this step, it was used the referred second dataset file to put each model to the test which will provide a new value of accuracy. Figure 4.2 gives a intuitive explanation of the process.

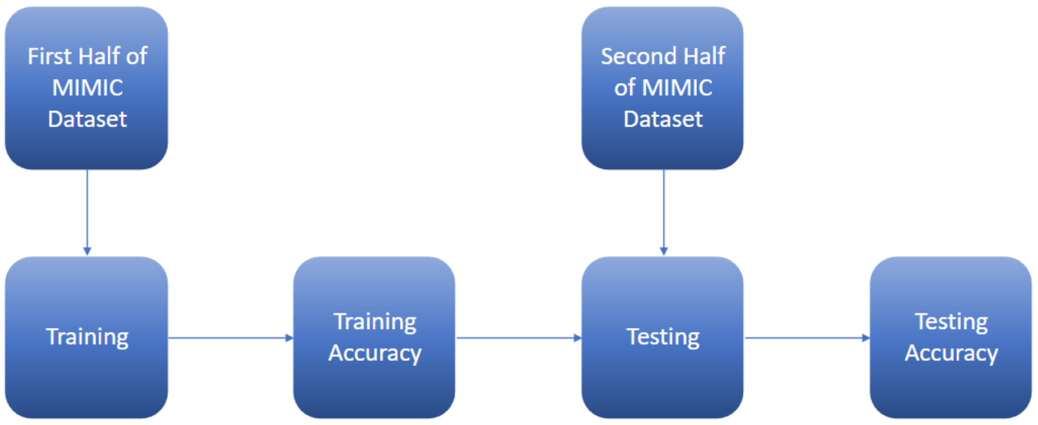


Figure 4.2: Training and testing accuracy.

The technologies to be used in the training process are TensorFlow, Keras, and scikit-learn. TensorFlow and Keras are more directed to the development of deep learning models, while scikit-learn relates more to statistical models and machine learning models outside the deep learning area. Hence, TensorFlow and Keras will be used for ANN training and scikit-learn for LR and SVM.

4.1.1 Logistic Regression

The LR model was implemented as the first model. In preparation for its implementation, the “Target” column was omitted from the dataset, as LR models produce binary outputs of either “true” or “false”. To facilitate the development of this model, the scikit-learn library was utilised.

The LR model training process was promply implemented. The accuracy obtained after the training phase was 98.420%, however, this value dropped to 90.139% when presented to the second dataset file. Even so, the results achieved were promising.

4.1.2 Support Vector Machine

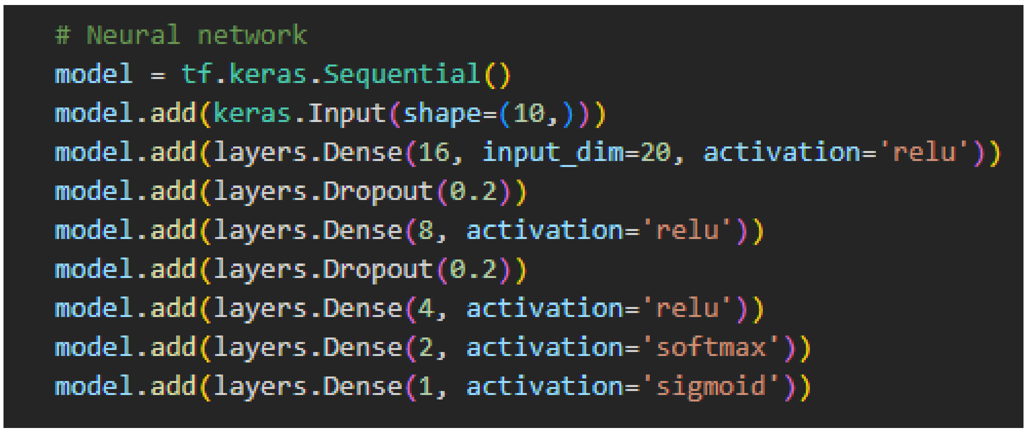
The SVM was the second model to be put into action. Similar to the LR, the "Target"column was dropped from the dataset as both produce identical outputs, and the scikit-learn library was utilised to aid in the implementation of the SVM.

Contrary to the LR model, the SVM training was a time-consuming process, and the accuracy obtained was below the previously obtained. This value was 92.428% which then had a significant drop to 67.576% when tested with the second dataset file.

4.1.3 Artificial Neural Network

Finally, the ANN was trained. The training preparation for the ANN was more complex. Firstly, it is necessary to define the ANN structure, and secondly, its accuracy will vary depending on multiple hyperparameters, specifically the batch size and the number of epochs.

Regarding the ANN structure, the ANN is implemented using TensorFlow's Keras library. The network is composed of eight layers and constructed as a sequential model, meaning that the layers are added one after another. The first layer, which serves as the input layer, has 10 units. The next layer is a Dense layer with 16 units and a Rectified Linear Unit (ReLU) activation function. A Dense layer is a type of layer where every neuron in the previous layer is connected to every neuron in this layer. Since this one is composed of many neurons and would have many connections if the next layer was another Dense layer, it was decided to insert a Dropout layer with a rate of 0.2 between them. A Dropout layer is used to randomly turn off a portion of the connections to prevent overfitting. The next layer is another Dense layer with 8 neurons and a ReLU activation function and, for the same reasons as before, it is followed by another Dropout layer. This is followed by two more Dense layers, with 4 and 2 units, respectively and ReLU and sigmoid activation functions. The final layer is a Dense layer with 1 unit and a sigmoid activation function, which provides the output of the network. Figure 4.3 provides better visualisation of the neural network.



```
# Neural network
model = tf.keras.Sequential()
model.add(keras.Input(shape=(10,)))
model.add(layers.Dense(16, input_dim=20, activation='relu'))
model.add(layers.Dropout(0.2))
model.add(layers.Dense(8, activation='relu'))
model.add(layers.Dropout(0.2))
model.add(layers.Dense(4, activation='relu'))
model.add(layers.Dense(2, activation='softmax'))
model.add(layers.Dense(1, activation='sigmoid'))
```

Figure 4.3: Trained Neural Network.

Concerning the ANN's accuracy variability, the best values for batch size and the number of epochs were found by training the network multiple times while varying these hyperparameters. This is what is called a grid search. A grid search is a technique used for tuning hyperparameters of a ML model. A grid search algorithm will train the model with different hyperparameters combinations and evaluate their performance using cross-validation. Grid search will verify the best combination of hyperparameters that result in the best performance. To implement this technique the network was trained nine times. The batch size number was varying between 16, 32 and 64, and the epochs number between 25, 50 and 100. The network was trained one time for each possible combination and its accuracy results were saved and can be seen in 4.1 as well as the network loss. A loss value in a neural network refers to the error or discrepancy between the actual output values of the network and the expected output values. It measures how far the network's predictions are from the target values and is used as a performance metric for training the network. The goal of training a neural network is to minimise the loss value. A lower loss value indicates that the network is making better predictions and is therefore better trained [24]. In this case, the mean squared error was used to calculate the loss value, and its expression is as follows:

$$Mean\ Squared\ Error = \frac{1}{n} \cdot \sum_{i=1}^{n} (y_i - \bar{y_i})^2 \tag{4.2}$$

Where:

- n is the number of samples in the dataset
- y represents the actual (true) values
- $\bar{y}$ represents the predicted values

While doing the grid search technique, it was verified that the training time of the network is dependent on the number of epochs and the batch size. As can be seen in table 4.1, the loss and the accuracy values were very similar in every case, so it was decided to use the network whose training time was the lowest. In this case, it was the one with 25 epochs and a batch size of 64.

Table 4.1: Artificial Neural Network accuracy with different parameters.

Batch Size	Epochs	Final Loss	Accuracy
16	25	0.5190	88.898%
16	50	0.5189	88.867%
16	100	0.5188	88.887%
32	25	0.5188	88.882%
32	50	0.5188	88.885%
32	100	0.5188	88.883%
64	25	0.5187	88.885%
64	50	0.5187	88.880%
64	100	0.5188	88.841%

Once the network to use was chosen, it was tested with the second part of the dataset. Similarly, to the previous models, its accuracy dropped to 75.063%.

4.1.4 Models Comparison

Only one model will be used for the actual prediction, so this section presents a comparison between the three tested models.

Among the three models trained, the LR was the one that got the highest accuracy in both phases. Additionally, it required less time for training compared to the SVM and the ANN. While SVM exhibited higher accuracy during the training, its performance dropped to values below the ANN's accuracy when fed with new data. Besides, the SVM takes more time to train than any other model trained and was excluded from the models to use in the following steps. The final selection came down to the LR model and the ANN. Despite the LR yielding higher results and being faster to train than the ANN, the ANN provided continuous outputs instead of binary outputs. The fact that it provides continuous outputs was a decisive factor since it allows doctors to precisely know the expected reading value and MAP variations. In contrast, the LR binary classification provided limited information, with the value 0 indicating that a subsequent reading will be some value below 65 mm Hg and the value 1 some value above 65 mm Hg.

Considering these factors, the ANN model was chosen to proceed with the simulation due to its ability to provide continuous predictions, offering more detailed insights for medical professionals.

4.2 presents in a tabular form, the comparisons made in this section.

Table 4.2: Models' comparison.

Model	Training Accuracy	Final Accuracy	Time of Training	Output
LR	98.420%	90.139%	Fast	Discrete
SVM	92.428%	67.576%	Long	Discrete
ANN	**88.841%**	**75.063%**	**Medium**	**Continuous**

4.2 Data Collection

As mentioned in the architecture diagram in Section 3.3.1, Data Acquisition Layer is responsible for collecting the data generated by the multiple sensors present in an ICU environment. This data is then transmitted to upstream layers for processing and analysis.

To implement this layer, a partnership with Philips Portugal [96] was made. From this partnership, a simulator of ICU bedside systems was provided by Philips. Its objective is to generate real-time medical data which will be then sent to be processed in the following processes presented in the following sections. The simulator received is named PIIC iX Demo and is a patient monitoring simulator that allows healthcare professionals to practice and develop their clinical skills in a safe and controlled environment. It simulates the vital signs of a patient, including heart rate, blood pressure, and oxygen saturation, as well as alarms and other features found in a typical patient monitoring system.

The PIIC iX is designed to provide an accurate simulation of a patient's vital signs and can be used to train healthcare professionals in the use of patient monitoring equipment, as well as in the recognition and management of critical situations. The system can be configured to simulate a wide range of patient conditions, including normal vital signs, critical conditions such as sepsis and cardiac arrest, and various alarm scenarios.

It is widely used in healthcare training and education programs, to help prepare healthcare professionals for real-world scenarios and ensure that they are equipped to provide the best possible care for their patients. Once in operation, the PIIC iX Demo broadcasts pre-recorded monitor simulation waveforms and data that can be processed and used to make a prediction.

The simulator comprised eight beds, each equipped with distinct equipment. Different equipment means different broadcasted data, and, unfortunately, not all beds transmit the intended data due to the variances in equipment. As a resolution, it was imperative to access the simulator definitions and modify the equipment assigned to beds whose data transmission was not the desired. This process entailed a considerable amount of trial and error. Eventually, the appropriate equipment was identified, although the amount available was not sufficient to make all the beds transmit the intended data. Nonetheless, there were sufficient beds to test the entire system which broadcast the wanted data for each reading done.

The trained model analyses the broadcasted information and makes predictions for future MAP values. However, instead of receiving the expected data in FHIR format, the system received data in HL7 format. In PIIC iX, the usage of the FHIR format is reserved for batch data transmission, whereas the HL7 format is employed for continuous data streaming. Given the requirement for real-time data processing and the fact that generating FHIR resources is more time-consuming compared to HL7, a decision was made to proceed with the available HL7-formatted data. It is essential to highlight that the format choice does not impact the actual content of the transmitted data.

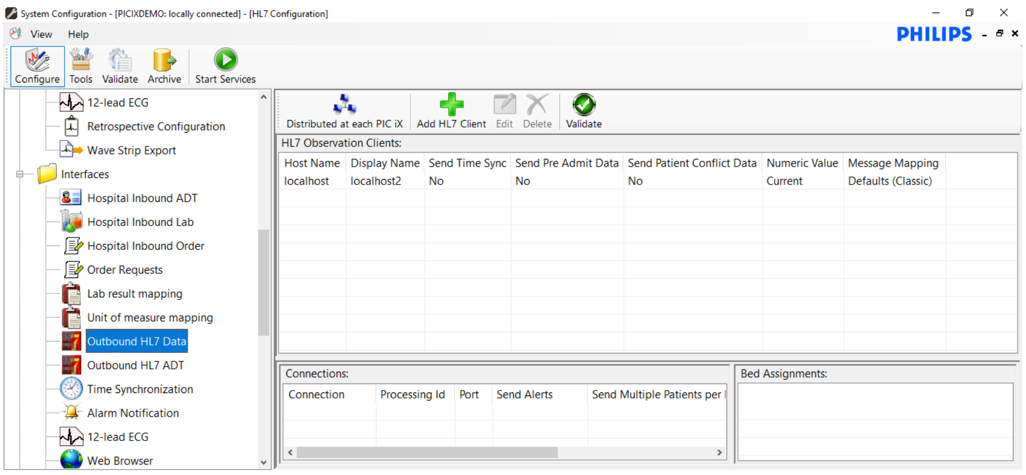
Moreover, the simulator runs in a virtual machine environment. Not only a virtual machine places high demands on the computer processing capabilities but also takes a lot of storage space so, it was decided to send the broadcasted information to the local machine. To simplify the exchange of data between the virtual machine and the local machine, a trusted and secured shared folder was configured. A shared folder is a directory accessible by multiple machines (in this case the virtual machine and the local machine), allowing for the seamless transfer of files. This configuration allows the simulator to write the generated information to a designated shared folder, which can later be accessed and analysed by the local machine. This approach facilitates the integration of data generated by the simulator with the rest of the system, enabling subsequent processing and analysis. To do that, some configurations were done. The next section explains all the configurations step by step.

4.2.1 Simulator Configuration

1. First of all, to install the simulator is necessary to have virtualisation software. The software used is Virtual Box [97], which is provided by Oracle [98] and can be downloaded from their website. After its installation, a new virtual machine is created using the provided virtual disc image.

2. Inside the simulator, a channel must be configured so that when the simulation is running, the generated data is sent to the desired port. In this case, the port in use is port 6661, and, as mentioned in the next section (Section 4.3), the port chosen will be the same as the one defined in Mirth (further details on Mirth are provided in the Section 4.3). It does not have to be mandatory to be port 6661. The only thing to bear in mind is that it must correspond to the mirth port, or else the data will be sent elsewhere and not received.

   The steps mentioned are done through the simulator settings (PIC iX System Configuration), more specifically in the tab "Outbound HL7 Data", as shown in Figure 4.4.



HL7 Observation Clients:

Host Name	Display Name	Send Time Sync	Send Pre Admit Data	Send Patient Conflict Data	Numeric Value	Message Mapping
localhost	localhost2	No	No	No	Current	Defaults (Classic)

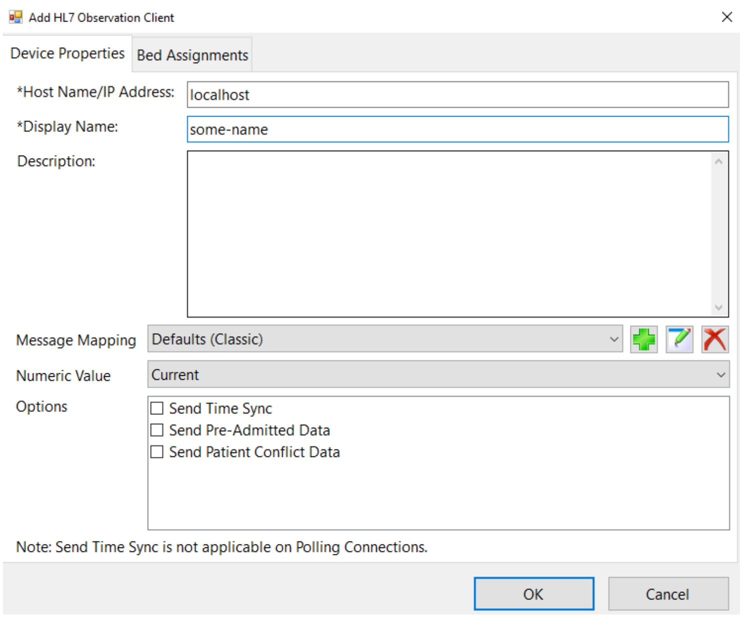
Connections:

Connection	Processing Id	Port	Send Alerts	Send Multiple Patients per

Bed Assignments:

Figure 4.4: Simulation configuration.

3. While on this tab, the button "Add HL7 Client" allows the creation of new channels and to do so, it is required to give it a name, define the port, as mentioned before, the beds that will send information and the frequency at which they send the data. Figure 4.5 shows the initial steps to configure a client.


Add HL7 Observation Client

Device Properties | Bed Assignments

*Host Name/IP Address: localhost

*Display Name: some-name

Description:

Message Mapping: Defaults (Classic)

Numeric Value: Current

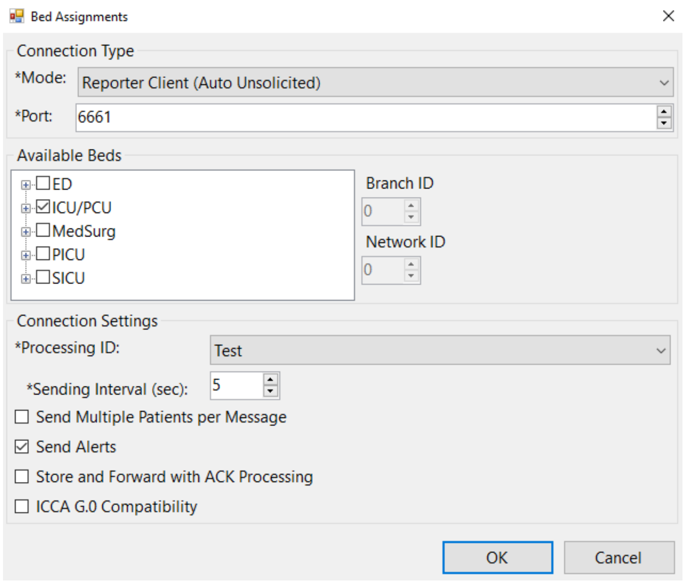
Options:
☐ Send Time Sync
☐ Send Pre-Admitted Data
☐ Send Patient Conflict Data

Note: Send Time Sync is not applicable on Polling Connections.

OK | Cancel

Figure 4.5: Configuration of an HL7 Observation Client.

4. In the tab "bed assignments", the beds to send information are chosen, which, in this case, will be all those present in the ICU. In addition to the beds, it is also chosen the transmission frequency, which has a minimum value of 5 seconds. Besides this, the simulator also allows sending extra information, such as some alerts. For that, simply select the desired checkboxes. Figure 4.6 shows the bed assignment process.


Bed Assignments

Connection Type

*Mode: Reporter Client (Auto Unsolicited)

*Port: 6661

Available Beds
☐ ED
☑ ICU/PCU
☐ MedSurg
☐ PICU
☐ SICU

Branch ID: 0

Network ID: 0

Connection Settings

*Processing ID: Test

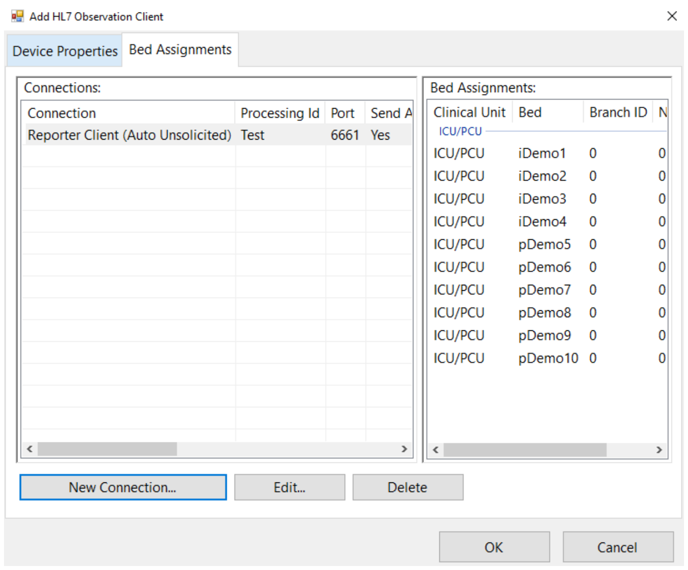
*Sending Interval (sec): 5

☐ Send Multiple Patients per Message
☑ Send Alerts
☐ Store and Forward with ACK Processing
☐ ICCA G.0 Compatibility

OK | Cancel

Figure 4.6: Bed assignments.

By the end of the process, the initial Bed Assignments tab should look like Figure 4.7.



Connections:

Connection	Processing Id	Port	Send A
Reporter Client (Auto Unsolicited)	Test	6661	Yes

Bed Assignments:

Clinical Unit	Bed	Branch ID	N
ICU/PCU			
ICU/PCU	iDemo1	0	0
ICU/PCU	iDemo2	0	0
ICU/PCU	iDemo3	0	0
ICU/PCU	iDemo4	0	0
ICU/PCU	pDemo5	0	0
ICU/PCU	pDemo6	0	0
ICU/PCU	pDemo7	0	0
ICU/PCU	pDemo8	0	0
ICU/PCU	pDemo9	0	0
ICU/PCU	pDemo10	0	0

Figure 4.7: Observation Client with beds assigned

5. It is also necessary to establish a method which enables the virtual machine to communicate with the local one. For the sake of simplicity, a shared folder was configured, which, as the name implies, is a feature that allows the access of files from a host system within a guest system.

   To establish a shared folder, first, the virtual machine must be turned on and the Guest Additions installed. To install the Guest Additions go to the window of a running virtual machine and select "Insert Guest Additions CD Image"from the "Devices"menu. Then, "This PC"from Windows File Explorer should have a new drive, and inside it, there is a new executable file which has to run to install the Guest Additions. Figure 4.8 shows how the “This PC” window should look.

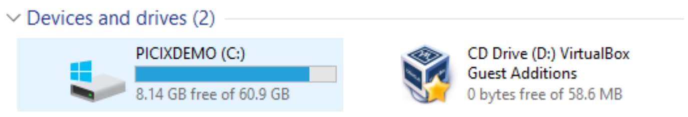
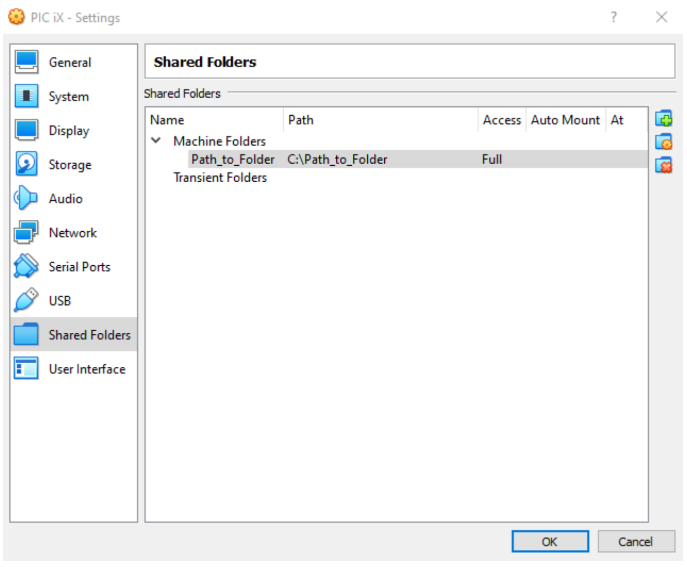


Figure 4.8: New drive in “This PC”.

6. To round off, in the top left corner of the Virtual Machine window, the option Shared Folder Settings from the Shared Folder option in the Devices menu has to be selected. In the new menu, it is only required to open the option to add a shared folder and set the path of the local machine to the intended folder. Figure 4.9 displays the new menu and its configuration.



Shared Folders

Shared Folders

Name	Path	Access	Auto Mount	At
Machine Folders				
Path_to_Folder	C:\Path_to_Folder	Full		
Transient Folders				

Figure 4.9: New shared folder.

7. To verify if the shared folder was properly configured, one can access the Network tab in Windows File Explorer and check if the folder is present. If that is the case, data can now be transferred between the two machines.

4.3 Data Transfer and Cleaning

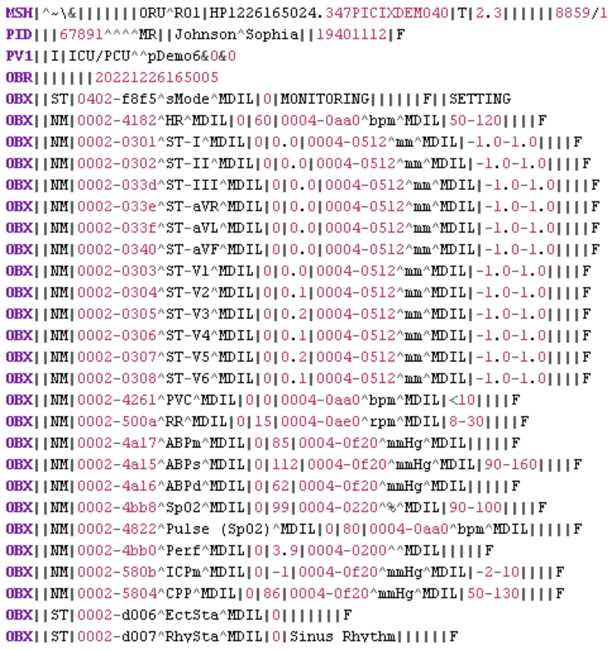
This activity regards the second and third layers of the architecture, responsible for the transmission of data and subsequent removal of any unuseful information from the incoming data obtained from the sensors, which are generated by the simulator. The purpose of these layers is to enable smooth transfer of data to the machine that is responsible for processing the information.

To transfer the information from the virtual machine to the local machine, it is necessary to have an application that works as a middleman. The application is responsible to receive the data and send it to the defined shared folder. The chosen application to do this work was Mirth Connect.

Mirth Connect is an open-source healthcare integration engine that allows organisations to efficiently and securely exchange health data between systems. It provides a scalable platform for transforming, routing, and transferring data between various healthcare systems, including EHRs, lab systems, and medical devices. With its intuitive interface, Mirth Connect makes it easy to manage complex data integration workflows and handle a wide range of data formats. Additionally, its support for a variety of protocols and standards makes it a versatile solution for healthcare data integration.

Mirth's framework is an ideal choice as it effectively handles data transmitted in HL7 format from the simulator. Mirth's framework provides a range of tools that facilitate data transformations, making it suitable for performing essential data cleaning operations.

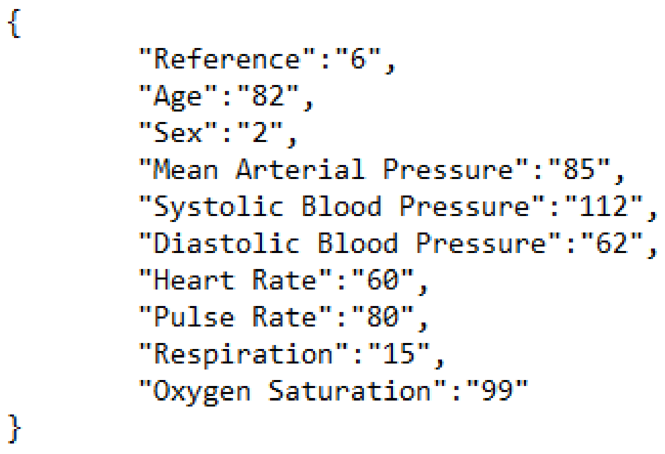
The framework enables the extraction and transformation of data, ensuring that only the relevant information required for the model's prediction is retained. This ensures that the data passed on to the subsequent layer is refined, streamlined, and devoid of unnecessary elements. Figure 4.10 illustrates an example of a received message.



```
MSH|^~\&|||||||ORU^R01|HP1226165024.347PICIXDEM040|T|2.3||||||8859/1
PID|||67891^^^^MR||Johnson^Sophia||19401112|F
PV1||I|ICU/PCU^^pDemo6&0&0
OBR|||||||20221226165005
OBX||ST|0402-f8f5^sMode^MDIL|0|MONITORING||||||F||SETTING
OBX||NM|0002-4182^HR^MDIL|0|60|0004-0aa0^bpm^MDIL|50-120||||F
OBX||NM|0002-0301^ST-I^MDIL|0|0.0|0004-0512^mm^MDIL|-1.0-1.0||||F
OBX||NM|0002-0302^ST-II^MDIL|0|0.0|0004-0512^mm^MDIL|-1.0-1.0||||F
OBX||NM|0002-033d^ST-III^MDIL|0|0.0|0004-0512^mm^MDIL|-1.0-1.0||||F
OBX||NM|0002-033e^ST-aVR^MDIL|0|0.0|0004-0512^mm^MDIL|-1.0-1.0||||F
OBX||NM|0002-033f^ST-aVL^MDIL|0|0.0|0004-0512^mm^MDIL|-1.0-1.0||||F
OBX||NM|0002-0340^ST-aVF^MDIL|0|0.0|0004-0512^mm^MDIL|-1.0-1.0||||F
OBX||NM|0002-0303^ST-V1^MDIL|0|0.0|0004-0512^mm^MDIL|-1.0-1.0||||F
OBX||NM|0002-0304^ST-V2^MDIL|0|0.1|0004-0512^mm^MDIL|-1.0-1.0||||F
OBX||NM|0002-0305^ST-V3^MDIL|0|0.2|0004-0512^mm^MDIL|-1.0-1.0||||F
OBX||NM|0002-0306^ST-V4^MDIL|0|0.1|0004-0512^mm^MDIL|-1.0-1.0||||F
OBX||NM|0002-0307^ST-V5^MDIL|0|0.2|0004-0512^mm^MDIL|-1.0-1.0||||F
OBX||NM|0002-0308^ST-V6^MDIL|0|0.1|0004-0512^mm^MDIL|-1.0-1.0||||F
OBX||NM|0002-4261^PVC^MDIL|0|0|0004-0aa0^bpm^MDIL|<10||||F
OBX||NM|0002-500a^RR^MDIL|0|15|0004-0ae0^rpm^MDIL|8-30||||F
OBX||NM|0002-4a17^ABPm^MDIL|0|85|0004-0f20^mmHg^MDIL|||||F
OBX||NM|0002-4a15^ABPs^MDIL|0|112|0004-0f20^mmHg^MDIL|90-160||||F
OBX||NM|0002-4a16^ABPd^MDIL|0|62|0004-0f20^mmHg^MDIL|||||F
OBX||NM|0002-4bb8^Sp02^MDIL|0|99|0004-0220^%^MDIL|90-100||||F
OBX||NM|0002-4822^Pulse (Sp02)^MDIL|0|80|0004-0aa0^bpm^MDIL|||||F
OBX||NM|0002-4bb0^Perf^MDIL|0|3.9|0004-0200^^MDIL|||||F
OBX||NM|0002-580b^ICPm^MDIL|0|-1|0004-0f20^mmHg^MDIL|-2-10||||F
OBX||NM|0002-5804^CPP^MDIL|0|86|0004-0f20^mmHg^MDIL|50-130||||F
OBX||ST|0002-d006^EctSta^MDIL|0|||||||F
OBX||ST|0002-d007^RhySta^MDIL|0|Sinus Rhythm||||||F
```

Figure 4.10: Received HL7 message.

Mirth's framework enables the user to create JavaScript scripts that can perform transformations on incoming data, particularly in the case of HL7 data. In this study, a small script was created to perform data cleaning on the data sent by the simulator. The script examines the received data, selects the relevant information required for the model, and discards unnecessary data for the subsequent steps. Once the data is processed, it is transmitted in a JSON format, following the structure depicted in Figure 4.11.



```
{
        "Reference":"6",
        "Age":"82",
        "Sex":"2",
        "Mean Arterial Pressure":"85",
        "Systolic Blood Pressure":"112",
        "Diastolic Blood Pressure":"62",
        "Heart Rate":"60",
        "Pulse Rate":"80",
        "Respiration":"15",
        "Oxygen Saturation":"99"
}
```

Figure 4.11: Generated JSON file.

For every message received, Mirth is capable of transforming the data into the pretended JSON file format, excluding any excess of data that was initially generated by the simulator. The cleaned data is then written in the configured shared folder. By removing the unnecessary data, Mirth facilitates a more streamlined and efficient data transfer process to the subsequent layers.

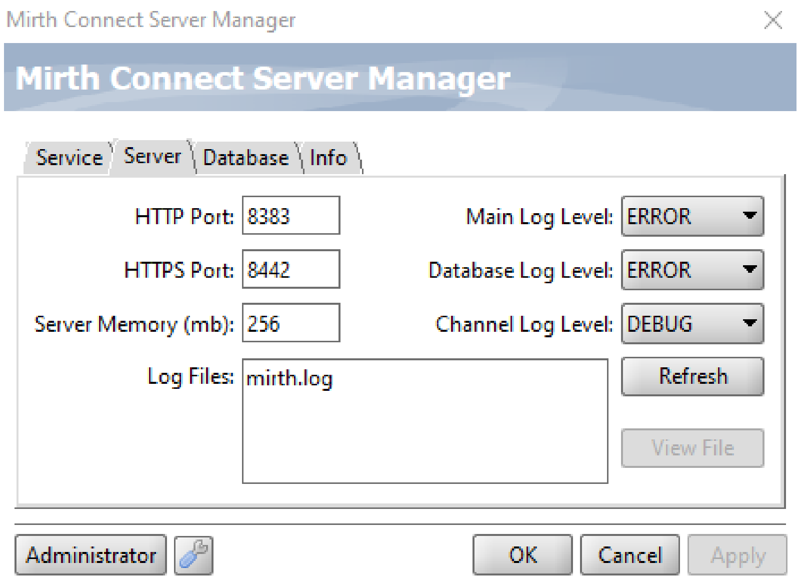
In the system, Mirth is employed once enough beds are transmitting the required data but first, some configurations had to be made for Mirth to do the desired work. Those configurations are described in the next section (Section 4.3.1).

4.3.1 Mirth Configuration

1. First and foremost, Java 8 [99] must be installed for Mirth to function. Installing Java 8 is possible through Oracle's official website. Once installed, Mirth installation can begin.

2. Mirth was installed on the virtual machine using the official website to download the executable file. The executable file installs two Mirth components: Server Manager and the Administrator Launcher, which have to be configured before the channel is activated.

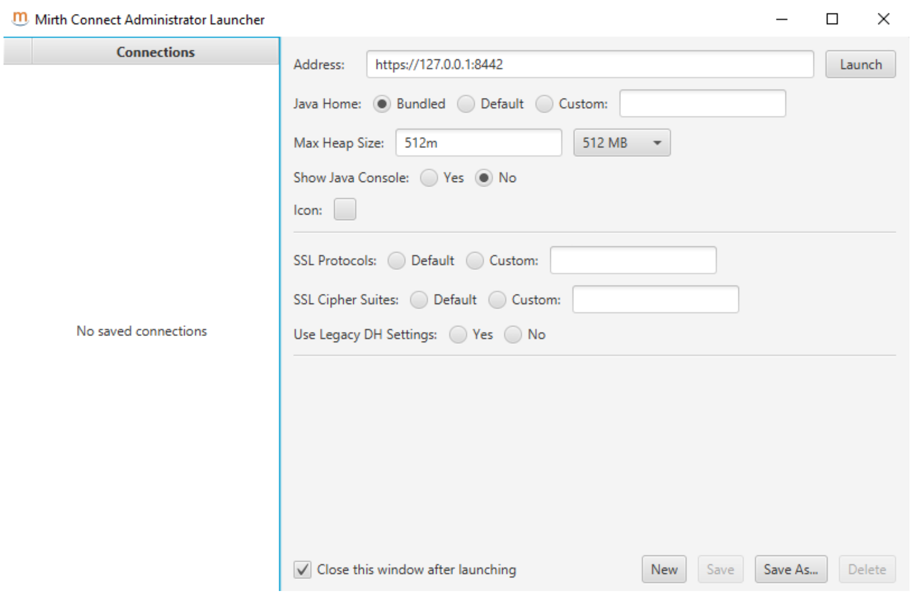
3. In the Server Manager is necessary to define the HTTP and Hypertext Transfer Protocol Secure (HTTPS) ports since it is in those that Mirth will be running. In this case, the ports chosen are 8383 and 8442, respectively, however, it is not necessary to be these two as long as there is no conflict with other ports already in use. Figure 4.12 shows the final configuration of the Server Manager.



HTTP Port: 8383
HTTPS Port: 8442
Server Memory (mb): 256
Log Files: mirth.log
Main Log Level: ERROR
Database Log Level: ERROR
Channel Log Level: DEBUG

Figure 4.12: Mirth Server Manager.

4. The port of the Internet Protocol (IP) address in which Mirth is launched has to match the HTTPS port chosen when configuring the Server Manager, which is 8442. Moreover, Mirth will be running on the local host represented as 127.0.0.1. This configuration is done in the Administrator Launcher before launching Mirth itself. Figure 4.13 shows the configuration of the Administrator Launcher, where it is possible to see that the address field is composed of the local host followed by the 8442 port.


Mirth Connect Administrator Launcher

Connections

No saved connections

Address: https://127.0.0.1:8442

Java Home: [✓ Bundled] Default Custom:

Max Heap Size: 512m 512 MB

Show Java Console: Yes [✓ No]

Icon:

SSL Protocols: Default Custom:

SSL Cipher Suites: Default Custom:

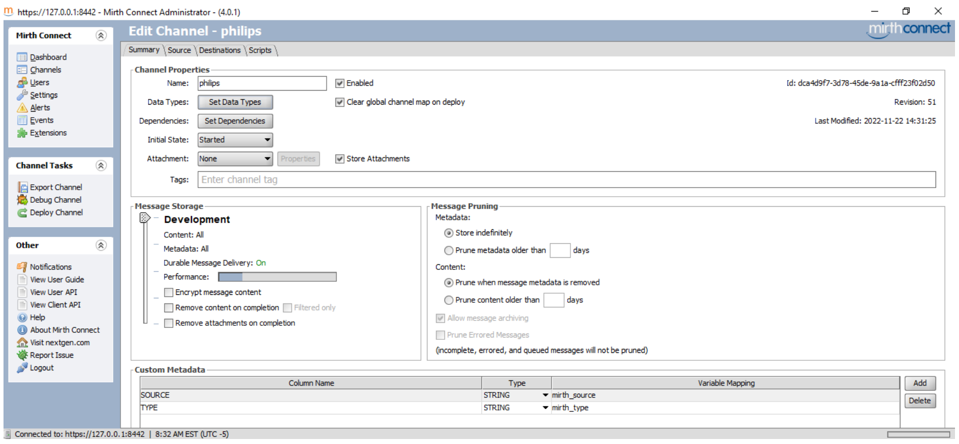
Use Legacy DH Settings: Yes No

[✓ Close this window after launching]

Figure 4.13: Mirth Administrator Launcher.

5. While on Mirth, it is necessary to do three different things to configure a channel. Firstly, the data type that will be received on Mirth has to be set to HL7 v2.x., which corresponds to the data sent by the simulator. Secondly, the channel connector type has to be a Transmission Control Protocol (TCP) Listener, and the local port in which the channel runs needs to correspond to the same as the simulator (6661).

6. Thirdly and lastly, Mirth provides tools to process the information received and send it elsewhere, making it possible to establish operations that aim to extract the required data and send it to the desired destination. The destination can be a folder, a web server or a local server, but in this case, the desired destination is the shared folder configured between the local machine and the virtual machine. It is also necessary to set the type of data and the name of the file which is going to be forwarded. The following figures show each of these steps. Figure 4.14 shows the first step, where the data type is chosen.



Channel Properties

Name: philips [✓ Enabled]
Id: dca4d9f7-3d78-45de-9a1a-cfff23f02d50
Data Types: Set Data Types [✓ Clear global channel map on deploy]
Revision: 51
Dependencies: Set Dependencies
Last Modified: 2022-11-22 14:31:25
Initial State: Started
Attachment: None [✓ Store Attachments]
Tags: Enter channel tag

Message Storage

Development
Content: All
Metadata: All
Durable Message Delivery: On
Performance:
Encrypt message content
Remove content on completion Filtered only
Remove attachments on completion

Message Pruning

Metadata:
[✓ Store indefinitely]
Prune metadata older than days
Content:
[✓ Prune when message metadata is removed]
Prune content older than days
[✓ Allow message archiving]
Prune Errored Messages
(incomplete, errored, and queued messages will not be pruned)

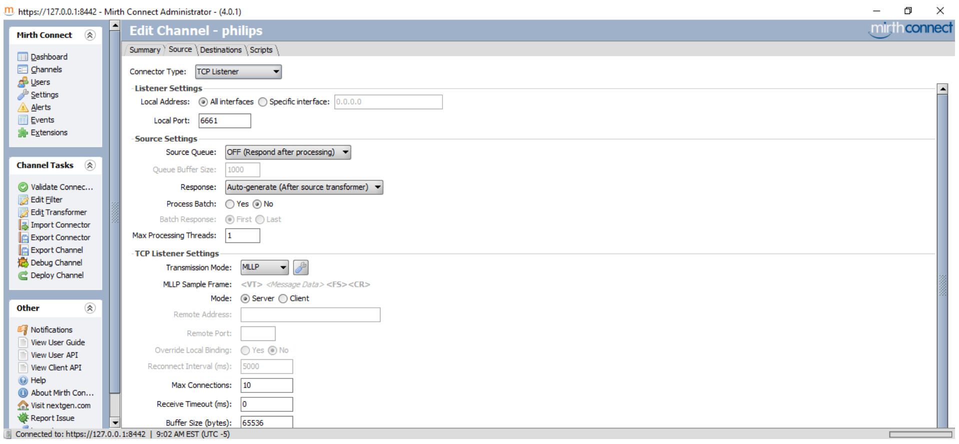
Custom Metadata

Column Name	Type	Variable Mapping
SOURCE	STRING	mirth_source
TYPE	STRING	mirth_type

Connected to: https://127.0.0.1:8442 | 8:32 AM EST (UTC -5)

Figure 4.14: Mirth Channel summary.

Figure 4.15 shows the channel type and port.



Connector Type: TCP Listener

Listener Settings
Local Address: [✓ All interfaces] Specific interface: 0.0.0.0
Local Port: 6661

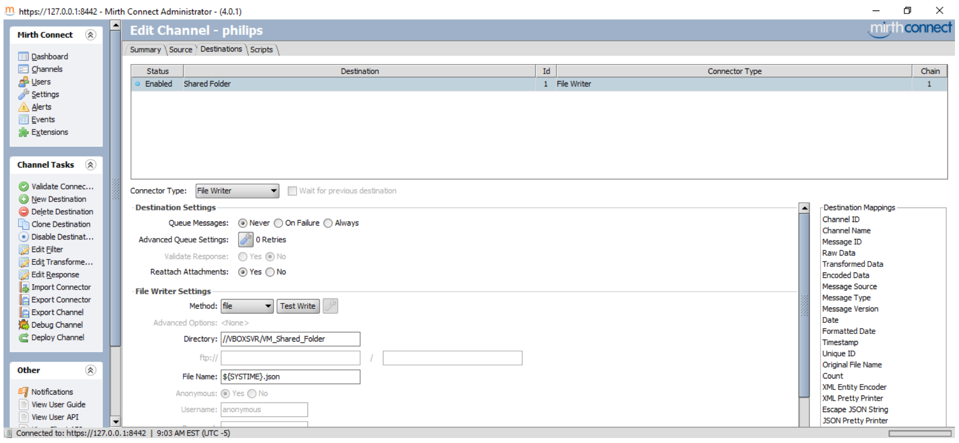
Source Settings
Source Queue: OFF (Respond after processing)
Queue Buffer Size: 1000
Response: Auto-generate (After source transformer)
Process Batch: Yes [✓ No]
Batch Response: [✓ First] Last
Max Processing Threads: 1

TCP Listener Settings
Transmission Mode: MLLP
MLLP Sample Frame: <VT> <Message Data> <FS><CR>
Mode: [✓ Server] Client
Remote Address:
Remote Port:
Override Local Binding: Yes [✓ No]
Reconnect Interval (ms): 5000
Max Connections: 10
Receive Timeout (ms): 0
Buffer Size (bytes): 65536

Connected to: https://127.0.0.1:8442 | 9:02 AM EST (UTC -5)

Figure 4.15: Mirth Channel source.

Figure 4.16 displays the configuration made to work on and forward a message.



Status	Destination	Id	Connector Type	Chain
Enabled	Shared Folder	1	File Writer	1

Connector Type: File Writer Wait for previous destination

Destination Settings
Queue Messages: [✓ Never] On Failure Always
Advanced Queue Settings: 0 Retries
Validate Response: Yes [✓ No]
Reattach Attachments: [✓ Yes] No

File Writer Settings
Method: file
Advanced Options: <None>
Directory: //VBOXSVR/VM_Shared_Folder
ftp://
File Name: ${SYSTIME}.json
Anonymous: [✓ Yes] No
Username: anonymous

Destination Mappings
Channel ID
Channel Name
Message ID
Raw Data
Transformed Data
Encoded Data
Message Source
Message Type
Message Version
Date
Formatted Date
Timestamp
Unique ID
Original File Name
Count
XML Entity Encoder
XML Pretty Printer
Escape JSON String
JSON Pretty Printer

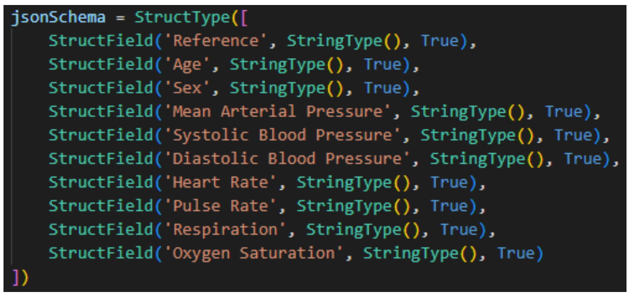
Connected to: https://127.0.0.1:8442 | 9:03 AM EST (UTC -5)

Figure 4.16: Mirth Channel destinations.

4.4 Data Streaming and Classification

Real-time data classification is an important task in many domains, including healthcare. In recent years, there has been a surge in the volume and velocity of data generated in real time, making it increasingly challenging to process and classify data promptly. To address this challenge, several tools have been developed, such as the ones analysed in Section 2.2.2. This section relates to the architecture's third layer and following the proposed solution, Spark Streaming was the tool used to stream the generated data, as it is stated to be an easy to implement tool and has multiple tools to manage incoming data. Section 2.2.2.2 dives into more detail about Spark Streaming. Its implementation was done through a small python script. Since Spark Streaming is capable of doing different tasks, it needs to know what it will be doing. In this case, its objective is to firstly, load the ML model and, secondly, keep actively checking if the shared folder has any new files so the model can process them.

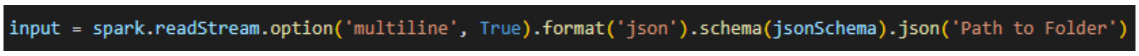
It is mandatory to specify two things: the JSON file schema and the tasks which Spark Streaming is responsible for. The JSON file has to be manually defined and must match the one sent by Mirth. Figure 4.17 shows the JSON schema.



```
jsonSchema = StructType([
    StructField('Reference', StringType(), True),
    StructField('Age', StringType(), True),
    StructField('Sex', StringType(), True),
    StructField('Mean Arterial Pressure', StringType(), True),
    StructField('Systolic Blood Pressure', StringType(), True),
    StructField('Diastolic Blood Pressure', StringType(), True),
    StructField('Heart Rate', StringType(), True),
    StructField('Pulse Rate', StringType(), True),
    StructField('Respiration', StringType(), True),
    StructField('Oxygen Saturation', StringType(), True)
])
```

Figure 4.17: JSON schema.

The task is defined by a simple code line shown in Figure 4.18.



```
input = spark.readStream.option('multiline', True).format('json').schema(jsonSchema).json('Path to Folder')
```

Figure 4.18: Spark Streaming task.

Afterwards, Spark Streaming starts running and actively monitors the shared folder for any new files. Once Mirth has processed and sent the data into a JSON format, and as soon as new files are detected, Spark Streaming automatically performs data processing on the file. The JSON file is parsed, and the contained information is classified before feeding to the model for prediction. Following the prediction, the result is written, along with the information received, in a .csv file. The saved information is then used for future model improvement by feeding the model with new data.

The use of Spark Streaming in this pipeline allows for real-time data processing, providing the ability to respond to incoming data as soon as it is available. Additionally, by storing the model prediction along with the original data in a .csv file, the system can keep track of previous predictions and adjust the model accordingly to improve its accuracy.

4.4.1 Difficulties with Spark Streaming

The installation of Spark Streaming was not straightforward. To be able to use Spark Streaming, it is necessary to install or download the zip file, of Hadoop, Java, and Python.

Scoop [100] was used to help with the Spark Streaming installation process. Scoop is a package manager for Windows that makes it easy to install and manage software from the command line. It enables users to install and manage applications, dependencies, and system tools without having to manually download and install each component. With Scoop, it is possible to install, update, and remove packages on a Windows computer with a few simple commands. Scoop provides a large repository of pre-compiled software packages, making it a convenient alternative to manual installations, especially for users who are not familiar with Windows command-line tools. It also adds the newly installed softwares to the environment variables automatically.

Java and Python were both installed using Scoop.

Spark Streaming is said to be compatible with Java 8/11/17, however, it was impossible to run the developed Spark Streaming script using version 8. By that, the version in use is 11.0.15. To install it, the instruction used in PowerShell was "scoop install microsoft11-jdk", which installs Java through the Microsoft website. It should be noted that Java 8 support is deprecated for versions of Hadoop higher than 3.2.0.

In Python's case, the 3.7 and above versions are compatible, and the version in use is 3.10.8. To install Python, the instruction was "scoop install python", which installs Python from the official website.

Finally, the Spark Streaming version being used is 3.1.3, not the most recent at the moment (3.3.0), and the Hadoop version in use is 3.2.2, also not the latest version (3.3). To install Spark Streaming, it is necessary to access the official website and download the corresponding zip file. After unzipping it, it is necessary to add the environment variable SPARK_HOME, which will be pointing to the path where Spark Streaming files were unzipped. Regarding Hadoop, to download it is necessary to access the following GitHub repository: `https://github.com/cdarlint/winutils`, download the version in use, and unzip it, just as done for Spark Streaming. Then, the HADOOP_HOME variable has to be added to the system variables and pointed to Hadoop files. In Hadoop's case, it is also necessary to add the bin folder to the path variables. After all the steps, the installation should be correctly done and completed.

4.5 Prediction

The fourth layer in the system architecture under consideration is the model's prediction, which plays a pivotal role in achieving the desired objective. As previously mentioned, Spark Streaming is responsible for loading the model, which is deemed to be the core element of this study, as it is designed to deliver the required prediction. Upon receiving the classified data from Spark Streaming, the model processes it and generates the intended output. Since the predictions yielded by the model exhibit a high degree of similarity to the actual values, it can be safely inferred that the developed work has been successful in achieving the intended objective.

To ensure the accuracy and reliability of the system, the system was tested multiple times. The repeated tests were instrumental in verifying the system's consistency and the variability of the model's outputs based on the received input values. Moreover, the input values received are not static. New values, that are distinct from the previous ones, are continuously received over time, resulting in new predictions being generated. The variation in the predicted values is evident from Figure 4.19 presented below. In the presented graph it is noticeable that the predictions are not constant and are identical to the next readings. The next readings refer to the next values read by the sensors or, to simplify, the real values. The red line represents the threshold, the blue and orange are related to predictions and readings, respectively, from patient 39, and the grey and yellow are the same but for the patient with identification number 6.

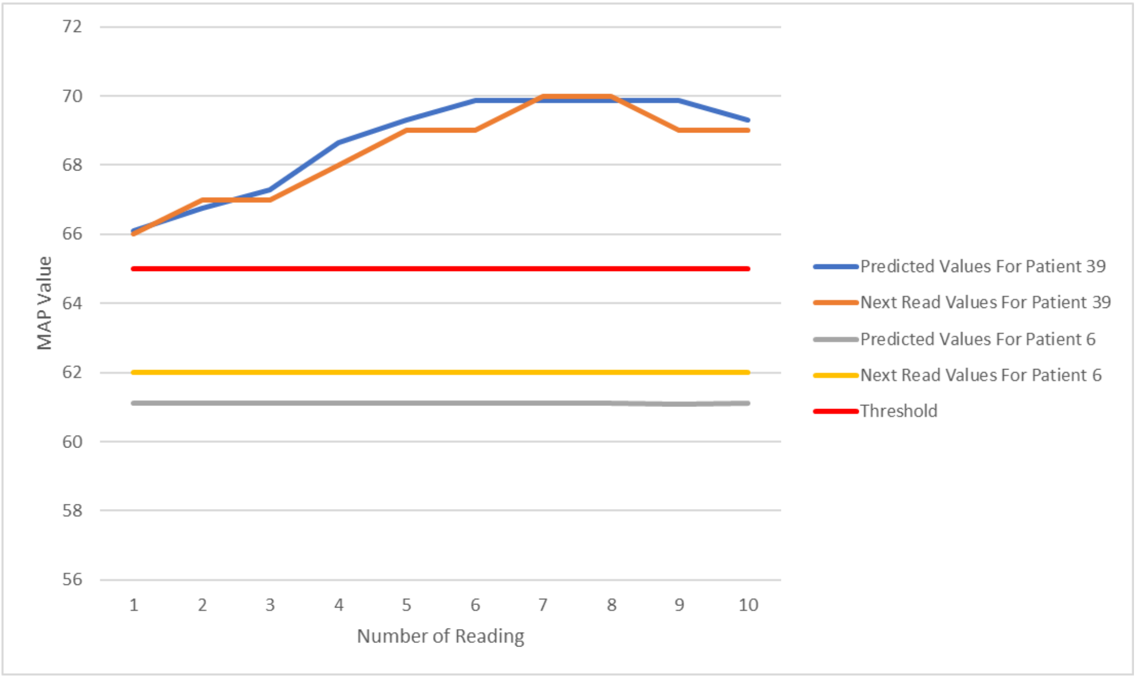


Figure 4.19: Graph with predictions and the following readings.

It can be inferred from the figure that the model's predictions are not constant and exhibit a noticeable degree of variability over time. This finding underscores the robustness and effectiveness of the model in generating accurate predictions based on the real-time data stream received from the sensors. The continuous flow of input data and the corresponding variability in the model's output values indicate that the system is well-equipped to adapt to changing conditions and provide accurate and reliable predictions over an extended period.

In addition to predicting the MAP value, the developed system includes an alarm mechanism that raises a visual alert on the computer when a predicted MAP value is below 65 mm Hg. However, as with any automated system, there is a possibility of system errors or poorly generated data. To prevent unnecessary alarm notifications, the system incorporates a count mechanism to track the persistence of such low MAP values. Specifically, the system counts the number of consecutive predictions that are below the 65 mm Hg threshold. If the count exceeds a predetermined threshold, an alarm is triggered to simulate a request for attention by healthcare professionals. It is important to note that this alarm mechanism is solely designed for testing purposes and should not be relied upon as a substitute for medical attention. In a real-world scenario, the alarm mechanism would be replaced with an audible or mobile notification to ensure that healthcare professionals can respond promptly and appropriately to any potentially critical situations. Overall, this alarm mechanism serves as an essential component of the system, ensuring that any abnormalities in the predicted MAP values are appropriately addressed and reducing the risk of harm to patients.

4.6 Explainability

Developing an XAI algorithm can be challenging, especially if the model is complex or not inherently interpretable. However, it is becoming more demanding, especially in fields where transparency is required, such as healthcare. Therefore, there is a growing interest in developing XAI algorithms that can provide interpretable and transparent explanations of the decision-making process of a model.

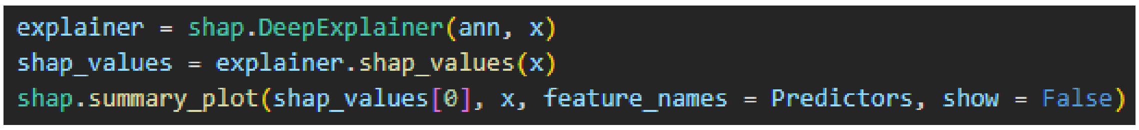
Several approaches exist for developing an XAI algorithm, which varies depending on the specific application domain and model architecture. It is important to emphasise that there is no one-size-fits-all solution, and developing an XAI algorithm requires careful consideration of the particular use case and model architecture. Moreover, it is worth noting that developing an XAI algorithm can be a demanding task, and it may not be feasible to create a single algorithm that can work for all trained models.

The applicability of an XAI technique depends on the nature of the model and the requirements of the specific use case. Some XAI techniques can be applied across a wide range of models with minimal modifications while others may need to be tailored to the specific requirements of a particular model. It is therefore essential to evaluate the strengths and limitations of different XAI techniques and select the one(s) that best meet the specific requirements of the different models.

In light of the above, it is generally advisable to develop an XAI algorithm for each trained model separately. This approach ensures that the algorithm is optimised for the specific model and use case, thereby increasing its interpretability and transparency. Due to its demanding nature, it was determined that an algorithm would be developed exclusively for the selected model, rather than for each of the trained models. Although not explicitly mentioned in the proposed solution, a model based on XAI was deemed a valuable addition to the ongoing study.

Based on Section 2.3, and because the model is an ANN, it was decided to use a model-agnostic approach, specifically a feature relevance estimation, as it enables healthcare professionals to know which variable were most impactful in the model's prediction. To help with the implementation, the SHAP library was used. As stated in Section 2.3, it provides a unified way of explaining the output of any ML model, and it can be used with various types of models such as tree-based models, deep learning models, and linear models (more information is provided in Section 2.3.

Its implementation consists of three code lines as shown in Figure 4.20, where the first instruction generates an explainer object for a given model and data, the second returns the matrix of values for the given set of input instances, and the last regards SHAP's final plot to interpret the model.



```
explainer = shap.DeepExplainer(ann, x)
shap_values = explainer.shap_values(x)
shap.summary_plot(shap_values[0], x, feature_names = Predictors, show = False)
```

Figure 4.20: SHAP instructions.

Figure 4.21 shows a plot example with multiple predictions made by the model and the features' importance. As can be seen, the features with the most impact on the model's prediction were the Mean Arterial Pressure and the Respiration Rate while the rest did not have a significant impact, if none. The feedback provided, can help increase the healthcare professionals trust towards the model as they can confirm that the model reached a conclusion based on the variables from which they themselves would use to reach the same conclusion.

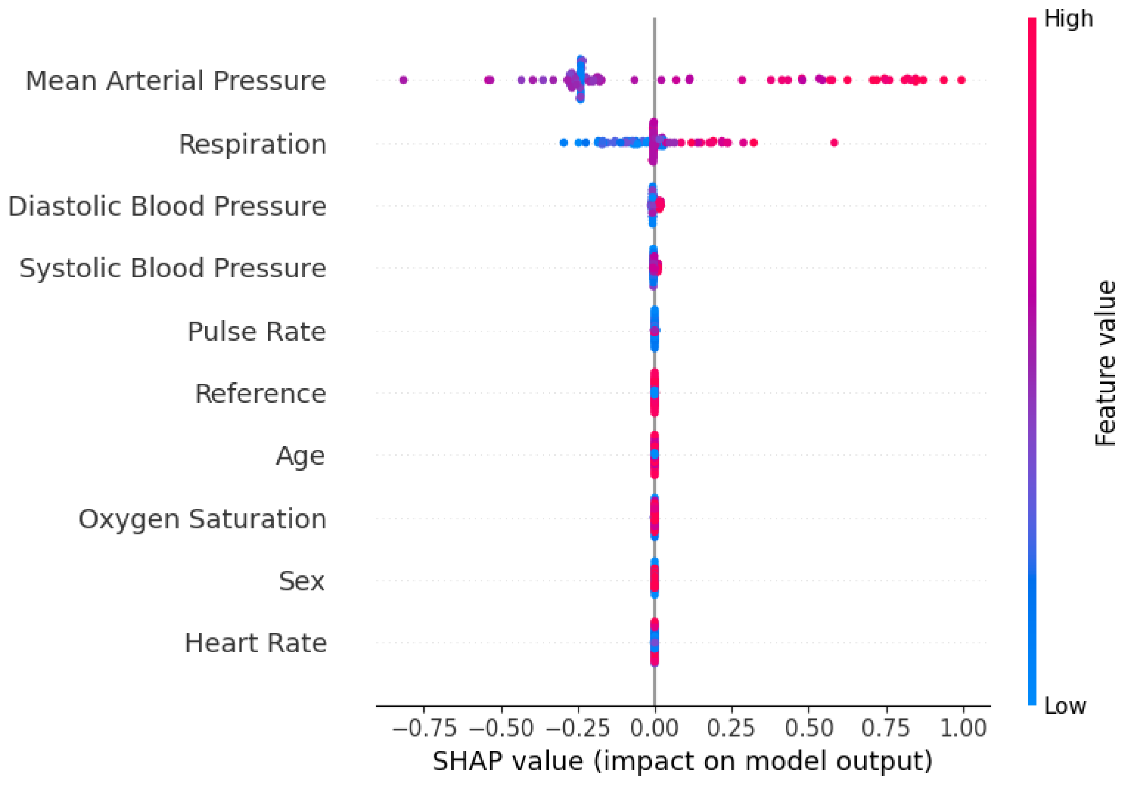


Figure 4.21: SHAP plot.

4.7 ICU4Covid Integration and Validation

After the system test was a success, it was decided to incorporate the solution developed in the CPS4TIC as foreseen and explained in Section 3.4, more precisely in the B-Health IoT Box [95].

The B-Health IoT Box is a key component of the CPS4TIC. It is an IoT device designed to be placed near a patient's bed in the ICU to support the health and well-being of healthcare professionals that consists of a Raspberry Pi and multiple environmental sensors such as CO2, temperature, humidity, and luminosity sensors. The B-Health IoT Box serves as a data collection and transmission hub. It collects vital patient data from ICU sensors, including from the PIIC iX, and sends the collected data to a centralised observation centre for further analysis and interpretation. Figure 4.22 shows the B-Health IoT Box as a data acquisition platform in an ICU environment.

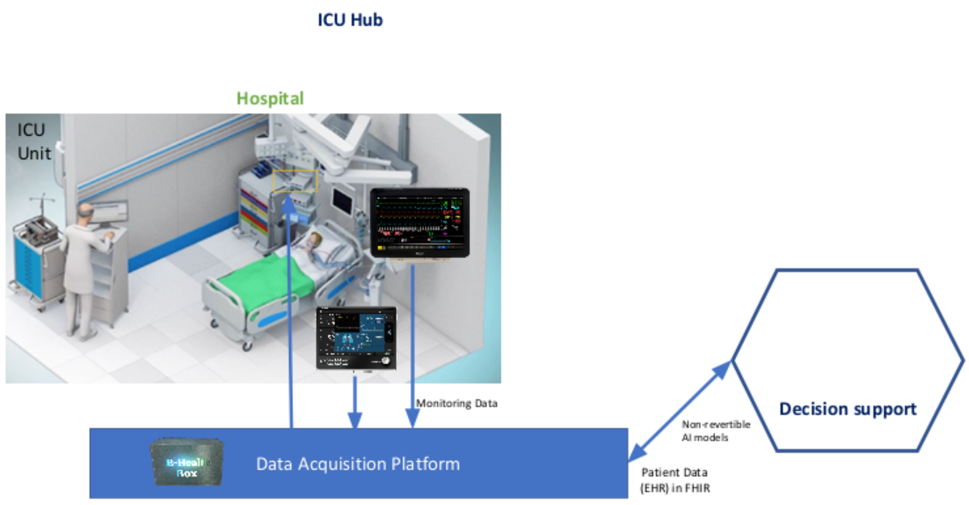


Figure 4.22: B-Health IoT Box in ICU environment.

One key features of the B-Health IoT Box is its versatility and adaptability. It is a customisable device that allows for the installation of various applications. These applications can enhance the functionality of the B-Health IoT Box, enabling additional features and functionalities based on specific requirements. By supporting the installation of applications, the B-Health IoT Box can be adapted to different healthcare settings and integrated with advanced algorithms, data processing tools, and decision support systems. Figure 4.23 depicts where the B-Health IoT Box is placed in a ICU environment and its function in the CPS4TIC component of the ICU4Covid project.

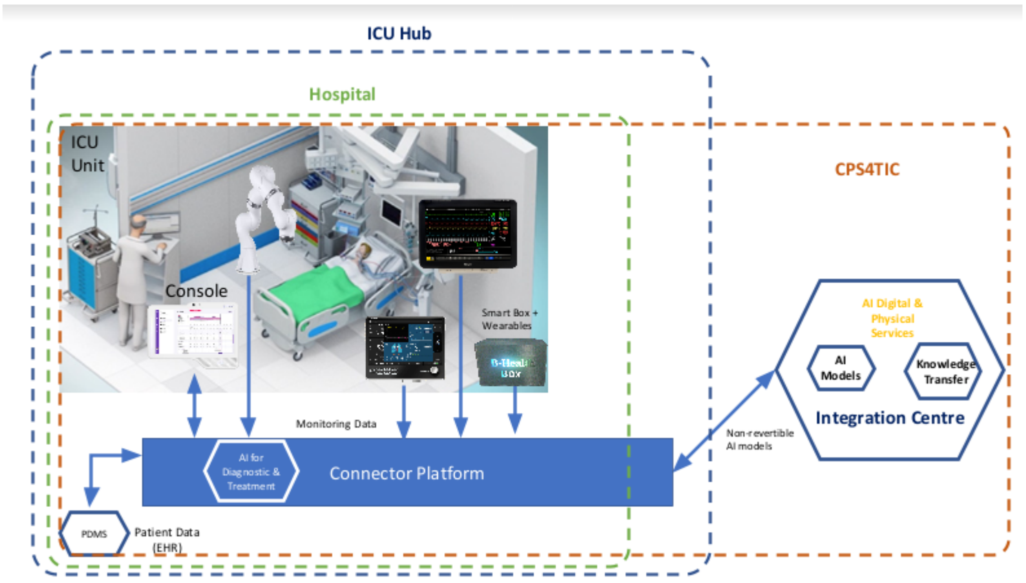


Figure 4.23: B-Health IoT Box in a ICU and CPS4TIC context.

The integration of the program with the B-Health IoT Box not only increases its functionality but also enables remote monitoring and management of the ICU environment, which is essential in telemedicine. Moreover, implementing the solution directly in the B-Health IoT Box could take advantage of the sensor data collected by the box itself, in addition to the PIIC iX signals. Hence, using the B-Health IoT Box to collect data from the ICU sensors and making it available to authorised hospital staff, including remote healthcare professionals, the B-Health IoT Box with the integrated program can support telemedicine practices, providing real-time information and facilitating remote decision-making. Furthermore, the integration can enhance the safety of healthcare professionals and patients, allowing for better control of the ICU environment, which is crucial in the context of telemedicine. A photo of the B-Health IoT Box is provided in Figure 4.24.

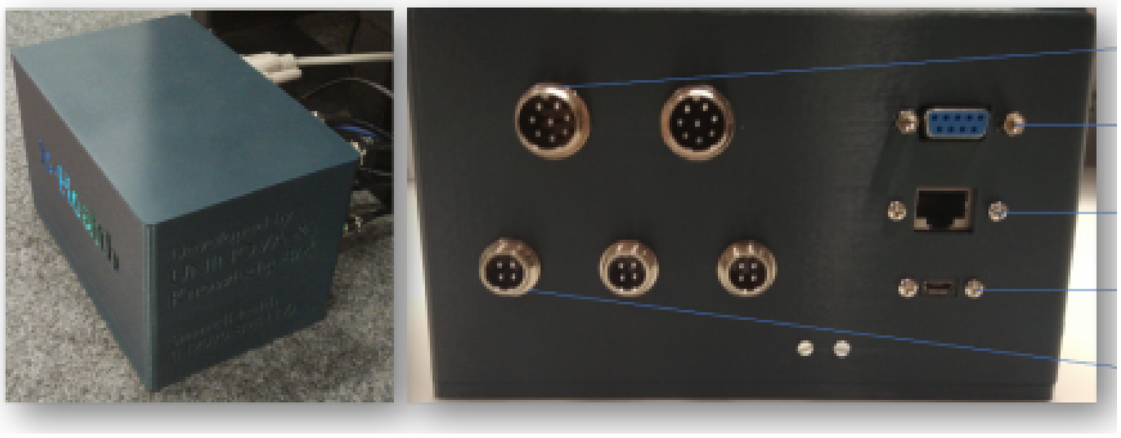
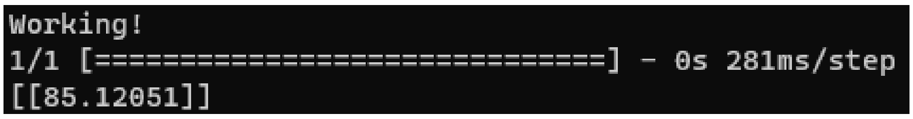


Figure 4.24: B-Health IoT Box [95].

To deploy the developed program into the B-Health IoT Box it is necessary to have access to its Raspberry. It was necessary to install several tools and libraries such as Java, Python and Spark Streaming, as well as multiple files that had to be transferred to execute the developed solution in the box and test its functionality. Although the setup process was time-consuming, the outcome was in accordance with the expected results. One thing that should be noted is that the box runs in a Linux environment and the version of Spark Streaming installed was not the same as the one installed when doing the initial tests. So, the version of Spark Streaming being used is the latest (3.3.2 and is already pre-built for Hadoop) and was installed using the Linux instructions designated for this. The Java and Python versions are the same. None of this affected the outcome of the program.

It is worth noting that the computing power of the B-Health IoT Box is inferior to that of a computer, resulting in slower program execution. However, this did not impede the consistency and accuracy of the program outputs and predictions, which remained unchanged across both machines. Also, considering that the training of the model (ANN network) is the most demanding activity in terms of physical resources, the training can be performed in advance in an external environment. An example of the solution prevision execution time (0.281s) in the B-Health IoT Box, is depicted in Figure 4.25, as well as a prevision.



```
Working!
1/1 [==============================] - 0s 281ms/step
[[85.12051]]
```

Figure 4.25: B-Health Box output.

The solution was subjected to testing in a simulation room at Hospital Dr. Melo Mendonça, a collaborative partner of the ICU4Covid project. The preliminary results of the integrated solution are encouraging, although its effectiveness in a real clinical setting has yet to be tested. These initial outcomes provide grounds for optimism regarding the potential of the solution. However, it is imperative to acknowledge that comprehensive assessment in a real clinical setting is necessary to validate its performance.

If the performance is maintained, the integration of the solution can even have the potential to enhance the field of telemedicine. This is due to the fact that the intelligent system is capable of evaluating a patient and making predictions, thus reducing the need for constant monitoring by a healthcare professional. Implementing such a system could result in increased efficiency, more timely interventions, and ultimately, improved patient outcomes. Nonetheless, it should be emphasised that such technology does not replace the need for medical expertise and clinical judgement. It is crucial that healthcare professionals continue to be promptly alerted when abnormalities are detected, enabling them to take appropriate actions on time.

5 Conclusion and Future Plans

The motivation for the presented work was the potential of AI in ICUs because these are areas where a lot of data is available, intervention has to be done quickly and are susceptible to errors. For this reason, it is essential to continuously innovate by bringing in new forms of technology.

The susceptibilities exposed during the COVID-19 pandemic were the main drivers of this dissertation and were what led to the development of this prototype that could be integrated with a telemedicine environment as the ICU4Covid. During its development, the AI potential in the health area has become evident, supporting healthcare professionals in decision making.

The primary objective of this study was to establish a system capable of managing medication administration and intensive care environments, however, during its research, it became interesting to focus more on blood pressure values, specifically MAP. MAP is an significant aspect in ICUs so, predicting future MAP values through the analysis of multiple variables collected from sensors ICU became the goal of this study. To achieve this goal, three distinct AI models were trained and tested: LR, SVM, and ANN. The purpose of developing these models was to perform a comparison of their respective performances. Moreover, the variables used for the model's training were all read by bedside monitors. Some examples are respiration rate, heart rate, pulse rate, oxygen saturation, besides the components of blood pressure (systolic, diastolic and MAP).

To make a fair comparison, the same dataset was used to train all three models, which enabled a more rigorous evaluation of their respective performances. Following the models' training it was noticeable that the training time for each model was different, with the SVM taking almost a day to finish its process, the LR finishing in mere minutes, and the ANN standing in the middle. One specificity of ANN training was that it was trained several times with different hyperparameters to check if there was any change in its performance and these were also found to have an impact on training time.

After evaluating the results, the ANN model was deemed to be the most suited for the task at hand. Even though the ANN did not get the highest accuracy (75.063%), it was not only the sole model that outputs continuous values, but it also did not have an excessive training time. Therefore, the ANN was chosen for further testing and validation. To validate the performance of the ANN model, the Phillips Simulator was used to generate patient data in real time, creating a controlled environment for testing the model. This allowed for a thorough evaluation of the ANN model's performance and helped to determine the model's potential for practical application in a hospital setting.

The proposed prediction system used Spark Streaming as a means of classifying the data generated from the Phillips Simulator. It was necessary to appeal to a simulator since testing a model directly in a hospital environment can be risky and should only be done after careful consideration of the potential risks and benefits, and after conducting thorough validation and testing to ensure that the model is accurate and reliable. The simulator's data was transmitted through the Mirth Connect software, which acted as a mediator between the simulator and the Spark Streaming. Upon receipt, Spark Streaming utilised its capabilities to process and categorise the incoming data for subsequent use in prediction by the ANN model. This approach allowed for a robust and efficient method of handling the large amounts of data generated by the simulator and facilitated the prediction process by the ANN model. In addition, Spark Streaming was also responsible for storing the received and processed data in a .csv file to later retrain the ANN.

After the implementation of the proposed system, an evaluation of the performance of the ANN was conducted. The results indicated that the ANN's predictions were consistent with the subsequent readings taken. This validates the hypothesis that the system can accurately predict future MAP. The successful implementation and evaluation of the proposed system suggests that it has the potential to be scaled up and applied in a larger clinical settings. The system's ability to accurately predict future MAP could improve patient care and outcomes by providing clinicians with real-time information about a patient's blood pressure. With additional testing and validation in larger populations, the system could become a valuable tool for clinicians seeking to improve patient care and outcomes.

To enhance the trustworthiness of the system, an XAI algorithm was trained to provide a clear explanation of how the ANN arrived at a particular output, thereby boosting a healthcare professional confidence in the system. The developed XAI model incorporates an interface that displays the various input variables fed into the ANN and their respective contributions towards the final output. This explanation serves as an important tool for clinicians to understand how the ANN makes its predictions, as it aids in justifying the decisions made by the model. Ultimately, the XAI model improves the transparency of the system and enhances the interpretability of the ANN, making it more useful for clinical decision-making.

Lastly, the developed program was deployed in the B-Health Box, which is a component of theCPS4TIC in the ICU4Covid project and serves as a tool for supporting clinicians in the ICU environment, connecting real data to a telemedicine environment. The developed program takes advantage of the data collected by the B-Health Box sensors to make MAP predictions and provide additional functionalities to the box. After the initial configurations, and once the program was successfully deployed, on the B-Health Hub, no significant differences were observed in comparison to the previously tested version. The predictions made by the program were identical.

Further studies and improvements to the system may be necessary to ensure its reliability and accuracy, but the current results are promising.

5.1 Future Plans

Although the implementation of the system was successful, several areas merit further consideration. Firstly, even though the trained model was successfully integrated into the ICU4Covid project, it was not integrated into the Federated Learning component. However, future enhancements aim to include the model in the Federated Learning aspect of the project. This would allow for a distributed and collaborative approach to training and refining the model using data from multiple healthcare institutions while preserving patient privacy. By incorporating the trained model into the Federated Learning aspect, the accuracy and generalisation capabilities of the model can be improved, which can ultimately lead to better patient outcomes. Also, utilising the most recent dataset would not only provide a larger amount of data but also incorporate the latest and diverse information, thereby improving the model's performance by uncovering new variables whose impact may be positive on its predictions.

Secondly, the use of a simulator for data generation and transmission in the current study was adequate for the study's purposes. However, it should be noted that the simulator used is a past version of the real software. As such, it would be beneficial to integrate the model with a more recent version of the software and other systems, so that it is not only compatible with Philips' systems. This would enable the model to be more versatile and adaptive to the changing technological landscape of the healthcare industry. Furthermore, the addition of new sensor variables to the model would also enhance its adaptability and robustness. Moreover, the communication between these systems must be secure and robust to protect the patients' privacy and prevent any unauthorised access to their data. This can be achieved through the use of technologies such as encryption, Virtual Private Network (VPN), firewalls, two-factor authentication, or other technologies as long as they can ensure safe data transmission.

Thirdly, a warning mechanism should be improved. This will ensure that healthcare professionals are promptly notified and take appropriate action to address the situation. The warning mechanism should be designed to be easy for healthcare professionals to understand the alert and take appropriate action. The design should also take into account the potential for false positives and false negatives and provide appropriate guidance to minimise these occurrences. Moreover, it is important to note that the alarm should not only be present in a monitor, as healthcare professionals may miss the notification and would not be capable of monitoring patients from a distance. Therefore, additional notification methods such as an audible signal or cellphone communication are viable solutions. These methods can provide healthcare professionals with immediate notification, enabling them to take prompt action to address the situation. By improving the warning mechanism, healthcare professionals can act quickly and prevent adverse events from occurring, ultimately improving patient outcomes.

Finally, the success of the model in predicting MAP suggests that it could potentially be applied to other clinical scenarios. However, any expansion of the model's use must be done with careful consideration of ethical and regulatory requirements. Before implementing the model in new use cases, it would be necessary to seek approval from an appropriate ethical committee. This would ensure that any potential risks to patients are identified and mitigated, and that the model's benefits are maximised while protecting patient privacy and autonomy. By following ethical guidelines and seeking appropriate approvals, the model's potential impact on patient care can be further realised.

Bibliography

[1] J. M. Lourenço. *The NOVAthesis LaTeX Template User's Manual*. NOVA University Lisbon. 2021. URL: https://github.com/joaomlourenco/novathesis/raw/master/template.pdf (cit. on p. iii).

[2] M. Haenlein and A. Kaplan. "A Brief History of Artificial Intelligence: On the Past, Present, and Future of Artificial Intelligence". In: 61.4 (2019-07), pp. 5–14. ISSN: 21628564. DOI: 10.1177/0008125619864925 (cit. on p. 1).

[3] A. Kaplan and M. Haenlein. "Siri, Siri, in my hand: Who's the fairest in the land? On the interpretations, illustrations, and implications of artificial intelligence". In: *Business Horizons* 62.1 (2019-01), pp. 15–25. ISSN: 0007-6813. DOI: 10.1016/J.BUSHOR.2018.08.004 (cit. on p. 1).

[4] D. L. Poole and A. K. Mackworth. *Artificial Intelligence: Foundations of Computational Agents*. Cambridge University Press, 2017-09. ISBN: 9781108164085. DOI: 10.1017/9781108164085 (cit. on p. 1).

[5] V. Kaul, S. Enslin, and S. A. Gross. "History of artificial intelligence in medicine". In: *Gastrointestinal Endoscopy* 92.4 (2020-10), pp. 807–812. ISSN: 0016-5107. DOI: 10.1016/J.GIE.2020.06.040 (cit. on p. 2).

[6] IBM. *Clinical Decision Support | Watson Health | IBM*. URL: https://www.ibm.com/watson-health/solutions/clinical-decision-support (visited on 2022-12-03) (cit. on p. 2).

[7] M. N. Ahmed, A. S. Toor, K. O'Neil, and D. Friedland. "Cognitive Computing and the Future of Health Care". In: *IEEE Pulse* 8.3 (2017-05), pp. 4–9. ISSN: 21542287. DOI: 10.1109/MPUL.2017.2678098 (cit. on pp. 2, 9).

[8] A. Powell. *Risks and benefits of an AI revolution in medicine*. 2020-11. URL: https://news.harvard.edu/gazette/story/2020/11/risks-and-benefits-of-an-ai-revolution-in-medicine/ (visited on 2022-12-03) (cit. on pp. 2, 33).

[9] S. O'Meara. "China's data-driven dream to overhaul health care". In: *Nature* 598.7879 (2021-10). ISSN: 14764687. DOI: 10.1038/D41586-021-02694-1 (cit. on p. 2).

[10] *Portugal regista novo máximo diário de infeções com covid-19*. 2021-01. URL: https://www.jn.pt/nacional/portugal-regista-novo-maximo-diario-de-infecoes-com-covid-19-13229848.html (visited on 2022-12-03) (cit. on p. 3).

[11] *Number of COVID-19 patients in hospital*. URL: https://ourworldindata.org/grapher/current-covid-patients-hospital?country=~PRT (visited on 2022-12-04) (cit. on p. 3).

[12] *Number of COVID-19 patients in intensive care (ICU)*. URL: https://ourworldindata.org/grapher/current-covid-patients-icu?country=~PRT (visited on 2022-12-04) (cit. on p. 3).

[13] *December 2022 | ICU4Covid*. 2022. URL: https://www.icu4covid.eu/december2022 (visited on 2023-03-31) (cit. on p. 4).

[14] ICU4Covid. *ICU4Covid*. URL: https://www.icu4covid.eu/ (visited on 2023-06-06) (cit. on p. 4).

[15] I. Project-CORDIS. *Cyber-Physical Intensive Care Medical System for Covid-19*. 2020. DOI: 10.3030/101016000. URL: https://cordis.europa.eu/project/id/101016000 (visited on 2022-12-13) (cit. on p. 4).

[16] M. A. De Georgia, F. Kaffashi, F. J. Jacono, and K. A. Loparo. "Information Technology in Critical Care: Review of Monitoring and Data Acquisition Systems for Patient Care and Research". In: *The Scientific World Journal* 2015 (2015). ISSN: 1537744X. DOI: 10.1155/2015/727694 (cit. on p. 5).

[17] J. Oliveira, N. Vafaei, V. Delgado-Gomes, P. Figueiras, C. Agostinho, and R. Jardim-Gonçalves. "Adaptive Learning and AI to Support Medication Management". In: *29th ICE IEEE/ITMC Conference, Edinburgh* (2023) (cit. on p. 7).

[18] C. Bartneck, C. Lütge, A. Wagner, and S. Welsh. *An Introduction to Ethics in Robotics and AI*. SpringerBriefs in Ethics. Springer Cham, 2021. ISBN: 978-3-030-51109-8. DOI: 10.1007/978-3-030-51110-4 (cit. on pp. 9, 11).

[19] S. Brown. *Machine learning, explained*. 2021-04. URL: https://mitsloan.mit.edu/ideas-made-to-matter/machine-learning-explained (visited on 2022-12-03) (cit. on p. 9).

[20] W. AI. *What is a machine learning model?* 2021. URL: https://learn.microsoft.com/en-us/windows/ai/windows-ml/what-is-a-machine-learning-model (visited on 2022-11-09) (cit. on p. 10).

[21] A. Ławrynowicz and V. Tresp. "Introducing Machine Learning". In: *Perspectives On Ontology Learning*. Vol. 18. 2014-01, pp. 35–50 (cit. on pp. 10, 12, 13).

[22] E. Ilkou and M. Koutraki. "Symbolic Vs Sub-symbolic AI Methods: Friends or Enemies?" In: *CSSA'20: Workshop on Combining Symbolic and Sub-Symbolic Methods and their Applications*. 2020. DOI: 10.1145/3340531.3414072 (cit. on p. 10).

[23] *Amazon Machine Learning Developer Guide*. 2022. URL: https://docs.aws.amazon.com/machine-learning/index.html (cit. on p. 10).

[24] A. Géron. *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow Concepts, Tools, and Techniques to Build Intelligent Systems*. 2019. ISBN: 978-1-492-03264-9 (cit. on pp. 10, 11, 19, 21, 39, 54).

[25] TensorFlow. *TensorFlow*. URL: https://www.tensorflow.org/guide (visited on 2023-06-20) (cit. on p. 11).

[26] Keras. *Keras*. URL: https://keras.io/ (visited on 2023-06-20) (cit. on p. 11).

[27] *What is data labeling?* URL: https://aws.amazon.com/sagemaker/data-labeling/what-is-data-labeling/ (visited on 2022-12-04) (cit. on p. 11).

[28] J. Brownlee. *Difference Between Classification and Regression in Machine Learning*. 2019-05. URL: https://machinelearningmastery.com/classification-versus-regression-in-machine-learning/ (visited on 2022-12-03) (cit. on p. 12).

[29] S. Sah. "Machine Learning: A Review of Learning Types". In: (2020-07). DOI: 10.20944/preprints202007.0230.v1 (cit. on pp. 12, 13).

[30] E. K. Hashi, M. S. Uz Zaman, and M. R. Hasan. "An expert clinical decision support system to predict disease using classification techniques". In: *ECCE 2017 - International Conference on Electrical, Computer and Communication Engineering*. 2017-04, pp. 396–400. ISBN: 978-1-5090-5627-9. DOI: 10.1109/ECACE.2017.7912937 (cit. on p. 13).

[31] A. Ribeiro, F. Portela, M. Santos, A. Abelha, J. Machado, and F. Rua. "Patients' Admissions in Intensive Care Units: A Clustering Overview". In: *Selected Papers from the Workshop on Intelligent Systems and Applications in Healthcare (ISA'Health 2016)* (2017-02). ISSN: 2078-2489. DOI: 10.3390/INFO8010023. URL: https://doi.org/10.3390/info8010023 (cit. on p. 14).

[32] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2018. ISBN: 9780262352703 (cit. on p. 14).

[33] J. Rafati and D. C. Noelle. "Learning Representations in Model-Free Hierarchical Reinforcement Learning". In: *Proceedings of the AAAI Conference on Artificial Intelligence* (2019-10). DOI: 10.1609/aaai.v33i01.330110009. URL: http://arxiv.org/abs/1810.10096 (cit. on p. 14).

[34] R. Dastres and M. Soori. "Artificial Neural Network Systems". In: *International Journal of Imaging and Robotics* 21 (2021), pp. 13–25. ISSN: 2231–525X (cit. on pp. 15–17).

[35] A. Hange. *Target Prediction using Single-layer Perceptron and Multilayer Perceptron*. 2021-04. URL: https://medium.com/nerd-for-tech/flux-prediction-using-single-layer-perceptron-and-multilayer-perceptron-cf82c1341c33 (visited on 2022-12-04) (cit. on p. 16).

[36] *What is Deep Learning and Neural Network*. 2022. URL: https://www.thewindowsclub.com/deep-learning-and-neural-network (visited on 2022-12-04) (cit. on p. 17).

[37] B. Shickel, T. J. Loftus, L. Adhikari, T. Ozrazgat-Baslanti, A. Bihorac, and P. Rashidi. "DeepSOFA: A Continuous Acuity Score for Critically Ill Patients using Clinically Interpretable Deep Learning". In: *Scientific Reports* (2019-02). ISSN: 2045-2322. DOI: 10.1038/s41598-019-38491-0 (cit. on pp. 17, 18).

[38] S. Lambden, P. F. Laterre, M. M. Levy, and B. Francois. "The SOFA score - Development, utility and challenges of accurate assessment in clinical trials". In: *Critical Care* (2019-11). ISSN: 1466609X. DOI: 10.1186/S13054-019-2663-7/TABLES/4 (cit. on p. 17).

[39] E. Martinez-Ríos, L. Montesinos, M. Alfaro-Ponce, and L. Pecchia. "A review of machine learning in hypertension detection and blood pressure estimation based on clinical and physiological data". In: *Biomedical Signal Processing and Control* 68 (2021-07), p. 102813. ISSN: 1746-8094. DOI: 10.1016/J.BSPC.2021.102813 (cit. on p. 19).

[40] S. Abaimov and M. Martellini. *Understanding Machine Learning*. Cambridge University Press, 2014. ISBN: 978-1-107-05713-5. DOI: 10.1007/978-3-030-91585-8_2 (cit. on p. 19).

[41] P. R. F. Rodrigues, J. M. da Silva Monteiro Filho, and J. P. do Vale Madeiro. "Chapter 7 - The Issue of Automatic Classification of Heartbeats". In: *Developments and Applications for ECG Signal Processing: Modeling, Segmentation, and Pattern Recognition*. Academic Press, 2019, pp. 169–193. ISBN: 9780128140369. DOI: 10.1016/B978-0-12-814035-2.00013-X (cit. on p. 19).

[42] A. Munther, A. Alalousi, S. Nizam, R. R. Othman, and M. Anbar. "Network traffic classification - A comparative study of two common decision tree methods: C4.5 and Random forest". In: *2014 2nd International Conference on Electronic Design, ICED 2014* (). DOI: 10.1109/ICED.2014.7015800 (cit. on p. 20).

[43] M. W. Gardner and S. R. Dorling. "Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences". In: *Atmospheric Environment* 32.14-15 (1998-08), pp. 2627–2636. ISSN: 1352-2310. DOI: 10.1016/S1352-2310(97)00447-0 (cit. on p. 20).

[44] M. Buscema. "Back Propagation Neural Networks". In: *Substance Use & Misuse* 33.2 (2009-07), pp. 233–270. ISSN: 10826084. DOI: 10.3109/10826089809115863 (cit. on p. 20).

[45] C. C. Lee, P. C. Chung, J. R. Tsai, and C. I. Chang. "Robust radial basis function neural networks". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 29.6 (1999), pp. 674–685. ISSN: 1941-0492. DOI: 10.1109/3477.809023 (cit. on p. 20).

[46] J. C. Platt. "Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines". In: *Advances in Kernel Methods-Support Vector Learning* (1998-04) (cit. on p. 21).

[47] M. C. Moghadam, E. M. K. Abad, N. Bagherzadeh, D. Ramsingh, G. P. Li, and Z. N. Kain. "A machine-learning approach to predicting hypotensive events in ICU settings". In: *Computers in Biology and Medicine* 118 (2020-03), p. 103626. ISSN: 0010-4825. DOI: 10.1016/J.COMPBIOMED.2020.103626 (cit. on pp. 21, 24).

[48] A. E. Johnson, T. J. Pollard, L. Shen, L. W. H. Lehman, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. Anthony Celi, and R. G. Mark. "MIMIC-III, a freely accessible critical care database". In: *Scientific Data* 3.1 (2016-05), pp. 1–9. ISSN: 2052-4463. DOI: 10.1038/sdata.2016.35 (cit. on p. 21).

[49] M. Cherifa, A. Blet, A. Chambaz, E. Gayat, M. Resche-Rigon, and R. Pirracchio. "Prediction of an Acute Hypotensive Episode During an ICU Hospitalization With a Super Learner Machine-Learning Algorithm". In: *Anesthesia and analgesia* 130.5 (2020-05), pp. 1157–1166. ISSN: 1526-7598. DOI: 10.1213/ANE.0000000000004539 (cit. on pp. 22, 24).

[50] M. Saeed, M. Villarroel, A. T. Reisner, G. Clifford, L. W. Lehman, G. Moody, T. Heldt, T. H. Kyaw, B. Moody, and R. G. Mark. "Multiparameter Intelligent Monitoring in Intensive Care II (MIMIC-II): A public-access intensive care unit database". In: *Critical care medicine* (2011-05). ISSN: 15300293. DOI: 10.1097/CCM.0B013E31820A92C6 (cit. on p. 22).

[51] M. C. Moghadam, E. Masoumi, S. Kendale, and N. Bagherzadeh. "Predicting hypotension in the ICU using noninvasive physiological signals". In: *Computers in Biology and Medicine* 129 (2021-02), p. 104120. ISSN: 0010-4825. DOI: 10.1016/J.COMPBIOMED.2020.104120 (cit. on pp. 22, 24).

[52] J. Lee and R. G. Mark. "An investigation of patterns in hemodynamic data indicative of impending hypotension in intensive care". In: *BioMedical Engineering Online* 9 (2010-10). ISSN: 1475-925X. DOI: 10.1186/1475-925X-9-62/FIGURES/5. URL: https://biomedical-engineering-online.biomedcentral.com/articles/10.1186/1475-925X-9-62 (cit. on pp. 22, 24).

[53] R. Patnaik, M. Chandran, S. C. Lee, A. Gupta, and C. Kim. "Predicting the occurrence of essential hypertension using annual health records". In: *Proceedings of 2018 2nd International Conference on Advances in Electronics, Computers and Communications, ICAECC 2018*. Institute of Electrical and Electronics Engineers Inc., 2018. ISBN: 978-1-5386-3785-2. DOI: 10.1109/ICAECC.2018.8479458 (cit. on pp. 22, 24).

[54] M. Nour and K. Polat. “Automatic Classification of Hypertension Types Based on Personal Features by Machine Learning Algorithms”. In: *Mathematical Problems in Engineering* (2020). ISSN: 15635147. DOI: 10.1155/2020/2742781 (cit. on pp. 23, 24).

[55] “PPG Blood Pressure”. In: *IEEE Dataport* (2019). URL: http://dx.doi.org/10.21227/crpd-4b52 (cit. on p. 23).

[56] E. W. Y. Kwong, H. Wu, and G. K. H. Pang. “A prediction model of blood pressure for telemedicine”. In: *Health Informatics Journal* 24.3 (2018), pp. 227–244. ISSN: 17412811. DOI: 10.1177/1460458216663025 (cit. on pp. 23, 24).

[57] Y. Luo, Y. Li, Y. Lu, S. Lin, and X. Liu. “The prediction of hypertension based on convolution neural network”. In: *2018 IEEE 4th International Conference on Computer and Communications, ICCC 2018*. Institute of Electrical and Electronics Engineers Inc., 2018-12, pp. 2122–2127. ISBN: 978-1-5386-8339-2. DOI: 10.1109/COMPCOMM.2018.8780834 (cit. on pp. 23, 24).

[58] R. Patgiri and A. Ahmed. “Big Data: The V’s of the Game Changer Paradigm”. In: *Proceedings - 18th IEEE International Conference on High Performance Computing and Communications, 14th IEEE International Conference on Smart City and 2nd IEEE International Conference on Data Science and Systems, HPCC/SmartCity/DSS 2016*. Institute of Electrical and Electronics Engineers Inc., 2017, pp. 17–24. ISBN: 978-1-5090-4297-5. DOI: 10.1109/HPCC-SMARTCITY-DSS.2016.0014 (cit. on pp. 25, 26).

[59] K. Adam, I. Hammad, M. Adam, I. Fakharaldien, J. M. Zain, and M. A. Majid. “Big Data Analysis and Storage”. In: 2015 (cit. on p. 25).

[60] A. S. Roy. *How does facebook handle the 4+ petabyte of data generated per day? Cambridge Analytica - facebook data scandal.* 2020-09. URL: https://medium.com/@srank2000/how-facebook-handles-the-4-petabyte-of-data-generated-per-day-ab86877956f4 (visited on 2022-12-04) (cit. on p. 25).

[61] M. E. Arass, I. Tikito, and N. Souissi. “Data lifecycles analysis: Towards intelligent cycle”. In: *2017 Intelligent Systems and Computer Vision, ISCV 2017*. Institute of Electrical and Electronics Engineers Inc., 2017. ISBN: 978-1-5090-4062-9. DOI: 10.1109/ISACV.2017.8054938 (cit. on pp. 26, 27).

[62] J. Zakir, T. Seymour, and K. Berg. “Big Data Analytics”. In: *Issues in Information Systems* 16.2 (2015), pp. 81–90 (cit. on p. 27).

[63] I. Ahmed, M. Ahmad, G. Jeon, and F. Piccialli. “A Framework for Pandemic Prediction Using Big Data Analytics”. In: *Big Data Research* 25 (2021-07). ISSN: 2214-5796. DOI: 10.1016/J.BDR.2021.100190 (cit. on p. 27).

[64] B. Ristevski and M. Chen. "Big Data Analytics in Medicine and Healthcare". In: *Journal of Integrative Bioinformatics* 15.3 (2018-05). ISSN: 1613-4516. DOI: 10.1515/JIB-2017-0030 (cit. on pp. 27, 33).

[65] J. Archenaa and E. A. Anita. "A Survey of Big Data Analytics in Healthcare and Government". In: *Procedia Computer Science* 50 (2015), pp. 408–413. ISSN: 1877-0509. DOI: 10.1016/J.PROCS.2015.04.021 (cit. on p. 28).

[66] A. Singh, A. Khamparia, and A. K. Luhach. "Performance comparison of Apache Hadoop and Apache Spark". In: *ICAICR '19*. Association for Computing Machinery, 2019. ISBN: 9781450366526. DOI: 10.1145/3339311.3339329 (cit. on p. 28).

[67] A. Hadoop. *Apache Hadoop*. URL: https://hadoop.apache.org/ (visited on 2023-06-20) (cit. on p. 28).

[68] A. Spark. *Apache Spark*. URL: https://spark.apache.org/ (visited on 2023-06-20) (cit. on p. 29).

[69] A. Storm. *Apache Storm*. URL: https://storm.apache.org/index.html (visited on 2023-06-20) (cit. on p. 29).

[70] R. Sint, S. Stroka, S. Schaffert, and R. Ferstl. "Combining Unstructured, Fully Structured and Semi-Structured Information in Semantic Wikis." In: *4th Semantic Wiki Workshop (SemWiki 2009) at the 6th European Semantic Web Conference (ESWC 2009)*. 2009-06 (cit. on pp. 30, 31).

[71] I. C. Education. *Structured vs. Unstructured Data: What's the Difference?* 2021-06. URL: https://www.ibm.com/cloud/blog/structured-vs-unstructured-data (visited on 2022-12-04) (cit. on p. 30).

[72] E. Ford, M. Oswald, L. Hassan, K. Bozentko, G. Nenadic, and J. Cassell. "Should free-text data in electronic medical records be shared for research? A citizens' jury study in the UK". In: *Journal of medical ethics* (2020-06). ISSN: 1473-4257. DOI: 10.1136/MEDETHICS-2019-105472 (cit. on p. 31).

[73] B. Siwicki. *Top 5 nightmares hiding in a healthcare organization's unstructured data*. 2021-05. URL: https://www.healthcareitnews.com/news/top-five-nightmares-hiding-healthcare-organizations-unstructured-data (visited on 2022-12-04) (cit. on p. 31).

[74] S. Karim, C. Fegeler, D. Boeckler, L. H. Schwartz, H. U. Kauczor, and H. von Tengg-Kobligk. "Development, Implementation, and Evaluation of a Structured Reporting Web Tool for Abdominal Aortic Aneurysms". In: *JMIR Research Protocols* (2013). ISSN: 19290748. DOI: 10.2196/RESPROT.2417 (cit. on p. 31).

[75] O. of the National Coordinator for Health Information Technology. *What Is FHIR®?* URL: http://www.hl7.org/fhir (cit. on p. 32).

[76] A. Walinjkar and J. Woods. “FHIR Tools for Healthcare Interoperability”. In: *Biomedical Journal of Scientific & Technical Research* (2018). ISSN: 2574-1241. DOI: 10.26717/BJSTR.2018.09.001863 (cit. on p. 32).

[77] M. Ayaz, M. F. Pasha, M. Y. Alzahrani, R. Budiarto, and D. Stiawan. “The Fast Health Interoperability Resources (FHIR) Standard: Systematic Literature Review of Implementations, Applications, Challenges and Opportunities”. In: *JMIR Med Inform* 9.7 (2021-07). ISSN: 22919694. DOI: 10.2196/21929 (cit. on p. 32).

[78] J. R. Henry, D. Lynch, J. Mals, S. P. Shashikumar, A. Holder, A. Sharma, and S. Nemati. “A FHIR-Enabled Streaming Sepsis Prediction System for ICUs”. In: *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*. Vol. 2018-July. Institute of Electrical and Electronics Engineers Inc., 2018-10, pp. 4093–4096. ISBN: 978-1-5386-3646-6. DOI: 10.1109/EMBC.2018.8513347 (cit. on p. 32).

[79] M. Singer, C. S. Deutschman, C. Seymour, M. Shankar-Hari, D. Annane, M. Bauer, R. Bellomo, G. R. Bernard, J. D. Chiche, C. M. Coopersmith, R. S. Hotchkiss, M. M. Levy, J. C. Marshall, G. S. Martin, S. M. Opal, G. D. Rubenfeld, T. D. Poll, J. L. Vincent, and D. C. Angus. “The Third International Consensus Definitions for Sepsis and Septic Shock (Sepsis-3)”. In: *JAMA* 315.8 (2016-02), p. 801. ISSN: 15383598. DOI: 10.1001/JAMA.2016.0287 (cit. on pp. 32, 38).

[80] G. Carra, J. I. Salluh, F. J. da Silva Ramos, and G. Meyfroidt. “Data-driven ICU management: Using Big Data and algorithms to improve outcomes”. In: *Journal of Critical Care* 60 (2020-12), pp. 300–304. ISSN: 0883-9441. DOI: 10.1016/J.JCRC.2020.09.002 (cit. on p. 33).

[81] T. H. Payne. “Computer Decision Support Systems”. In: *Chest* 118.2 (2000-09), 47S–52S. ISSN: 0012-3692. DOI: 10.1378/CHEST.118.2_SUPPL.47S (cit. on p. 33).

[82] G. Kou and W. Wu. “Multi-criteria decision analysis for emergency medical service assessment”. In: *Annals of Operations Research 2014* 223.1 (2014-05), pp. 239–254. ISSN: 1572-9338. DOI: 10.1007/S10479-014-1630-6 (cit. on p. 34).

[83] A. Barredo Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. Garcia, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, and F. Herrera. “Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI”. In: *Information Fusion* 58 (2020-06), pp. 82–115. ISSN: 1566-2535. DOI: 10.1016/J.INFFUS.2019.12.012 (cit. on pp. 34–36).

[84] SHAP. *SHAP*. URL: https://shap.readthedocs.io/en/latest/index.html (visited on 2023-06-20) (cit. on p. 35).

[85] S. M. Lundberg, P. G. Allen, and S.-I. Lee. “A Unified Approach to Interpreting Model Predictions”. In: *Advances in Neural Information Processing Systems*. Vol. 30. 2017. DOI: 10.5555/3295222.3295230 (cit. on p. 35).

[86] A. Adadi and M. Berrada. “Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI)”. In: *IEEE Access* 6 (2018-09), pp. 52138–52160. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2018.2870052 (cit. on p. 36).

[87] M. Hallengren, P. Åstrand, S. Eksborg, H. Barle, and C. Frostell. “Septic shock and the use of norepinephrine in an intermediate care unit: Mortality and adverse events”. In: *PLOS ONE* (2017-08). ISSN: 1932-6203. DOI: 10.1371/JOURNAL.PONE.0183073 (cit. on p. 38).

[88] Y. Tang, S. M. Brown, J. Sorensen, and J. B. Harley. “Physiology-Informed Real-Time Mean Arterial Blood Pressure Learning and Prediction for Septic Patients Receiving Norepinephrine”. In: 68.1 (2021-01), pp. 181–191. ISSN: 1558-2531. DOI: 10.1109/TBME.2020.2997929 (cit. on p. 38).

[89] Kaggle. *Cardiovascular Disease dataset*. URL: https://www.kaggle.com/datasets/sulianova/cardiovascular-disease-dataset (visited on 2023-06-16) (cit. on p. 40).

[90] Kaggle. *Heart Attack Analysis & Prediction Dataset*. URL: https://www.kaggle.com/datasets/rashikrahmanpritom/heart-attack-analysis-prediction-dataset (visited on 2023-06-16) (cit. on p. 40).

[91] G. B. Moody and R. G. Mark. “A database to support development and evaluation of intelligent intensive care monitoring”. In: *Computers in Cardiology* (1996), pp. 657–660. ISSN: 0276-6574. DOI: 10.1109/CIC.1996.542622 (cit. on p. 40).

[92] N. Vafaei, V. Delgado-Gomes, C. Agostinho, and R. Jardim-Gonçalves. “Analysis of Data Normalization in Decision Making Process for ICU’s Patients During the Pandemic”. In: *Procedia Computer Science.er* (2022) (cit. on p. 46).

[93] G. Paragliola and A. Coronato. “Definition of a novel federated learning approach to reduce communication costs”. In: *Expert Systems with Applications* 189 (2022-03). ISSN: 0957-4174. DOI: 10.1016/J.ESWA.2021.116109 (cit. on p. 46).

[94] N. Healthcare. *Healthcare Integration Engine*. URL: https://www.nextgen.com/solutions/interoperability/mirth-integration-engine (visited on 2023-06-19) (cit. on p. 49).

[95] F. Januário, C. Lopes, V. Delgado-Gomes, C. Agostinho, and M. Marques. “Smarterization of Medical Device using a CPS approach”. In: *CEUR Workshop Proceedings* 3214 (2022). ISSN: 1613-0073 (cit. on pp. 49, 70, 72).

[96] Philips. *Philips Health Care*. URL: https://www.philips.pt/healthcare/product/HCNOCTN171/intellivue-information-center-piic-ix (visited on 2023-06-19) (cit. on p. 55).

[97] V. Box. *Oracle VM VirtualBox*. URL: https://www.virtualbox.org/ (visited on 2023-06-20) (cit. on p. 57).

[98] Oracle. *Oracle*. URL: https://www.oracle.com/ (visited on 2023-06-20) (cit. on p. 57).

[99] Oracle. *Information about Java 8*. URL: https://www.java.com/en/download/help/java8.html (visited on 2023-06-19) (cit. on p. 62).

[100] Scoop. *Scoop*. URL: https://scoop.sh/ (visited on 2023-06-19) (cit. on p. 66).

2023 ADOPTION OF BIG DATA AND AI METHODS TO MANAGE MEDICATION ADMINISTRATION AND INTENSIVE CARE ENVIRONMENTS João Oliveira

NOVA SCHOOL OF