

Embedded AI for Real-time Health State Assessment and Treatment Recommendation in Rescue Operations

DISSERTATION
to obtain the degree of
Doctor of Engineering

submitted by
M.Sc Abu Shad Ahammed

submitted to the School of Science and Technology
of the University of Siegen
Siegen 2025

Supervisor and first appraiser
Prof. Dr.-Ing. Roman Obermaisser
University of Siegen

Second appraiser
Prof. Dr.-Ing. habil. Ali Jannesari
Iowa State University

Date of the oral examination: 18.06.2025

Abstract

Rescue emergencies are generally quite strenuous and challenging because they deal with human lives in a situation where it is difficult to apprehend precisely the health status of a distressed patient due to personal and work limitations. Often the rescue personnel have to deal with high levels of mental and physical stress, trauma, and emotional strain during rescue operations, especially when dealing with injured and vulnerable victims. This certainly impacts their decision-making abilities and overall well-being. Time, the main constraint in such rescue situations, also plays a vital role in decision-making. Therefore, it is necessary to recognize relevant situations i.e. health complications of the rescue patients on site, and to take appropriate first aid measures. The situation may change during the further course of initial treatment, with rescue workers judging this primarily by the visible condition of the emergency patient, the data from medical equipment (e.g., ventilator, ECG), and the mission description from the control room. Such changes are required to be responded to immediately, for example by resuscitation or appropriate medication. That's why, waiting for results from medical tests like MRI and ECG which are time-consuming and not suitable for emergency cases is not considered suitable for rescue cases. Considering these technical constraints, the doctoral thesis focuses on employing artificial intelligence (AI) models in two aspects that can expedite and improve the rescue process. Firstly, for the diagnosis of health complications in rescue situations and secondly to identify the correct medications as a part of initial treatment. A major part of this research is focused on advanced data analysis techniques that were used to extract information from 12 years of rescue records of 273,283 cases in the German city of Siegen-Wittgenstein. The initial data received from the rescue station was raw and in many cases contained incomprehensible information for which Natural Language Processing (NLP) techniques were applied to extract and interpolate relevant attributes. Subsequently, a detailed method for creating various AI models to promptly detect six key complications— Cardiovascular, Respiratory, Psychiatric, Neurological, Metabolic, and Abdominal—was conducted and is detailed in this dissertation. To develop the detection models for each complication, Artificial Intelligence(AI) algorithms like machine learning including both classical and deep learning approaches were used. To train these models attributes like patients' medical history, health diagnoses including neurological assessment, vital signs, initial impression of the rescue personnel, administered medications, and other treatment paths were used. During the course of development, one primary objective was to identify the model achieving the greatest accuracy and precision. Based on this re-

search, Extreme Gradient Boosting (XGB) and Random Forest (RF) algorithms were found as the most promising, showcasing accuracy rates ranging from 80% to 96%. After recognizing health complications, further research was done to find out if AI can also be implemented to determine possible medications based on detected complications and patients' health vitals. The result achieved from it also was impressive with accuracy close to 80%. AI models are further tested by deploying them into various accelerators, such as ARM processors, FPGAs and microcontrollers, to evaluate their performance based on inference time. The overall focus of this research is to overcome the rescue challenges in real-time by recognizing rescue situations and improving the quality of care and efficiency of rescue personnel.

Kurzfassung

Rettungseinsätze sind in der Regel äußerst herausfordernd, da sie mit menschlichem Leben in einer Situation umgehen, in der es aufgrund arbeitsbedingter Einschränkungen schwierig ist, den Gesundheitszustand eines hilfsbedürftigen Patienten genau einzuschätzen. Häufig müssen Rettungskräfte während der Einsätze mit hohen psychischen und physischen Belastungen, Traumata und emotionalem Stress umgehen, insbesondere im Umgang mit verletzten und gefährdeten Opfern. Dies beeinträchtigt ihre Entscheidungsfähigkeit. Die Zeit, der Hauptfaktor in solchen Rettungssituationen, spielt ebenfalls eine entscheidende Rolle bei der Entscheidungsfindung. Daher ist es notwendig, relevante Situationen, d. h. die gesundheitlichen Komplikationen der Patienten vor Ort, zu erkennen und entsprechende Erste-Hilfe-Maßnahmen zu ergreifen. Die Situation kann sich im weiteren Verlauf der Erstbehandlung ändern, wobei Rettungskräfte dies in erster Linie anhand des sichtbaren Zustands des Notfallpatienten, der Daten von medizinischen Geräten (z. B. Beatmungsgerät, EKG) und der Einsatzbeschreibung aus der Leitstelle beurteilen. Solche Änderungen müssen sofort berücksichtigt werden, beispielsweise durch Wiederbelebung oder die Verabreichung geeigneter Medikamente. Aus diesem Grund gelten zeitaufwändige medizinische Tests wie MRT und EKG, die sich für Notfälle nicht eignen, als ungeeignet für Rettungseinsätze. Angesichts dieser technischen Einschränkungen konzentriert sich die vorliegende Doktorarbeit auf den Einsatz von künstlicher Intelligenz (KI) in zwei Aspekten, die den Rettungsprozess beschleunigen und verbessern können: Erstens zur Diagnose von gesundheitlichen Komplikationen in Rettungssituationen und zweitens zur Identifikation der Medikamente im Rahmen der Erstbehandlung. Ein wesentlicher Teil dieser Forschung widmet sich fortschrittlichen Datenanalysetechniken, die angewandt wurden, um Informationen aus 12 Jahren Rettungsdaten von 273.283 Fällen in der deutschen Stadt Siegen-Wittgenstein zu extrahieren. Die ursprünglichen Daten, die von der Rettungsleitstelle empfangen wurden, waren unstrukturiert und enthielten in vielen Fällen unverständliche Informationen, für die Natural Language Processing (NLP)-Techniken eingesetzt wurden, um relevante Attribute zu extrahieren und zu interpolieren. Daraufhin wurde eine detaillierte Methode zur Erstellung verschiedener KI-Modelle entwickelt, um sechs zentrale Komplikationen – Kardiovaskuläre, respiratorische, psychiatrische, neurologische, metabolische und abdominale – schnell zu erkennen. Für die Entwicklung der Erkennungsmodelle für jede Notfälle wurden Algorithmen der KI, einschließlich maschinellem Lernen und sowohl klassischen als auch Deep-Learning-Ansätzen, eingesetzt. Zur Schulung dieser Modelle wurden Attribute wie die Krankengeschichte der Patienten, Diagnosen einschließlich neurologischer Be-

wertungen, Vitalzeichen, der Ersteindruck der Rettungskräfte, verabreichte Medikamente und andere Behandlungspfade verwendet. Im Verlauf der Entwicklung war ein Hauptziel, das Modell mit der höchsten Genauigkeit und Präzision zu identifizieren. Auf Basis dieser Forschung erwiesen sich die Algorithmen Extreme Gradient Boosting (XGB) und Random Forest (RF) als die vielversprechendsten und erreichten Genauigkeitsraten von 80 % bis 96 %. Nach der Erkennung gesundheitlicher Komplikationen wurde weiter geforscht, ob KI auch zur Bestimmung möglicher Medikamente auf Basis der erkannten Komplikationen und der Vitalzeichen der Patienten eingesetzt werden kann. Auch hier waren die erzielten Ergebnisse beeindruckend, mit einer Genauigkeit von fast 80 %. Die KI-Modelle wurden zusätzlich getestet, indem sie in verschiedenen Beschleunigern wie ARM-Prozessoren, FPGAs und Mikrocontrollern eingesetzt wurden, um ihre Leistung anhand der Inferenzzeit zu evaluieren. Der Schwerpunkt dieser Forschung liegt insgesamt darauf, die Herausforderungen von Rettungseinsätzen in Echtzeit zu bewältigen, indem Rettungssituationen erkannt und die Qualität der Versorgung sowie die Effizienz des Rettungspersonals verbessert werden.

To Ammu

Acknowledgments

This thesis was created during my employment at the Chair for Embedded Systems at the University of Siegen in the context of the research project KIRETT funded by the BMBF.

My deepest gratitude to my supervisor, Prof. Dr. Roman Obermaisser, for his invaluable guidance, encouragement, and expertise throughout the course of my doctoral research. We had many interesting discussions on how to improve the research and explore new opportunities in a similar direction. His insightful advice and unwavering support have been instrumental in shaping this thesis.

I would like to thank Prof. Dr. Alberto Garcia, my master thesis supervisor at University of Bremen, who led me to the path of artificial intelligence and embedded systems. His motivation and guidance in learning machine learning had a great impact on my Ph.D. research.

I would also like to extend my sincere thanks to my colleagues, especially Amran Hossain, Micheal Ezekiel, Mario Qosa, Veit Wiese, Simon Meckel, Sayeri Mukherjee, Steffan Otterbach, and Manuela Popp, who supported me in many different ways at the chair, which served as the foundation for this thesis. Their teamwork, dedication, and shared knowledge made this research endeavor both productive and rewarding.

Throughout the duration of the research, my research has been supported by the colleagues from Knowledge-base Department at University of Siegen, KIRETT project coordinator CRS Medical GmbH (Aßlar, Germany), partner organization mbeder GmbH (Siegen, Germany) with the associative partners of the project: Kreis Siegen-Wittgenstein, City of Siegen, the German Red Cross Siegen (DRK) and the Jung-Stilling-Hospital in Siegen. I would like to acknowledge the invaluable support of all partners, whose cooperation and provision of critical data on rescue operations and patient management significantly enriched the scope and relevance of my research.

I am deeply grateful to my wife and two daughters for their constant love, patience, and sacrifices. Their encouragement and understanding have been my source of strength and motivation throughout this challenging journey.

Lastly, the person who was behind this whole journey was my late mother. With-

out her motivation and guidance, it was not possible for me to step on this path. I am thankful to my parents for all their inspiration and support.

To all those who contributed to this work, directly or indirectly, your support has been greatly appreciated and will always be remembered with gratitude.

Contents

1	Introduction	1
1.1	Motivation and Problem Statement	1
1.1.1	Research Contributions	2
1.1.2	Detection of Health Complications	3
1.1.3	Medication Requirements	5
1.2	Research Objectives	6
1.3	Thesis Structure	7
1.4	Publications	8
1.4.1	Journal	8
1.4.2	Conference Paper	8
2	Related Research	9
2.1	Research Requirements	9
2.2	Related Work	12
2.2.1	Assessment of Health Complication with ML	12
2.2.2	Machine learning in Treatment Recommendation	14
2.2.3	Embedded System in Healthcare	14
2.2.4	Natural Language Processing in Healthcare	16
2.2.5	Digital Healthcare Software for Rescue Events	17
2.3	Research Gap	18
3	Research Concepts	20
3.0.1	Natural Language Processing Algorithms	20
3.1	Machine Learning Models	22
3.1.1	Classical Machine Learning Algorithms	22
3.1.2	Deep Learning Algorithms	32
3.2	Real Time Applicability	34
3.2.1	Rescue Care Software	34
3.2.2	Embedded Accelerators	35
4	Rescue System Model	38
4.1	Organization of Rescue Services and Stakeholders	39
4.2	AI Integration in the Existing Rescue System	42
4.3	Smart Wearable	43

5	Rescue Data Analysis and Organization	47
5.1	Raw Rescue Data	48
5.1.1	Data on Health Vital Information	49
5.1.2	Text Data on Rescuer's Impression	50
5.1.3	Data on Diagnostic Information	51
5.2	Data Organization	52
5.2.1	Data Merging	53
5.2.2	Data Pre-processing	53
5.2.3	Data Dictionary	55
5.3	Attributes on Health Complications	57
5.3.1	Heart Complication	58
5.3.2	Respiratory Complication	58
5.3.3	Neurological Complication	59
5.3.4	Psychiatric Complication	59
5.3.5	Metabolic Complication	60
5.3.6	Abdominal Complication	60
5.4	Database Management System	60
5.4.1	Software Development	61
5.5	Data on Treatment Path in Rescue Events	63
5.5.1	Medications	64
6	Feature Engineering	66
6.1	Feature Identification	67
6.2	Feature Generation from Text Data	67
6.3	Feature Selection and Elimination	70
6.4	Outlier Detection and Removal	72
7	Machine Learning in Rescue Event	74
7.1	Advantages of ML in Rescue Situations	74
7.2	Training Data	76
7.2.1	Training Data for Situation Recognition Model	76
7.2.2	Training Data for Medication Recommendation Model	77
7.3	Tuning ML Models	79
7.3.1	Grid Search	79
7.3.2	Random Search	80
7.3.3	Neural Network Tuning	81
7.4	ML Model Development	83
7.4.1	Situation Recognition Models	84
7.4.2	Medication Recommendation Models	86
8	Embedded Software in Rescue Situations	89
8.1	Rescue Care Assistance Software	89
8.1.1	Software Requirement Analysis	90
8.2	Software Design and Implementation	91

8.3	Hardware Acceleration	93
8.3.1	Acceleration in FPGA	93
8.3.2	Acceleration in Microcontroller	98
8.3.3	Acceleration in Raspberry Pi	100
9	Results and Discussion	103
9.1	Performance Matrix	103
9.2	Model Performance for Situation Recognition	105
9.2.1	Cardiovascular Complication	105
9.2.2	Respiratory Complication	108
9.2.3	Neurological Complication	111
9.2.4	Psychiatric Complication	114
9.2.5	Metabolic Complication	118
9.2.6	Abdominal Complication	121
9.3	Model Performance for Medication Recommendation	125
9.3.1	Cardiovascular Treatment Recommendation	125
9.3.2	Respiratory Treatment Recommendation	129
9.3.3	Neurological Treatment Recommendation	131
9.3.4	Psychiatric Treatment Recommendation	135
9.4	Performance of Software	138
9.4.1	Use-Case 1	139
9.4.2	Use-Case 2	139
9.4.3	Use-Case 3	139
9.5	Performance in Hardware Platform	141
10	Conclusions and Future Work	145
10.1	Summary of the Thesis	145
10.2	Future Enhancement	147
A	Appendix	152
A.1	Appendix A: List of Hyperparameters Relevant to Health Complications	152
A.1.1	Situation Recognition Models	152
A.1.2	Treatment Recommendation Models	155
A.2	Appendix B: Cross-validation Results	158
A.2.1	Situation Recognition Models	158
A.2.2	Treatment Recommendation Model	164
A.3	Appendix C: Text Features for Individual Complication	168

List of Figures

3.1	Example of NLP-based Preprocessing Steps	21
3.2	Support Vector Machine	24
3.3	Random Forest Algorithm	26
3.4	XGB Algorithm	27
3.5	Logistic Regression Algorithm	28
3.6	K-nearest Neighbors Algorithm	30
3.7	Working Methodology of Adaptive Boosting Algorithm	31
3.8	Artificial Neural Network Algorithm	33
4.1	Principal Stakeholders in Rescue System	40
4.2	Integration of AI in Rescue Activity [35, 132]	44
4.3	Proposed Methodology on Data Analysis and ML Model Development.	45
4.4	Rescue Patient Management with ML Accelerated Smart Wearable Having Interactive Display	46
5.1	Number of rescue patients with different types of health complications as per the 10 years record of the rescue station, Siegen–Wittgenstein. Many rescue events were found without proper diagnostic information, and were thereby uncategorized.	52
5.2	GUI of Rescue Data Dictionary	56
5.3	Health Vital Information in Data Dictionary	57
5.4	Software Architecture of Database Management Software Showing Fea- tures Relevant to Central Nervous System (CNS) or Neurological Disease	62
5.5	GUI Displaying Major Health Complications	63
5.6	GUI Displaying Patient’s Record	63
6.1	Outlying Respiratory Rate Values in Cardiovascular Complication . .	72
7.1	Basic Procedures Inside Machine Learning Based Software	84
7.2	Software Architecture- Situation Recognition Model	85
7.3	Software Architecture - Medication Recommendation Model	88
8.1	Rescue Health Management SW Architecture to Support Patient in Emergency Situations	92
8.2	PYNQ Z2 FPGA	96

8.3	A Block Diagram Consist of Zynq 7000 Processor and Custom PL Core to Accelerate Decision Tree-based Machine Learning Model	98
8.4	BLE Sense 33 Microcontroller with Block Diagram [64]	100
8.5	Raspberry Pi 5B [170]	101
9.1	Comparative Analysis of Performance - Cardiovascular Complication Detection Algorithms	107
9.2	ROC-AUC Curve to Depict Performance of Cardiovascular Complication Detection Algorithms	107
9.3	Feature Importance Score Used by Random Forest Algorithm for Cardiovascular Complication	108
9.4	Comparative Analysis of Performance - Respiratory Complication Detection Algorithms	110
9.5	ROC-AUC Curve to Depict Performance of Respiratory Complication Detection Algorithms	111
9.6	Feature Importance Score Used by XGBoost Algorithm for Respiratory Complication	112
9.7	Comparative Analysis of Performance - Neurological Complication Detection Algorithms	113
9.8	ROC-AUC Curve to Depict Performance of Neurological Complication Detection Algorithms	114
9.9	Feature Importance Score Used by Random Forest Algorithm for Neurological Complication	115
9.10	Comparative Analysis of Performance - Psychiatric Complication Detection Algorithms	116
9.11	ROC-AUC Curve to Depict Performance of Psychiatric Complication Detection Algorithms	117
9.12	Feature Importance Score Used by Random Forest Algorithm for Psychiatric Complication	118
9.13	Comparative Analysis of Performance - Metabolic Complication Detection Algorithms	120
9.14	ROC-AUC Curve to Depict Performance of Metabolic Complication Detection Algorithms	120
9.15	Feature Importance Score Used by Random Forest Algorithm for Metabolic Complication	121
9.16	Comparative Analysis of Performance - Abdominal Complication Detection Algorithms	123
9.17	ROC-AUC Curve to Depict Performance of Abdominal Complication Detection Algorithms	124
9.18	Feature Importance Score Used by Random Forest Algorithm for Abdominal Complication	124
9.19	Comparative Analysis of Performance Metrics for Cardiovascular Complication	127
9.20	Confusion Matrix for Individual Labels with RF Algorithm	128

9.21	Confusion Matrix for Individual Labels with LR Algorithm	128
9.22	Comparative Analysis of Performance Metrics for Respiratory Complication	130
9.23	Confusion Matrix for Individual Labels with XGB Algorithm	131
9.24	Confusion Matrix for Individual Labels with LR Algorithm	132
9.25	Comparative Analysis of Performance Metrics for Neurological Complication	133
9.26	Confusion Matrix for Individual Labels with LR Algorithm	134
9.27	Confusion Matrix for Individual Labels with XGB Algorithm	134
9.28	Comparative Analysis of Performance Metrics for Psychiatric Complication	136
9.29	Confusion Matrix for Individual Labels with LR Algorithm	137
9.30	Confusion Matrix for Individual Labels with RF Algorithm	137

List of Tables

3.1	Traditional Feature Extraction Algorithms of NLP	22
4.1	Type of Emergencies	39
5.1	Categorization of Rescue Information	48
5.2	Common Health Vitals Recorded in Rescue Event	50
5.3	Text Information Recorded by Rescuers in Rescue Cases	51
5.4	Diseases Under Neurological Complication.	59
5.5	Common Medications used in Rescue Situations	65
7.1	Distribution of Data for Patients and Non-Patients for Situation Recognition Models	76
7.2	Number of Patients Considered to be Used for Treatment Recommendation Models	77
9.1	Performance Metrics of ML Models- Cardiovascular Complication. All Values are in Percentage	106
9.2	Performance Metrics of ML Models- Respiratory Complication. All Values are in Percentage	109
9.3	Performance Metrics of ML Models- Neurological Complication. All Values are in Percentage	113
9.4	Performance Metrics of ML Models- Psychiatric Complication. All Values are in Percentage	116
9.5	Performance Metrics of ML Models- Metabolic Complication. All Values are in Percentage	119
9.6	Performance Metrics of ML Models- Abdominal Complication. All Values are in Percentage	122
9.7	Performance of Medicine Recommendation Model - Cardiovascular Complications. All Values are in Percentage	127
9.8	Performance of Medicine Recommendation Model - Respiratory Complications. All Values are in Percentage	130
9.9	Performance of Medicine Recommendation Model - Neurological Complications. All Values are in Percentage	133
9.10	Performance of Medicine Recommendation Model - Psychiatric Complications. All Values are in Percentage	136

9.11	PATIENT DATA FROM USE CASES	138
9.12	Health Complication Probability Prediction For Use Case - 1	139
9.13	Health Complication Probability Prediction For Use Case - 2	140
9.14	Health Complication Probability Prediction For Use Case - 3	140
9.15	Health Complication Probability After Modifying Health Vital Data of Use Case - 3	141
9.16	Average Inference Time (milliseconds) Across Accelerators for Differ- ent ML Models and Health Complications	142
A.1	Hyperparameter Grid for XGBoost	152
A.2	Tuned Hyperparameters of Health Complications for XGBoost Algorithm	152
A.3	Hyperparameter Grid for Random Forest	153
A.4	Tuned Hyperparameters of Health Complications for Random Forest Algorithm	153
A.5	Hyperparameter Grid for Support Vector Machine	153
A.6	Tuned Hyperparameters of Health Complications for Support Vector Machine Algorithm	153
A.7	Hyperparameter Grid for K-Nearest Neighbors	153
A.8	Tuned Hyperparameters of Health Complications for K-Nearest Neigh- bors (KNN) Algorithm	153
A.9	Hyperparameter Grid for Logistic Regression	154
A.10	Tuned Hyperparameters of Health Complications for Logistic Regres- sion Algorithm	154
A.11	Hyperparameter Grid for Naive Bayes	154
A.12	Tuned Hyperparameters of Health Complications for Naive Bayes Al- gorithm	154
A.13	Hyperparameter Grid for XGB Algorithm	155
A.14	Tuned Hyperparameters of Treatment Recommendation Models with XGB Algorithm	155
A.15	Hyperparameter Grid for Random Forest Algorithm	155
A.16	Tuned Hyperparameters of Treatment Recommendation Models with Random Forest Algorithm	155
A.17	Hyperparameter Grid for Logistic Regression Algorithm	155
A.18	Tuned Hyperparameters of Treatment Recommendation Models with Logistic Regression Algorithm	156
A.19	Hyperparameter Grid for Multi-Layer Perceptron Algorithm	156
A.20	Tuned Hyperparameters of Treatment Recommendation Models with Multi-Layer Perceptron Algorithm	156
A.21	Hyperparameter Grid for AdaBoost Algorithm	156
A.22	Tuned Hyperparameters of Treatment Recommendation Models with AdaBoost Algorithm	156
A.23	Hyperparameter Grid for K-Nearest Neighbors Algorithm	156
A.24	Tuned Hyperparameters of Treatment Recommendation Models with K-Nearest Neighbors Algorithm	157

A.25 Cross Validation Result of ML Models for Cardiovascular Complication	158
A.26 Cross Validation Result of ML Models for Respiratory Complication .	159
A.27 Cross Validation Result of ML Models for Neurological Complication	160
A.28 Cross Validation Results for Psychiatric Complication	161
A.29 Cross Validation Results for Metabolic Complication	162
A.30 Cross Validation Results for Abdominal Complication	163
A.31 Cardio Medication Recommendation Model Cross Validation Results	164
A.32 Respiratory Medication Recommendation Model Cross Validation Re- sults	165
A.33 Neurological Medication Recommendation Model Cross Validation Re- sults	166
A.34 Psychiatric Medication Recommendation Model Cross Validation Results	167
A.35 Text Features For Abdominal Complication	168
A.36 Text Features For Psychiatry Complication	168
A.37 Text Features For Neurology Complication	169
A.38 Text Features For Metabolic Complication	169
A.39 Text Features For Respiratory Complication	170

1

Introduction

1.1 Motivation and Problem Statement

Rescue stations around the world receive millions of emergency rescue calls each year, most of which are due to health complications. Based on statistics provided by the European Commission in their 2022 annual report [34], an estimated 141 million calls are received yearly only in Europe, where callers are often required for emergency medical treatment. The rescue data used in this thesis work consist of more than 273,000 rescue cases of approximately 10 years only for the rescue station in the medium-sized city Siegen-Wittgenstein with a population of approximately 277,000. These statistics clearly indicate the high frequency and necessity of rescue services and raise the issue of how effective and efficient rescue services are in a health crisis and whether there is a need to improve quality management. During an emergency health situation, the rescue service can be divided into two major steps: 1) assess the health situation and 2) find out the treatment requirements by administering medication, resuscitation, or transfer to the nearby hospital. The biggest challenge here is the rescue time because in a medical emergency every second counts. According to recent statistics from healthcare organizations and emergency response agencies, a high percentage of patient mortality is attributed to delays in the identification and early treatment of patient complications. In addition to time limitation, the complexity of disease recognition increases due to their similarity in symptoms and obstacles faced during situation-based data collection. This proportionately heightens the difficulty for paramedics in making clinical decisions. An additional limitation lies in resource constraints, as unlike hospitals, rescue vehicles lack modern medical equipment to assess the underlying health complications patients may have. These aspects now lead to the important question of whether state-of-the-art Artificial Intelligence algorithms can be used to improve these steps in rescue operations.

Artificial intelligence techniques ranging from machine learning to deep learning are widely used in health care for disease diagnosis, treatment path recommendation, and patient health management. Machine learning (ML) algorithms, e.g. support vector machine (SVM), K-nearest neighbor (KNN), random forest (RF), logistic regression (LR), etc. are widely used in the prognosis of diseases like skin cancer, diabetes, epilepsy, etc. The algorithms are generally trained with data extracted from patients' signs and symptoms or medical tests, including images. They are expected to provide accurate evaluations and diagnoses of diseases when new patient data is provided. The purpose of using artificial intelligence in the modern health sector is to reduce the workload of physicians, minimize errors and diagnostic times, and improve performance in the prediction and detection of various diseases. Although the detection accuracy achieved through the algorithms is quite significant, it should be mentioned that the experience of physicians in diagnosing such complications is undeniable. In this thesis, detailed research has been performed that focuses on how artificial intelligence can be used in rescue cases to detect major health complications and to suggest treatment protocols like medications. The use of AI in rescue cases is a novel approach and is found through research to be quite resourceful as it can expedite disease diagnoses and treatment with high precision.

Several challenges were faced during the AI implementation that motivated the advanced data analytics performed during the doctoral period. A core challenge was the different modalities of input information. This includes linguistic input from the rescue call center and paramedics, as well as time series of vital signs from medical devices. Therefore, existing AI frameworks and architectural approaches cannot be applied directly to the aforementioned problem without processing the information. Another big challenge was the ubiquitous, incomplete, and informal information about previous missions in the databases. Therefore, appropriate software tools were required to be developed that standardize and formalize the data sets. Fully manual processing is infeasible due to the large number of records. Likewise, fully automated processing is unrealistic. Therefore, interactive tools for semiautomatic processing of the data sets have to be developed.

1.1.1 Research Contributions

To make AI work efficiently, the most important step during algorithm development was model training with pre-processed data. That is why a significant contribution was made to create a dataset with the right features and attributes considering the state of rescue workers during emergency operations. Data science algorithms were used as preprocessing steps to convert raw data into a usable format, such as data mining, filtering, and advanced natural language processing techniques. As the focus of AI is two folds in the research: 1) detecting health complications and 2) finding the right treatment, two different data sets were prepared. Each of those datasets had a combination of different health features and required various data preprocessing steps discussed in later chapters of the dissertation. When defining the attributes of the dataset, rescue scenarios were considered, as not all information can be collected flexibly under emergency conditions due to the time constraint. An important

contribution was provided by exploring classical machine learning and deep learning algorithms with the pre-processed data to find the most efficient algorithm in terms of different evaluation matrices. Another goal of this research was to develop a real-time embedded system with hardware including an FPGA, Raspberry Pi, and microcontroller using acceleration platforms and to check the efficiency in terms of inference time if deployed in practical scenarios.

1.1.2 Detection of Health Complications

Early diagnosis of health complications in rescue situations is vital to initiate proper treatment and improve patient outcomes. Numerous medical data sources are used to diagnose diseases using artificial intelligence techniques, such as ultrasound, magnetic resonance imaging, mammography, genomics, computed tomography scan, etc. However, due to resource constraints, such diagnostic processes are not viable in rescue situations; instead, a combination of thorough assessment, monitoring, and common health symptoms is generally considered for the health assessment. The rescue personnel primarily perform the assessment of a patient's health from the rescue call and then by consulting with the family members or nearby persons of the rescue patients. Sometimes, for recurring patients, health records can be available, but tracking the real-time health situation is always of the highest importance. For the assessment and treatment of critically ill patients, rescue stations in Germany consider the CABCDE approach to determine the basic health status of the patient. Generally, these approaches are widely accepted by experts in rescue scenarios and likely improve results by helping rescue personnel focus on the severe and life-threatening health complexities. Portable diagnostic tools are carried, for example, handheld devices for measuring vital signs, portable ECG monitors, etc. to support the identification of potential health problems. However, there are certain limitations associated with these approaches when trying to diagnose the actual patient's health scenario as listed below:

- Patients may not be able to communicate due to impaired consciousness. It causes a lack of access to the patient's medical history and prior diagnostic information
- Patients' representation of their symptoms and health complaints can vary widely and lead to false diagnosis
- Emergency patients may show atypical symptoms due to the acute nature of their illness or injury which poses a risk of delayed or wrong diagnosis
- Due to high stress and time shortage, distortion of the cognitive abilities of first aid providers is quite common, potentially affecting their diagnostic accuracy
- Some complications are difficult to prognose only with health vitals and requires further analysis of symptoms, leading to high time consumption

Taking into account these limitations, to improve patient care in rescue situations, artificial intelligence algorithms are exploited in this research due to their time-efficient predictive analysis capability. Currently, there is much research on how AI can facilitate state-of-the-art critical issues in sectors e.g. automotive, healthcare, network design, energy consumption, and many others. In a recent study, the Organization for Economic Co-operation and Development [OECD] [129] found how AI can create a strong impact in the healthcare sector by improving diagnostic accuracy while working with a vast amount of health data that are currently underutilized. But at the same time, there are also some visible threats of working with AI as the algorithms are yet to establish enough credibility to be used in real-life situations. The principal factor in creating an efficient AI algorithm is to have a good quality of data source that is often missing, resulting in poor outcomes. Data privacy is another concern when working with real patient data. To overcome these challenges, many countries are establishing AI oversight bodies and mechanisms to monitor the implementation and evolution of AI solutions in healthcare. An important step should be the inclusion of health professionals in the creation of AI solutions. Finally, a global effort is required to address the challenges and risks associated with the implementation of AI in healthcare systems, emphasizing the importance of governance, transparency, accountability, and ethical considerations in the use of AI for the benefit of patients and healthcare systems.

Several machine learning and deep learning algorithms were exploited during this research to identify how AI algorithms can contribute to the predictive analysis of health complications. Data organization and processing are the foundational step before developing machine learning algorithms. As the historical rescue data received was complex and contained many ambiguous/null information, different data filtering processes, e.g. outlier detection, data interpolation, normalization, standardization, etc. were considered. These processes ensure that the data are clean, relevant, and structured in a way that facilitates effective model training and prediction. The next major step was feature engineering, as many of the features were not providing meaningful insight and some features were missing values. Unfortunately, for complications such as neurological or psychiatric complications, the information available to the existing data set was not adequate either. So natural language processing techniques were utilized to parse through textual information and extract new features by extrapolation and interpolation.

After individual data sets were prepared for the major health complications, several classical machine learning and deep learning algorithms were trained for each complication. The goal was to achieve the highest possible performance in detecting underlying health complications. So, every model developed went through several feature refining techniques and cross-validation to find the optimized hyperparameters. Finally, six machine learning models were developed that can predict based on medical history, health signs, and rescuer's impression whether the patient is suffering from below health complications:

- Cardiovascular

- Respiratory
- Psychiatric
- Neurological
- Abdominal
- Metabolic

The predictive model for each health complication had a different level of accuracy between 80% and 97% and was tested with dynamic changes in the data through software. The research found that the Random Forest and Extreme Gradient Boosting algorithms performed better than other algorithms.

An important aspect was to understand how the models perform when there are rapid changes in health vital data during rescue events. When implemented in hardware such as Xilinx Pynq, Arduino Nano 33 BLE, and Raspberry Pi 5 the prediction accuracy was identical for all models and showed very low inference times ranging in milliseconds, confirming the eligibility for using such models in real rescue events. In addition, a Python-based rescue care assistance software was designed that has the potential to improve the efficiency and effectiveness of rescue situational awareness by rapidly assessing the health status of emergency patients using AI-driven decision support systems. The novelty in this software approach is its user-centered design principles to ensure that its solutions are specifically tailored to meet the unique requirements of emergency responders. The software utilizes pre-existing machine learning components designed to detect health issues and provides probability percentage of the major health complications when patient data is tested.

1.1.3 Medication Requirements

. The main purpose of the accelerated diagnosis of the rescue patient was to decide on an optimized treatment path in the quickest possible time. This may include providing crucial first aid, stabilizing people with life-threatening injuries, or administering the necessary medications. In many emergency scenarios, such as cardiac arrest, severe trauma, or respiratory failure, delay in treatment can be fatal. Countless lives are lost annually due to delays or errors in providing the correct medications at crucial times. According to recent data from healthcare organizations and emergency response agencies, a substantial percentage of deaths in emergency situations can be directly attributed to delays or errors in prescription medication [156]. That is why the ability to infer quick treatment paths in emergency rescue operations is a primary requirement to optimize rescue care. In this thesis, research has been conducted on how AI algorithms can predict correct medications required for an emergency patient based on health symptoms and diagnosed complication.

Artificial intelligence has gained significant traction in the field of treatment recommendation due to their remarkable ability to model complex, non-linear relationships within medical datasets. The rescue data set used in this investigation consists of

health vitals, patient complaints, pre-illness information, rescuer’s first impression, and administered medications. The medication information was considered as the basis of the treatment path and a detailed data analysis was performed to identify the decision made about medications in past events. As AI algorithms have the ability to grasp incredibly complex patterns and relationships within large datasets, the research showed significant precision in predicting the right medication for the main health complications normally visible in rescue situations. According to the rescue database, detailed medication data were available for only four health complications which are cardiovascular, respiratory, neurological, and psychiatric. Thus, the AI-based treatment plan was specifically tailored for each of these complications.

As multiple medications are administered to rescue patients, machine learning algorithms supporting multiclass i.e. Random Forest, Artificial Neural Network, Logistic Regression, Adaboost, K-nearest Neighbors, and XGBoost were considered for the medication prediction model. Like situation recognition, individual medication requirement models were developed for each major health complication. An important attribute required for these models was the identified health status of the rescue patient, which was extracted from the linked AI-based individual situation recognition model. The results of the model evaluation indicated that the trained ANN and Random Forest classifiers demonstrated commendable metrics in accuracy, sensitivity, specificity, precision, and F1 scores. For cardiovascular complications, the Random Forest algorithm recorded an accuracy of 87.3%. For respiratory cases, the XGB algorithm achieved an accuracy of 82.8%. Concerning neurological complications, both the XGB and the RF classifier achieved accuracy greater than 83%, while the MLP classifier had specificity greater than 90%. In terms of psychiatric complications, the RF algorithm had registered an impressive accuracy of 89% and a specificity of 89.1%, ensuring its effectiveness in medication recommendation.

1.2 Research Objectives

The main focus of the KIRETT project is to provide comprehensive data integration, improve primary care quality and efficiency, and improve traceability and quality management. As part of the research project, the doctoral thesis aims to achieve the following objectives:

Methods for situational recognition of emergency patients: Development of reliable, dynamic, and high-performance artificial intelligence models by evaluating the information from the call center, medical devices, and the input of the paramedic that can detect the condition of rescue patients in real time, when this knowledge is useful for triggering associated care actions by emergency paramedics. The development will be followed by a performance evaluation with validation datasets to verify the accuracy, precision, and robustness of different machine learning models.

Methods for recommending medications as treatment path: Another machine learning model to be developed to infer first-aid medical action instructions for emergency paramedics based on the semi-automatic interpretation of the current condition of the emergency patient in rescue situations.

Identifying usability of the AI models in the real world: Rescuers working in actual rescue situations need to have a usable, robust, and user-friendly device that can dynamically display the patient’s situation and propose the correct treatment. Deploying AI models on appropriate hardware platforms such as FPGAs, microcontrollers, or Raspberry Pi will verify feasibility and pave the way for future hardware advancements.

The overarching goal is to leverage advanced data analysis techniques and AI algorithms to overcome challenges faced by rescue workers, such as time constraints and environmental stressors, by providing timely and accurate diagnostic support to rescue personnel. This will ultimately improve the quality of care and efficiency of rescue operations, potentially saving more lives in emergency situations.

1.3 Thesis Structure

After highlighting the motivation, research challenges and objective in this chapter, below is an overview of the other chapters contained in this dissertation.

Chapter 2 reviews the existing literature on AI-based disease prognosis and treatment recommendation models. It also discusses the past work of embedded systems in healthcare.

Chapter 3 delves into the core concepts that underpin the research done in this doctoral thesis. It will provide an overview of the collected rescue data and also discusses AI algorithms such as machine learning and natural language processing used for situation recognition. An important part is how AI models are deployed in different hardware platforms.

Chapter 4 presents the current rescue system architecture and its shareholder. It will also provide an overview of how integrating AI can modify this architecture and what happens when it is deployed in rescue events through a wearable.

Chapter 5 covers the data prospects starting from the initial analysis of the raw rescue data and how these data were filtered, pre-processed, and organized with database management software.

Chapter 6 focuses on feature engineering techniques, used to identify and extract new features. In addition, it sheds light on outlier detection and removal, null data management, and recursive feature elimination.

Chapter 7 covers the complete pipeline for the development of machine learning algorithms in disease prognosis and treatment recommendation, including the creation of training data, the hyperparameter search, and the tuning process.

Chapter 8 discusses the real-time applicability of algorithms developed through rescue care assistance software and the deployment of them on different hardware platforms. The results of different AI models and the findings of hardware deployment are explored in Chapter 9. It also provides a critical analysis of the research findings, discussing their implications and limitations.

Finally, in Chapter 10, the conclusions drawn from the research are summarized and potential areas for future work are identified.

1.4 Publications

The publications list during the research period is as follows:

1.4.1 Journal

- Ahammed, A. S., Ezekiel, A. M., & Obermaisser, R. (2023). Time-efficient identification procedure for neurological complications of rescue patients in an emergency scenario using hardware-accelerated artificial intelligence models. *Algorithm*, 16, AI Algorithms for Healthcare. MDPI.
- Oleh, U., Obermaisser, R., & Ahammed, A. S. (2024). A review of recent techniques for human activity recognition: Multimodality, reinforcement learning, and language models. *Algorithms*, 17(10), 434. MDPI.

1.4.2 Conference Paper

- Zenkert, J., Weber, C., Nadeem, M., Bender, L., Fathi, M., Ahammed, A. S., Ezekiel, A. M., Obermaisser, R., & Bradford, M. (2022). Kirett-a wearable device to support rescue operations using artificial intelligence to improve first aid. 2022 IEEE International Smart Cities Conference (ISC2). IEEE.
- Ahammed, A. S., Donthireddy, S. R., & Obermaisser, R. (2022). Detection of respiratory emergency situation of rescue patients with machine learning algorithms. IECON 2022–48th Annual Conference of the IEEE Industrial Electronics Society. IEEE.
- Ahammed, A. S., Ezekiel, M., & Obermaisser, R. (2022). A novel analysis of performance and inference time of machine learning models to detect cardiovascular emergency situations of rescue patients. 2022 International Conference on Artificial Intelligence of Things (ICAIoT). IEEE.
- A. S. Ahammed and R. Obermaisser, "Smart Health Software to Support Rescue Personnel in Emergency Situations," 2024 International Conference on Smart Systems and Technologies (SST), Osijek, Croatia, 2024, pp. 165-170, doi: 10.1109/SST61991.2024.10755467.
- Hossain, M. S. A., Ahammed, A. S., Biswas, D. P., & Obermaisser, R. (2024). Impact analysis of data drift towards the development of safety-critical automotive systems. 2024 International Symposium ELMAR, 325–330. IEEE.
- Ahammed, A. S., Hossain, M. S. A., & Obermaisser, R. (2024). A computer vision approach for autonomous cars to drive safely at construction zones. arXiv preprint arXiv:2409.15809.

2

Related Research

This chapter presents an extensive overview of the research questions addressed in this dissertation alongside prior studies conducted within a similar domain. The literature review focuses on previous works in the field of artificial intelligence-based predictive models to detect the underlying health complications, preferably in a rescue situation and the need for medications. The objective is to explore the current research within the existing body of knowledge and to present developments and findings that have shaped the approach to using AI in medical emergencies. In addition, a discussion is conducted on the development of relevant AI-based embedded systems that can assist rescue personnel in real time. After a comprehensive review of the literature, it was found that despite the current success of machine learning in detection models for many scientific areas including health complications, the use of machine learning in recognizing rescue situations is yet to be well explored. The field examined in this thesis, specifically rescue operations, is not previously addressed in this branch of health complication research according to the investigation.

2.1 Research Requirements

The research requirements were identified based on the challenges at the beginning of the research project KIRETT funded by Federal Ministry of Education and Research, Germany. Machine learning is an exciting concept currently used by many biomedical research groups to predict the progression of diseases, anticipate patient outcomes, disease localization, early detection, and prevention. Recent trends and approaches in machine learning-based disease diagnosis (MLBDD) consider factors such as the algorithm used, disease types, data, practical application in the real world and evaluation criteria to assess usability [9]. Some of the key areas of focus for MLBDD include: cancer prognosis and treatment [57], prediction of cardiovascular disease risk

[86], prediction of the progression of neurodegenerative diseases such as Parkinson or Alzheimer, diabetes management, chronic disease monitoring and identification of postoperative results. Machine learning algorithms were also found to be effective in the prognosis of recent COVID-19 pandemics [44]. Contemporary research continues to explore the use of machine learning in the biomedical field while putting great emphasis on increasing the precision, interpretability, and usability of these models by clinicians. The current trend of machine learning in health care lies in the advancement of predictive modeling techniques, the enhancement of data interoperability and sharing, the integration of multimodal exchange, the incorporation of domain knowledge and expert feedback into models, the development of robust validation frameworks and the translation of research findings into clinical practice. However, it is important to note that while machine learning offers promising solutions, it also comes with adjacent challenges such as data quality and interoperability issues, model interpretability and transparency, ethical and privacy concerns, regulatory compliance, and integration into existing healthcare workflows, and ensuring that models can generalize to different populations [119]. In particular, data privacy is a significant concern now a days when dealing with patient data concerning ethical health research. It is crucial to protect the well-being and rights of patients involved in research, as this is fundamental to conducting ethical studies. The main reason for upholding personal privacy is to defend the interests of the individuals involved [61]. Considering these aspects and challenges, to the objectives of the thesis, below research requirements are identified:

- **Req 1 Data Organization:** Developing the pipeline for data collection and organization is the primary requirement before establishing an AI-based model. Data for training algorithms is considered an indispensable element in machine learning, serving as the essential resource that drives the entire lifecycle of ML projects, from training and validation to deployment and continuous improvement. Well-organized data can always improve the performance of machine learning models and potentially lead to improved model accuracy. The requirement arises at this point because the data obtained from the rescue station contained numerous outliers and ambiguities. Having such flaws in the rescue data is not unexpected for rescue records, as the primary task of the rescuers involved in emergency situation is to first provide necessary care for patients rather than recording information. So to fulfill the requirement, it was required to filter the raw data to remove all those ambiguity and then new information was interpolated using advanced data pre-processing algorithm.
- **Req 2 Feature Identification:** Every health complication comes with different signs and symptoms which generally is recorded in clinical evaluation when a patient is taken care of by medical professionals or clinical organizations. But in rescue events, the situation is completely different and only the basic health information of a patient is recorded. An essential need thereby emerges to derive features from the recorded data that are applicable and can describe a patient's

underlying health condition. Also, for individual complication, separate feature sets need to be extracted and processed using state-of-the-art data preprocessing tool kits like NLP. The list of features required for each complication is identified through discussion with medical professionals and past research.

- **Req 3 Develop of Machine Learning Models:** The principal requirement in this research is to develop machine learning models that can function in rescue situations for identifying the major health complications. The challenge here is to use the existing data available during a rescue event which is generally not adequate to make a conclusive diagnostic decision because health complications are complex to diagnose only with health vitals. If requirement 1 and 2 are satisfied by creating filtered datasets with necessary features relevant to health complications, the next challenge will be to create a model training protocol that ensures high accuracy in real world applications. The training protocol encompasses the requirements like balanced dataset, feature validation, hyperparameter selection, cross validation and testing, and performance optimization. As the models are intended for using in the healthcare domain, It is necessary to choose appropriate machine learning models that require minimal computational resources, offer rapid inference, and are suitable to work with health data.
- **Req 4 Treatment Recommendation:** The next requirement after detecting health situation was to develop a module for recommending treatment path for emergency patients based on health diagnosis. As it is found in the rescue record, often the rescue doctors prescribe multiple medications during patient care because the patient might have been suffering from more than one complications. Also, the purpose of administering medication varies based on rescue situation. So to recommend the optimized treatment path, a critical requirement is raised to develop a multilabel machine learning algorithm that can consider all the above-mentioned rescue contexts and suggest medications as a part of treatment. However, the performance requirements for treatment recommendation model is similar to the situation detection model mentioned in Req 2.
- **Req 5 Hardware Acceleration:** The applicability of the rescue support system in real world scenarios is an important aspect to consider as the rescue operations are mobile and need quick support thorough tangible and portable devices. Therefore, the requirement of developing an embedded prototype comes forward which is required to have the same accuracy and precision of the ML models when developed as a software. The embedded prototype should be developed with different hardware programmable architectures and tested for their suitability within a testing framework to find out the best suitable ones in real world applications. Meeting this requirement will pave the way for the future creation of advanced and portable devices such as smart wearables.

- **Req 6 Rescue care assistance software:** The next requirement involves creating a rescue care assistance software application that integrates AI algorithms and incorporates natural language processing (NLP) technology to provide text-based recommendations for rescue personnel, aimed at enhancing patient care. The software will represent the complete rescue care system which is robust, reliable and sustainable. To assess the impact and performance of the software, different use cases can be used as test data. An important concern is to check its usability in real word scenarios which require the software to be portable and/or deployable in devices used by rescue personnel. Also, the time required to provide a valid outcome is required to be tested if used in the computer at rescue vehicles.

2.2 Related Work

Use of decision making systems in various domains is a major research field in the discipline of information systems. Quite a lot of medical research has been found on the development of ML-based software for disease identification and care management. However, after a thorough review of the literature, it was discovered that although machine learning has been successfully applied in various scientific domains of health management, including the detection of health complications, its importance in aiding emergency rescue patients remains underexplored. The type of data that was utilized in this thesis to develop the software, namely rescue data, is unprecedented in this research field. In this section, the published literature on the application of artificial intelligence, such as machine learning, in rescue or emergency events will be reviewed. The impact of AI-based embedded systems in healthcare and existing digital systems will also be a part of the discussion.

2.2.1 Assessment of Health Complication with ML

Incorporation of machine learning into diagnostic assistance systems is a notable aspect in the evaluation of major health complications, giving healthcare professionals a greater ability to improve clinical decision-making and ultimately improve patient outcomes. The major health complications discussed in this dissertation are common in rescue events and always require a quick assessment to save a person's life. To make a prognosis-based assessment, rescue workers often encounter complexities, such as limited time, inadequate patient information, complex and overlapped symptoms, and resource and environmental limitations. Machine learning can be highly resourceful in such adverse conditions, using the medical device and patient's basic health data to infer healthcare assessments. For its rapid decision-making capability, machine learning-based health assessment modules used in detecting, diagnosing, managing, and monitoring the prognosis of different diseases have received increasing attention in different medical sectors and biomedical research works [130] [183]. There are promising research works on the use of ML in the diagnosis and treatment of malignant tumors [186], cancer treatment [57], neurogenerative disease [81], and many others that offer opportunities for the diagnosis and management of disease.

In an emergency rescue event, there are also other applications of machine learning. The integration of machine learning algorithms has proven to be instrumental in detecting and managing critical health complications such as cardiovascular, neurological, and respiratory. [6, 7, 8]. The research papers discussed the entire pipeline on how rescue data is preprocessed, managed, and used for the generation of features for health complications. In addition, a comparative analysis of different machine learning and deep learning algorithms is provided to predict the probability of respective complications in the rescue patient. Palaniappan et. al [134] used support vector machine (SVM) and K-nearest neighbors (KNN) for the diagnosis of respiratory pathology using pulmonary acoustic signals. A comparison of performance between these machine learning algorithms in the diagnosis of respiratory pathology was reported using respiratory sounds from the Respiration Acoustics Laboratory Environment (R.A.L.E) database. The features of the acoustic signal were analyzed with one-way analysis of variance (ANOVA) and then fed separately into the classifiers. The classification accuracy of the SVM and KNN algorithms was 92.19% and 98.26%, respectively. S. Mohan et al. [121] offered an approach of using hybrid random forest with a linear model (HRFLM) to predict heart disease. The computational approach consists of three mining association rules, namely, a priori, predictive, and tertius, to find the factors of heart disease on the Cleveland UCI dataset. An accuracy of 88.4% and a sensitivity of 92.8% was achieved with this proposed method. A. A.-A. Valliani et al., 2019 [168] provided a systematic review of the use of deep learning in the field of clinical neuroscience with existing challenges and a look to the future. The research highlighted the broad spectrum of medical image-based data, their classification, segmentation, and their effectiveness in risk prognosis. The performance of machine learning models using non-image data such as electroencephalography is also reviewed, stating their high sensitivity and specificity for the detection of epileptic seizures. The authors concluded that for an improved diagnosis or early detection of acute neurological events, various domains of deep learning can play a significant role, especially when the patient is suffering from attention deficit (AD) or attention deficit hyperactivity disorder (ADHD). It was also emphasized that to understand the true potential of deep learning, two major challenges must be overcome. Firstly, while technical challenges surrounding the general usability and interpretability of machine learning models exist, more difficult challenges like data privacy and accessibility are important. The second challenge is data quality, which is also a problem in this thesis, as the data collected through rescue procedures is often very raw and required filtering before being used in machine learning models.

In [55], the possibility of applying portable ultrasound combined with artificial intelligence in pre-hospital emergency and disaster sites is highlighted. The described approach enables the diagnosis of numerous medical conditions such as abdominal bleeding, limb fractures, and hemopneumothorax, improving the precision and timeliness of medical interventions in emergency rescue operations. Some research works [109][117][5] also discussed the implementation of an intelligent triage system based on unmanned aerial vehicles in mass casualty incidents using 5G and artificial intelligence. The system helps sort injured individuals in pre-hospital environments after

mass casualty events, thus enhancing the effectiveness of emergency rescue operations.

2.2.2 Machine learning in Treatment Recommendation

In rescue situations, not only is the diagnosis of a health complication important, but also determining the path of treatment such as the medication requirements is necessary. Machine learning technology plays a crucial role in healthcare by enabling healthcare professionals to make more informed decisions, personalize treatment plans, optimize surgical techniques, and enhance patient outcomes. Using machine learning, surgeons can minimize risks and improve patient outcomes through personalized treatment plans. ML algorithms can assess preoperative information, including patient demographics, medical history, symptoms, and test results, to recognize possible risk factors and forecast patient prognosis, thus recommending the necessary medications to aid clinical decision-making [87] [15].

In this study [59], Gill et al. provided a comprehensive review on the use of machine learning algorithms to analyze patient data and predict future potential complications in conditions such as diabetes and heart disease, allowing healthcare providers to personalize interventions and reduce risks. The basis of personalized or patient-specific medicine is the knowledge that the genetic makeup, lifestyle, and health history of each person influence how they respond differently to medical treatments. When ML algorithms are trained with large datasets containing genetic data, medical histories, environmental factors, and treatment outcomes, they can suggest how each person can react to particular therapies, drugs, or interventions. However, the authors also emphasized ethical considerations to ensure patient autonomy, ensure data privacy, and address algorithmic bias that are essential to maintain trust and integrity. Another research by Saeed et al. in 2022 [142] focusing on state-of-the-art machine learning algorithms and their roles in modern healthcare systems mentioned how patient-specific treatment models can support individuals based on their particular characteristics by identifying genetic markers or biomarkers that influence treatment efficacy. The influence of deep learning algorithms to help diagnose and treat patients in the intensive care unit is discussed in [135].

Kumar et al. in their research [88] described how various machine learning algorithms can benefit healthcare professionals by optimizing personal treatment plans, supporting drug development, and improving risk adjustment. An emerging area in the sphere of customized medicine plans is the use of reinforcement learning to formulate dynamic treatment protocols. A benefit of using reinforcement learning is to predict and diagnose diseases early, therefore improving preventive medicine and care for vulnerable groups. In addition, the authors also discussed personalized risk assessment and prognostic models based on ML that can be utilized in clinical decision support systems.

2.2.3 Embedded System in Healthcare

AI models developed for the prognosis of health complications and the determination of the treatment plan can be used in the real world through applications of embedded systems. An embedded system is a microprocessor-based computer hardware system

with software that is designed to perform specific tasks within a larger system. These systems, known for their compact size and integration into various medical devices, play a crucial role in improving the efficiency, precision, and functionality of health-care technologies [154]. In rescue situations, rescue personnel have to use different sensor-based embedded devices that measure patient health vitals to understand overall health status.

These embedded devices are continuously evolving to contribute in different application domains like health data optimization, artificial vision systems, early disease diagnosis, patient telemonitoring, etc. Currently, smart embedded devices for health monitoring are evolving in the form of smartwatches or even sensors integrated into textiles or optical lenses. These smaller and connected devices are making it easier to capture and transmit vital data to AI modules that can serve as a clinical decision support system [13].

Al-Audidy et al. in their research [10] presented the design, implementation, and evaluation of a portable wireless healthcare monitoring system consisting of medical sensors connected to an embedded microcontroller with a wireless communication module. The focus of their research was to integrate a medical sensor-based embedded module with healthcare centers through wireless communication technology. The main health data considered was ECG, body temperature, blood pressure, and oxygen saturation. Similar embedded healthcare systems based on wireless sensor networks were also suggested in [94] where the focus was on developing a mobile embedded system to monitor ECG signals, which is aimed to be used for a ubiquitous health care system. Three different abnormal ECG signals, such as Beth Israel Deaconess Medical Center (BIDMC) congestive heart failure, malignant ventricular ectopy, and Creighton University ventricular tachyarrhythmia, as well as normal ECG signals, were filtered through multilayer perceptrons neural networks to differentiate between normal and abnormal ECG signals, facilitating effective monitoring of a patient's cardiac condition.

Embedding ML models in reprogrammable and reusable hardware is a basic step of prototyping for testing that can be tested in real world scenarios. Springer et al. [155] provided a brief review of the advantages of accelerating ML models directly in embedded devices instead of the conventional high-performance computing approach, especially the potential in medical applications. The advantages of the cited embedded prototypes include reduced latency and energy usage, saving network bandwidth, and enhanced privacy protection since the devices operate without the need for an internet connection. Currently, there are many ongoing state-of-the-art research projects running on ML-based embedded architectures using custom accelerators, where the focus is on achieving high accuracy with low inference time. One of the modern high-efficiency FPGA accelerating tools is Apache TVM and VTA that focus on optimizing machine learning workloads by deploying across diverse devices. The key contributors of Apache-TVM have provided a thorough review of it in [32], while summarizing the challenges faced during the acceleration and optimization of deep learning models at the computational graph level and the tensor operator level. The challenges are high-level data flow rewriting, thread-level memory reuse, intrinsic tensorized compute,

and latency hiding. The authors explained in the paper how using two optimization layers: a computation graph optimization layer and a tensor optimization layer with new schedule primitives TVM resolves the challenges. In another paper by the same contributors [125], a brief overview of the VTA hardware-software stack built into the TVM was provided. An end-to-end performance evaluation was analyzed on multiple edge systems equipped with CPU, GPU, and FPGA. The developers stated that for comparable systems, VTA provides a significant performance edge over conventional CPU- and GPU-based inference, but is unable to accelerate many of the classical machine learning algorithms.

Another promising and rapidly growing ML acceleration platform is tiny ML, which focuses on deploying compressed and optimized machine learning models on resource-constrained devices such as microcontrollers and low-power processors [1]. Saha et al. discussed in their research [143] how tinyML can be a potential tool to bring machine learning-based modules to edge computing. The authors highlighted that conventional ML deployment has high memory and a computing footprint that hinders its direct deployment on ultraresource-constrained microcontrollers. As a solution, they offered a coherent and closed-loop ML model development and deployment workflow for microcontrollers consisting of inference engine optimizations, operator optimizations, and model compression, along with embedded code generation.

2.2.4 Natural Language Processing in Healthcare

Natural language processing tools are widely used in current AI development from feature generation to context-based generative AI. NLP used in this research was mainly focused on extracting features from rescue personnel’ text data that indicate patients’ health condition with complication-specific symptoms. As mentioned in various research works, natural language processing is a critical component in healthcare, allowing the extraction of valuable information from unstructured data sources such as clinical notes and medical journals utilizing computer-based algorithms [75] [85]. Recent developments in NLP are limited not only in the English language, but also present opportunities and challenges in other languages as well [128]. Galatzan et al. published research [54] on the use of NLP software programs to extract structured and unstructured data from electronic health records and other healthcare databases. In their study, an innovative triangulation data analysis within methods was used to analyze nurses and physicians communicating patient information. This method of analysis uses traditional content analysis and an NLP program to provide a deeper contextual comprehension and understanding of the same transcribed text data. This data analysis method is performed in different steps such as 1)linguistic inquiry word count 2)content analysis with systematic coding and categorizing of text data and 3)triangulation.

Combining NLP with deep learning models can offer expert system support for e-health applications, contributing to the development of secure digital healthcare systems, as mentioned in [113]. The research approach used in the study involves the application of NLP to classify and analyze clinical data. The study utilizes a clustering system to gather detailed content from text or speech, categorizing it into an appre-

hensible dataset. Furthermore, the research incorporates parts of speech analysis to showcase the impact of AI and NLP on e-health, highlighting how these technologies can streamline and enhance healthcare services. The study further underscores the importance of metrics and behavior-centric functions in identifying health-related content from textual data, which is essential to understand patient requirements and improve healthcare. Examining comparable research [116] in this domain elucidates the extraction of critical information from diverse sources, including radiology reports, to improve healthcare procedures and decision making. Although the study focuses mainly on using natural language processing (NLP) systems to identify healthcare-associated pneumonia in newborns, it also defines the challenges of implementing an NLP-based monitoring system in a clinical setting, including the need for knowledge engineering, rule refinement, and validation. One of the main shortcomings highlighted was the ambiguous abbreviations and poor English grammar in the clinical text, which was similar to that faced in the rescue text database. The strategies suggested by the authors to overcome the shortcomings are contextual disambiguation, domain granularity, greater collaboration with medical professionals, refinement of the lexicon, and quality control to identify textual errors in the interpretation of clinical text.

2.2.5 Digital Healthcare Software for Rescue Events

Healthcare management software is a comprehensive integrated system that includes specialized applications designed to streamline and enhance various aspects of healthcare care delivery and administration. These software systems are customized to meet the unique needs of healthcare organizations to improve patient care quality, streamline administrative processes, and ensure compliance with regulations. Although extensive research has been conducted on health care management software, little or no empirical work has been done to support rescue events. Some studies like [108][49] defined healthcare software as a significant and critical factor for healthcare policy planning, which consequently affects the hospital management plan, operating cost, and use of resources. Their research emphasizes additional decision support systems that can be used to systematically evaluate current policies and then enhance the quality and efficiency of segmented support, create synergistic effects, and improve decision-making performance and value.

The rescue care assistance software developed in this doctoral study uses predictive analysis of ML algorithms to provide a health complication analysis with a possible treatment path. The significance of such software is well recognized with the advancement of deep learning algorithms. Many of these solutions are capable of analyzing electroencephalography or electrocardiography signals, radiography, X-rays, or CT scan images to detect the actual condition of a patient [152, 2]. Yang et al. [181] provided a conceptual overview of how the integration of machine learning and deep learning-based health care system promises to deliver precise prognostic predictions for diseases such as COVID-19, thus improving physician decision making. In [137], a critical suggestion for the operation of health management software is highlighted, emphasizing that the software should focus on interoperability and smooth integra-

tion with current healthcare systems to ensure effective data transfer and ongoing care. The ability to interoperate allows the software to retrieve essential patient data from multiple sources, including electronic health records and diagnostic reports, thus improving the precision of health complication forecasts and treatment strategies.

2.3 Research Gap

While there are many studies on the role of AI in disease diagnosis and healthcare management, this dissertation focuses specifically on rescue scenarios. The data sets used were also not recorded in clinical settings, instead in original rescue situations. So, health features that could have helped improve the AI models developed were not available in the existing data set. The AI model for deciding treatment paths, mainly medication administration, must consider only the medications that are provided by the emergency rescue personnel to survive the patient. Therefore, this research gap of data unavailability is evident and crucial, as mishandling during emergency rescue situations can lead to numerous deaths. Thus, this research seeks to fill this gap by examining the effectiveness of artificial intelligence (AI) models in diagnosing diseases and managing care, with a strong emphasis on rescue scenarios.

The five requirements outlined in Section 8.1.1 illustrate the research trajectory pursued in this study and facilitated the identification of gaps within the prior scientific literature. None of those previous works met the requirements of this research focused on supporting rescue situations. The identified gaps can be briefly described as follows.

- **Identifying health complication feature from rescue data:** Previous studies on identifying the health complication take place in clinical environments involving various medical tests and comprehensive follow-up by healthcare professionals. In addition, medical centers are well equipped with the instruments required and have the scope to perform laboratory or imaging tests, in stark contrast to rescue scenarios. Data in rescue situations is promptly recorded in the shortest time with some basic sensor-based devices, and often without any contribution from patients. This doctoral study presented a novel route in working with such rescue data and extracting attributes for individual complications from past rescue records. Instead of using medical test data, different AI algorithms were used to extract complication-relevant features from rescue personnel’ textual notes. Finally, a questionnaire was developed for each major complication that can be collected in each rescue event to predict a possible complication.
- **ML-based health complication identification module in rescue situations:** A predictive ML module that works with rescue features is a unique technology presented in the dissertation. Although much research has been done on the prediction of ML-based health complications, no controlled studies have been found for rescue situations.
- **ML based rescue treatment module:** The personalized treatment path

generated by ML explored before is developed for drug development, drug dosing optimization, and drug resistance prediction. using the patient’s genome sequence, clinical features such as abnormal health symptoms, and medical records [67]. This doctoral research explores a different approach by integrating available health information in a resource-constrained environment along with ML-based health complication prediction module to deduce required treatments.

- **Embedded prototype for rescue support:** The innovative portable and embedded prototypes developed in this study are distinguished by their use of AI methods adapted to rescue scenarios, which are capable of examining rescue data and offering practical insights in real time during emergency situations.
- **NLP-supported rescue health support software:** Limited research is conducted on the possibility of using NLP in rescue context, and most are restricted to the summarization of medical data, the interpretation of complex terms or the extraction and encoding of clinical data. The novelty in the developed decision support system software approach in this research is it’s user-centered design principles to ensure that its solutions are specifically tailored to meet the unique requirements of emergency responders in rescue environment while considering the data flaws and ambiguity. In addition, the software output is designed to take into account the probabilities of multiple health complications that will help paramedics in rescue events decide the treatment paths.

3

Research Concepts

To conduct the doctorate research, several research concepts and scientific algorithms were used. The focus of the research was on developing a reliable and high-performance machine learning-based module that extends existing rescue systems. To achieve this goal, various machine learning algorithms including neural networks were investigated to identify the best performing algorithm. Different feature extraction and attribute selection methods for each complication recognition module were utilized, such as data interpolation, extrapolation, natural language processing, etc. To understand the real-time applicability of machine learning algorithms, hardware embedding concepts were also utilized. This chapter will provide an overview of these software and hardware embedding concepts along with the description of the rescue data that created the foundation for the rest of the work.

3.0.1 Natural Language Processing Algorithms

Natural language processing (NLP) is a subset of AI, an important tool to extract features from textual data. It is done by transforming unstructured text data into structured data that can be analyzed and utilized by machine learning algorithms to find context. To extract feature from the text data, the first step is pre-processing. Below are some key components of NLP that generally used by researchers to extract information from text data [20] [42]:

- **Text normalization:** Converting all texts to lowercase and then eliminating characters that do not qualify as words or white space, as well as removing any numerical digits present
- **Tokenization:** Breaking text into manageable pieces, such as words or phrases, known as tokens

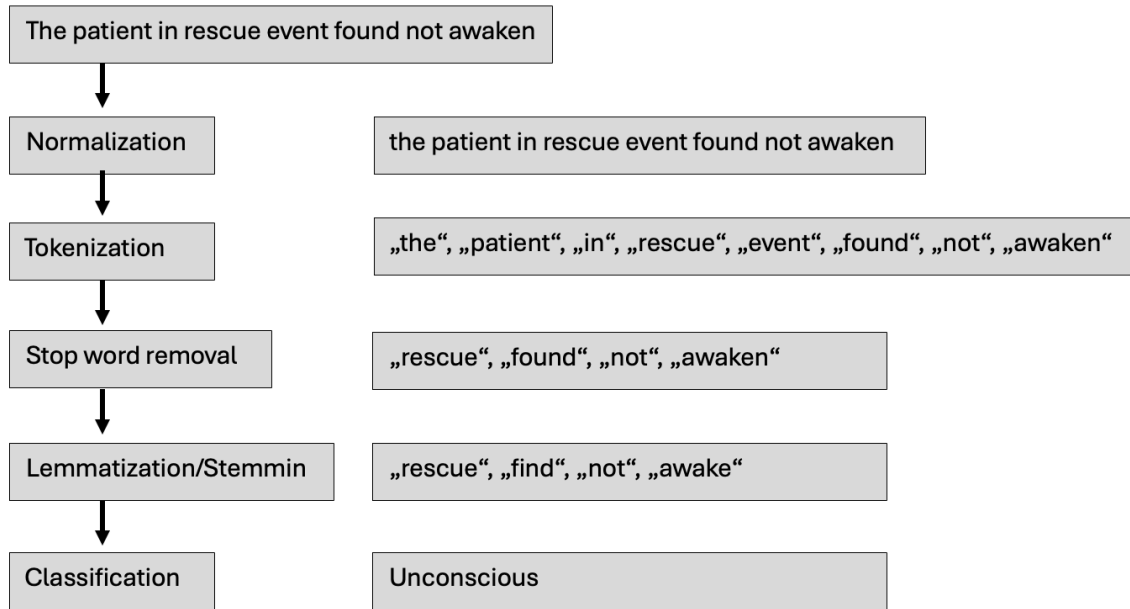


Figure 3.1: Example of NLP-based Preprocessing Steps

- **Lemmatization/Stemming:** To break down a word to it's root meaning to identify similarities
- **Stop word removal:** Removing common phrases, trivial words from texts for further processing
- **Text classification:** Categorizing text into predefined classes or categories

Figure 3.1 provides a classic example of how the preprocessing steps work. After the data preprocessing, the next step in NLP is feature extraction. This step follows two main tasks: first, creating a valid database with meaningful text information extracted with NLP and then categorizing that text information to generate features that can train machine learning models. Some common algorithms used in feature extraction are shown in Table 3.1 [103, 102].

To facilitate this concept with rescue data, a Python-based library such as the NLP toolkit 'NLTK' can be used, which is one of the most widely used libraries for natural language processing in Python programming. NLTK includes prebuilt functions and utilities designed to facilitate usage and implementation. In order to utilize NLP methodologies in the text data columns of the rescue record for the purpose of extracting necessary information, it is essential to establish the object and scope of the data source, as well as the objective of the information extraction. Afterward, an approach such as machine learning, deep learning, or a domain-specific rule-based algorithm based on data type is used. When data is highly unprocessed, constrained, and adheres to domain-specific patterns, such as rescue records, a rule-based algorithm is generally preferred [176].

Algorithm	Description
Bag of Words (BoW)	Represents text as a collection of word frequencies or binary occurrences without considering grammar and word order.
Term Frequency-Inverse Document Frequency (TF-IDF)	Determines how significant a term is to a text in a series or a corpus by assigning a weight to each word in a document based on its term frequency and the reciprocal document frequency.
N-grams	Captures neighboring sequences of words or characters in text. Useful for local pattern detection.
Part-of-Speech (POS)	Assigns grammatical labels such as nouns, verbs, and adjectives to words. It assists in recognizing syntactic structures.
Domain-Specific Dictionaries	Employs established terminologies or key phrases pertinent to particular domains or fields. Commonly utilized in specialized textual analysis.

Table 3.1: Traditional Feature Extraction Algorithms of NLP

3.1 Machine Learning Models

Machine learning is a subset of artificial intelligence that focuses on enabling systems to learn from data, identify patterns, and make decisions without explicit human instruction. By analyzing data provided by users, machine learning models can mimic aspects of human intelligence, extract valuable information, and make informed predictions or decisions. Popularity of ML algorithms revolves around two basic scientific needs:

- Development of an algorithm capable of continuously learning from a process or system to conduct future predictive analysis
- Develop the algorithm in a way so that it can improve itself in decision making

To satisfy those needs, for decades several algorithms were developed that have been found to be quite effective for datasets where the human brain is not capable of drawing necessary insights due to the complexity and size of the data. By analyzing large-scale medical data, ML algorithms can uncover patterns, predict disease progression, and assist healthcare providers in making data-driven decisions. There are a number of algorithms that researchers use now-a-days, but no single algorithm can be said to be optimal for all types of data and problems [106]. Therefore, implementing various algorithms is essential to determine which one offers optimal performance with a defined data set. The following subsections offer a concise discussion of different ML algorithms to apprehend their operational principles.

3.1.1 Classical Machine Learning Algorithms

Classical ML algorithms are foundational methods in the field of machine learning that have been extensively studied and applied in various domains for predictive analysis.

These algorithms usually have simpler structures and higher interpretability than deep learning models, making them ideal for numerous applications, particularly in scenarios where data availability is restricted or computational resources are limited. However, for databases that involve large-scale, high-dimensional, or unstructured data, the performance of classical algorithms is not satisfactory due to the complex nonlinear relationships between independent and dependent variables or lacking in feature interaction [164]. Generally, the classical approaches are divided into 4 types:

- **Supervised machine learning (SML)** - Supervised algorithms refer to a process of providing input data as well as labeled output data to train an ML model for predicting or classifying objects, regression analysis. There are two types of SML: classification and regression. Classification is the type of SML where the result has a discrete value such as 0, 1 and the aim is to predict the discrete values that fit a specific class. Regression is the type of SML that is acquired from the labeled datasets and continuous value results are predicted for the latest data that are given to the algorithm [120].
- **Unsupervised machine learning (UML)**- UML is an algorithm in machine learning which, unlike supervised algorithms, has no labeled data but still tries to form a pattern between feature vectors and create clusters with similar datasets. As described in [167], UML is divided into six major categories: hierarchical learning, data clustering, latent variable models, dimensionality reduction, and outlier detection.
- **Semi-supervised machine learning(SSML)** - SSML is a branch of machine learning where both labeled and unlabeled data are used for predictive and regressive analysis. Generally, a small amount of labeled data with a large amount of unlabeled data is used to train. Certain studies have shown that combining unlabeled data with a limited amount of labeled data can significantly improve learning accuracy. [93].
- **Reinforcement machine learning(RML)** - Reinforcement learning aims to discover the best strategy in a dynamic environment to achieve the highest possible reward. The algorithm enables the system to learn through iterative trial-and-error sessions by incorporating feedback from its actions, similar to how humans and animals learn from the results of their behaviors before choosing the best possible action. Reinforcement learning consists of two types: active and passive. In passive learning, the algorithm has to be told what tasks to perform and in what states. But in active learning, the algorithm has to determine its actions as no consistent policy is in place according to which it can decide.

The rescue data in this thesis has diagnostic information for almost all rescue events that was used for labeling data to categorize events under different health complications. Therefore, the use of a supervised machine learning algorithm was appropriate for the development of predictive analysis regarding health complications. As there

were many algorithms to select from, based on state-of-the-art research, six algorithms were considered, as described in the following parts.

Support Vector Machine

Support vector machine (SVM) is a popular supervised learning algorithm that is often used by machine learning researchers for both classification and regression tasks, specifically in high-dimensional spaces. It operates by finding the optimal hyperplane that separates data points of different classes in a high-dimensional space. The key principle behind SVM is to find the hyperplane with the maximum margin between the closest points of different classes, known as support vectors [18]. In Figure 3.2, the SVM working algorithm is depicted.

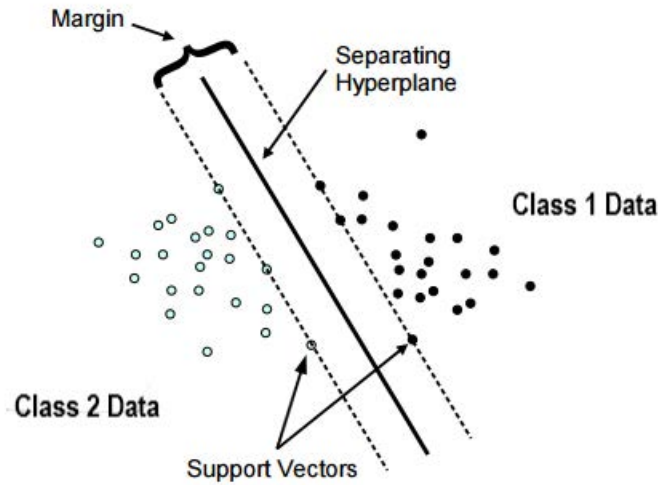


Figure 3.2: Support Vector Machine

SVM works very efficiently when the number of feature vectors is greater than the number of samples, ultimately helping to maintain robust performance even with a limited amount of training data [43]. Because the algorithm uses only a handful of feature vectors for classification and focuses on maximizing the margin between classes, it is resistant to overfitting. While developing a model with SVM, the developer has the opportunity to adapt different kernels such as linear, radial basis function, or polynomial which supports complex data relationships by creating nonlinear hyperplane. SVM is also quite efficient in biomedical research, particularly in diagnostic and predictive modeling, due to its ability to handle complex, high-dimensional data and to classify cases with high accuracy. In addition, in medical images, for example, X-ray, MRI analysis for anomaly segmentation, genomics to classify gene expression data, or early disease detection based on patient signs and symptoms, SVM has proven its effectiveness [104, 90, 151].

When training data are provided to the SVM, each feature vector is plotted in an

n-dimensional space, where n corresponds to the number of features. As the feature vector comes with a label indicating different classes, to separate them, a hyperplane is drawn focusing on the maximum margin between classes. The vector points closest to the hyperplane are chosen as support vectors. If a linear hyperplane is not possible, using 'Kernel trick', a non-linear hyperplane is chosen by mapping the input features into a higher-dimensional space. When a test feature vector is provided, then after mapping it, the SVM classifies the new vector point by determining which side of the hyperplane it may fall into.

Random Forest

Random forest(RF) is a popular supervised learning algorithm that uses the ensemble learning for classification and regression tasks. It operates by combining multiple decision trees during training and outputs the mode of the classes (classification) or mean prediction (regression) of the individual trees to deliver accurate and robust performance. The approach followed by RF is based on the principles of the bagging method, where multiple subsets of the training data set are created and later assigned each to a decision tree, thereby ensuring the model performance [24, 52]. In Figure 3.3, a graphical demonstration of the algorithm is provided. The algorithm can handle high-dimensional data effectively while delivering high accuracy due to the combination of multiple decision trees. Also, because of the use of multiple decision trees, it averages all of their results to minimize the risk of overfitting and variance, especially on noisy datasets in the medical field. Besides, it provides an estimate of the importance of the features that helps users identify the group of important features [161]. Within the medical field, the RF algorithm is often employed as a diagnostic method. As discussed in [12, 126], it has shown impressive accuracy in forecasting outcomes for patients with heart disease and diabetes using clinical data. Moreover, in many other domains like medical image analysis, genomics and bio-informatics, or in clinical decision support system, the algorithm is often used. The rescue data in this research often have many records where important features are missing. Random forest is quite suitable in such cases, as it can handle missing data and maintains accuracy even when a large proportion of the data is missing [112].

Extreme Gradient Boosting

Extreme Gradient Boosting (XGB) is known as a scalable machine learning algorithm which has high speed, and accuracy when working with structured data. It belongs to the family of ensemble learning, specifically the gradient boosting algorithm that combines the outputs of several weak learners, typically decision trees, into a single strong learner. The major idea behind XGB is to iteratively train models on the residuals of the wrong predictions made by previous models. Then it employs a technique known as "gradient boosting," which optimizes a differentiable loss function by fitting new models to the residuals of the existing models. The iterative training is achieved through the construction of decision trees as its base learners in a sequential manner, where each new tree attempts to correct the errors made by the previously

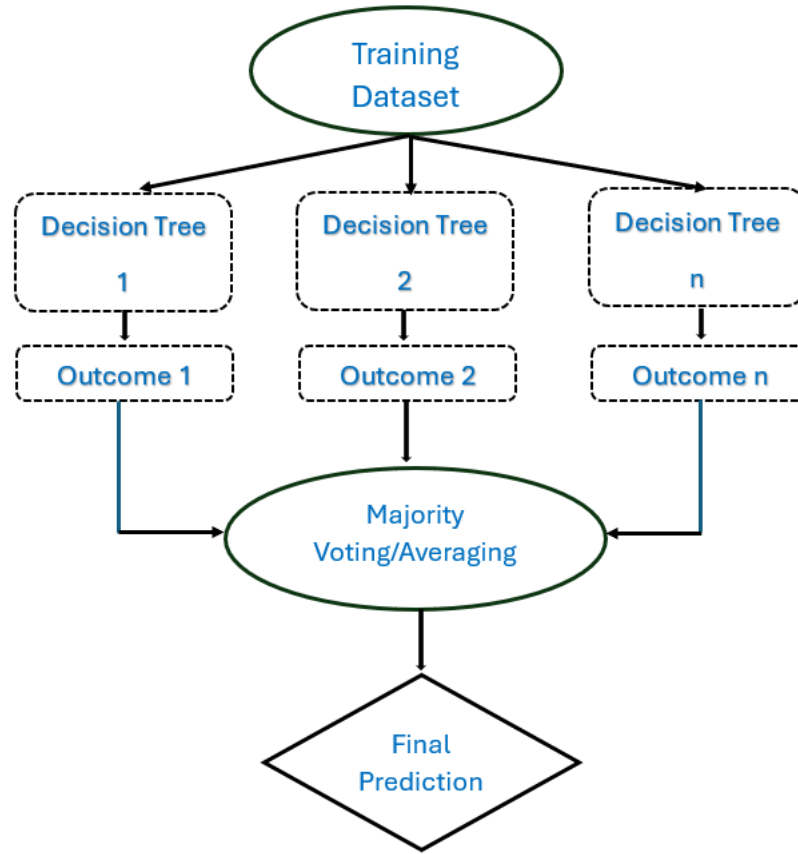


Figure 3.3: Random Forest Algorithm

built trees when a training dataset is provided. This process continues until a specified number of trees is reached or the improvement in the loss function falls below a certain threshold. This approach not only improves the performance of the model, but also enables regularization unlike the traditional gradient boost method, which aids in overcoming challenges like overfitting in machine learning tasks [31, 174, 39]. The performance of the XGB model is generally very high, surpassing other algorithms, including Random Forest and Support Vector Machines, in terms of accuracy and computational efficiency [101]. The structure of this algorithm can be observed in Figure 3.4. Use of the XGB model in the biomedical area is quite beneficial, as it often achieves a high precision and can handle complex patterns in medical data, making it suitable for tasks like disease prediction, patient risk assessment, and diagnostic classification. Besides, as it can handle large datasets with extensive features, it is particularly suitable for complex medical datasets, where it can effectively identify patterns and risk factors associated with certain health complications [62]. XGB's ability to manage missing values and outliers is advantageous, too, because the rescue data has a lot of ambiguous and missing values.

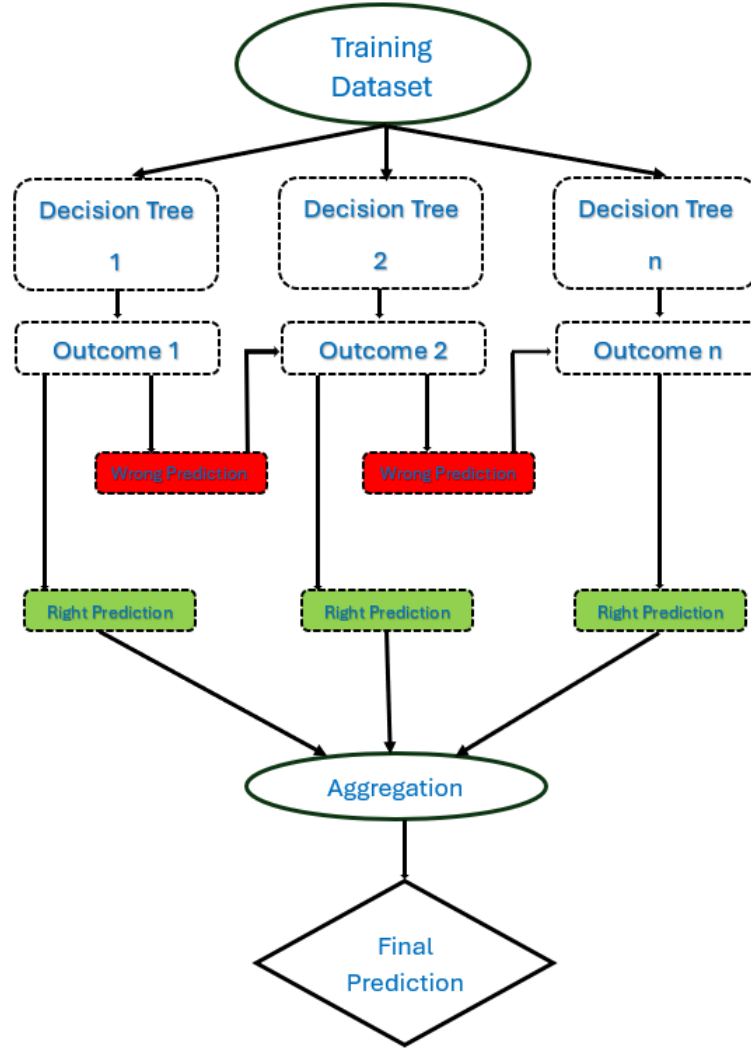


Figure 3.4: XGB Algorithm

Logistic Regression

Logistic regression (LR), a type of supervised machine learning algorithm, is a widely used statistical method for binary classification tasks, where the goal is to predict the probability of a certain event occurring based on one or more independent variables. The algorithm establishes the relationship between the dependent and independent variables by calculating probabilities with a logistic function that serves as the cumulative distribution function of the logistic distribution. This function maps any number with real values into the $(0, 1)$ interval, where 1 is the outcome if the predicted probability is greater than a threshold (commonly 0.5), making it suitable for predicting probabilities [147]. The fundamental formula for logistic regression can be

expressed as:

$$x(t) = \frac{1}{1 + e^{-t}} \quad (3.1)$$

Here, t is a linear combination of the input features often expressed as a polynomial equation. The coefficients of the equation are typically estimated using maximum likelihood estimation, which finds the parameter values that maximize the likelihood of observing the given data. $x(t)$ is the logistic (or sigmoid) function that transforms the linear output into a probability [165]. In Figure 3.5 an S curve is drawn between two classes to demonstrate how the regression line from the cumulative distribution function is formed that can separate the classes.

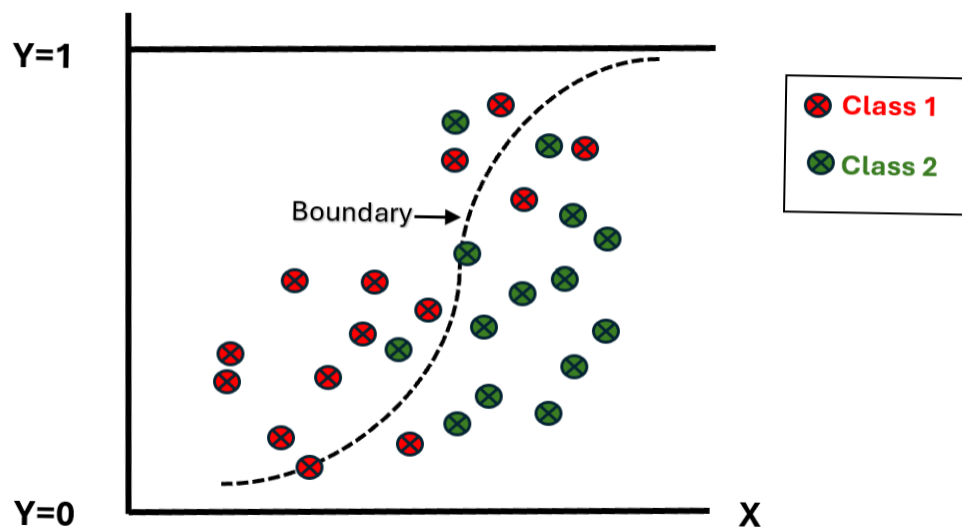


Figure 3.5: Logistic Regression Algorithm

LR is particularly useful in fields like medicine, finance, and social sciences due to its interpretability and effectiveness in handling binary outcomes. One of its biggest advantage is that it is computationally efficient and requires less computational power than more complex models like XGBoost. Recent studies, as described in [136, 78], indicate that employing logistic regression to predict the probability of diseases such as heart disease and diabetes using patient data has yielded high accuracy rates. However, the research also explained that for the algorithm to perform well with good accuracy, the data set needs to be small, or the relationships between variables should be approximately linear.

Naive Bayes

Naive Bayes (NB) is a probabilistic machine learning model following Bayes theorem which assumes that the features used for classification are conditionally independent although in real-world scenarios this 'naive' assumption is often untrue. The algorithm belongs to a family of generative learning and is used for text classification,

spam filtering, sentiment analysis, and recommendation systems. During the training period, it initially requires a training dataset to estimate the prior probabilities of each class and the conditional probabilities of each feature given the class. During classification, NB calculates the posterior probability for each class and assigns the class with the highest probability to the data point. The below equation expresses the algorithm [182]:

$$P(C|X) = \frac{P(X|C) \cdot P(C)}{P(X)} \quad (3.2)$$

Here:

- $P(C|X)$ is the posterior probability of class C given the features $X = (x_1, x_2, \dots, x_n)$.
- $P(X|C)$ is the likelihood of observing the feature vector X given that the class is C .
- $P(C)$ is the prior probability of class C .
- $P(X)$ is the marginal likelihood, or the total probability of the feature vector X .

The simplicity of the Naive Bayes classifier makes it computationally very fast with easier implementation, which is particularly advantageous in biomedical scenarios with large data sets or when real-time predictions are expected, such as in emergency events [172, 159]. In one promising research [38], the authors have expressed the high precision of 99% achieved by the NB algorithm to identify mucopolysaccharidosis type II using electronic medical records, indicating its potential as a diagnostic tool in the clinical environment. Sometimes, it may occur that the training dataset consists of meaningless characteristics which are particularly harmful to the accuracy of many ML models like decision trees, instance-based learning, and rule learning. However Naive Bayes can still perform well because it calculates the probabilities for each feature independently [174].

K-Nearest Neighbors

K-Nearest Neighbors (K-NN) is a simple, non parametric, and instance-based supervised machine learning algorithm primarily used for classification and regression tasks. KNN is a lazy learner that operates on the principle of instance-based learning instead of the provided training data. The classification of a new data point is determined based on the majority class among its k nearest neighbors in the feature space. The algorithm is non-parametric, meaning that it does not make any assumptions about the underlying data distribution, which allows it to adapt flexibly to various types of data [131, 163]. The math behind this algorithm is quite simple and fast and can be explained in the following steps:

- First, it maps the feature vectors in the feature space during training and avoids further modification to the data points

- When a test data set is provided, it calculates the distance between the new data point and each instance in the training set. Common distance metrics are: Euclidean Distance, Manhattan Distance, and Minkowski Distance
- Next, it selects the k closest neighbors to the test data point. The value of k is a crucial parameter that can significantly affect the performance of the algorithm; a smaller k may lead to noise sensitivity, while a larger k may smooth out class boundaries [163, 166]
- Afterward, it proceeds with a voting algorithm by checking the class labels of the selected k neighbors and assigns the majority class to the test data point. For regression, instead of taking a majority-based decision, it proceeds with the average value

Figure 3.6 demonstrates how a new test data point is categorized using K-NN.

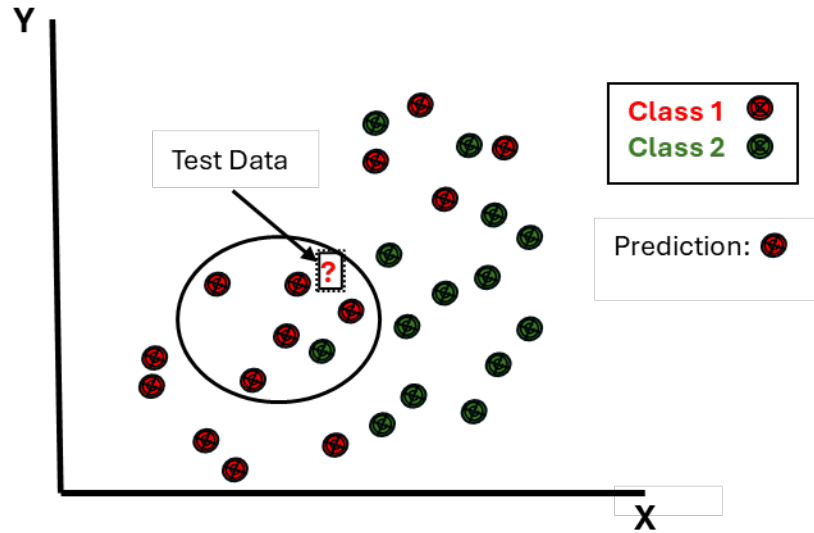


Figure 3.6: K-nearest Neighbors Algorithm

Like other classical algorithms, K-NN is a valuable tool in the medical domain. The algorithm is used in disease diagnosis, medical image-based classification, or risk identification due to its simplicity and interpretability, and to assist healthcare professionals in decision making processes [63]. In some studies [16, 71, 19], the high performance of K-NN is shown in detecting heart disease, classifying metabolic diseases, or diagnosing complicated body conditions such as primary biliary cirrhosis. However, K-NN also has some disadvantages, particularly in terms of computational efficiency and sensitivity to ambiguous features and the curse of dimensionality. As the number of features increases, distance metrics can become less meaningful, potentially affecting classification performance [166, 146].

Adaptive Boosting

Adaptive boosting, also known as AdaBoost, is also an ensemble machine learning algorithm that combines the strengths of multiple "weak" classifiers to form a "strong" classifier. The fundamental principle behind AdaBoost is to sequentially apply weak classifiers, like decision trees, to the training data, adjusting the weights of the instances based on the classification errors made by the former classifiers. This continuous process allows AdaBoost to focus more on instances that are difficult to classify, thereby improving the overall accuracy of the model [80]. The working tactics of the algorithm can be defined in 4 steps as shown in Figure 3.7.

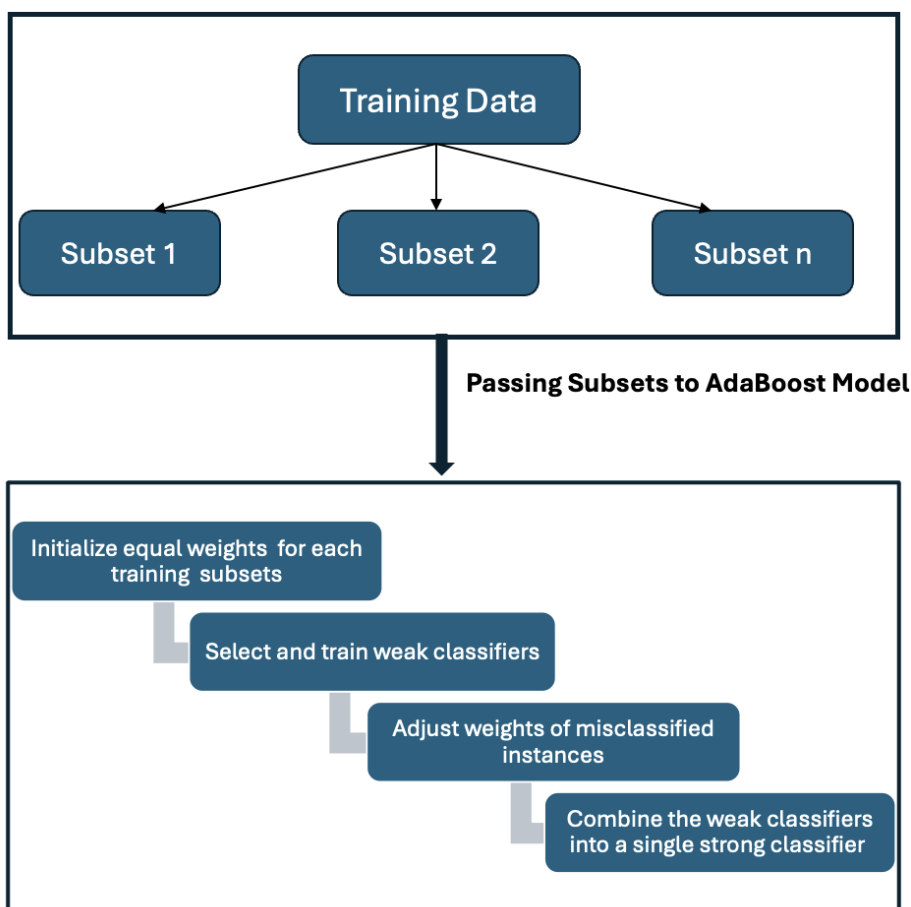


Figure 3.7: Working Methodology of Adaptive Boosting Algorithm

There are certain advantages in using the AdaBoost algorithm in medical data considering the complexity of the data. Several studies have shown that AdaBoost can be used effectively in the diagnosis of diseases like diabetes, hypertension, and a range of cancers, with notable success in achieving high precision [66, 145]. AdaBoost was chosen for drug recommendation models due to its robustness to overfitting and its ability to handle imbalanced data. Rescue data often lacks medication information and, in many cases, no medication was provided to patients. The number of cases

in which medication is administered and recorded was low, which can deteriorate the performance of the model due to data imbalance. Although there is a chance that the algorithm might overfit when dealing with noisy data, it generally has a lower likelihood of such a flaw compared to other boosting algorithms. A major benefit of AdaBoost lies in its emphasis on incorrectly classified instances as an integral aspect of its operation, resulting in improved accuracy. However, the algorithm was often found to be sensitive to noisy data during the development of the treatment model, as the data set was full of anomalies and outliers. The model with AdaBoost may not perform well if the data imbalance is not handled properly prior to the event.

3.1.2 Deep Learning Algorithms

Deep learning is a subset of machine learning that is inspired by the structure and function of the human brain. It employs artificial neural networks with numerous layers to capture intricate patterns within the data. The most significant advantage of deep learning is that it can automatically extract features from raw data without the need for manual feature engineering. Deep learning algorithms, especially convolutional neural networks (CNNs), have become the methodology of choice for tasks in various domains such as image recognition, segmentation, classification, object detection, natural language processing, and speech recognition [107, 21]. The most popular deep learning algorithms are: CNN, recurrent neural networks (RNN), long- and short-term memory networks (LSTM), and generative adversarial networks (GAN). The working principle of deep learning can be explained as follows:

- **Data input to multilayer:** The raw data for training is fed to the network which consists of multiple layers such as an input layer for receiving the data, user-defined hidden layers to modify data, and an output layer which produces a probability score as output for a given problem. This process is called forward propagation.
- **Weight distribution and adjustment:** Each layer has interconnections with its neurons, and the connection between neurons has a weight that determines the importance of the input. During model training, an initial weight assigned to the connection is adjusted for better probability results.
- **Activation function:** An activation function is applied to the neurons of each hidden layer to introduce nonlinearity into the network, allowing the model to learn more complex patterns. This function decides whether a neuron should be activated or not. Common activation functions are Sigmoid, Rectified linear unit (ReLU), and Tanh.
- **Loss calculation:** When the input data passed to the neural network layers reaches the output layer, the predicted outcome is compared to the actual value to calculate loss using a loss function such as mean square error or cross entropy.
- **Backpropagation and optimization:** The weight initially assigned between neurons undergoes continuous adjustment to minimize the errors between the

predicted and actual values. This process is generally executed using backpropagation and optimization techniques, such as gradient descent.

- **Training:** The training of the neural network model is done considering two parameters: epochs and batches. Batch indicates the number of times a training data set is divided and fed through the network to check the output followed by weight adjustment. The epoch is a hyperparameter that indicates how many times the learning algorithm will iterate over the entire training dataset. The batch size must be greater than or equal to one and less than or equal to the number of samples in the training dataset. The number of epochs is an integer value between one and infinity.

A detailed view on the construction of neural networks with the components mentioned above can be viewed from Figure 8.1.

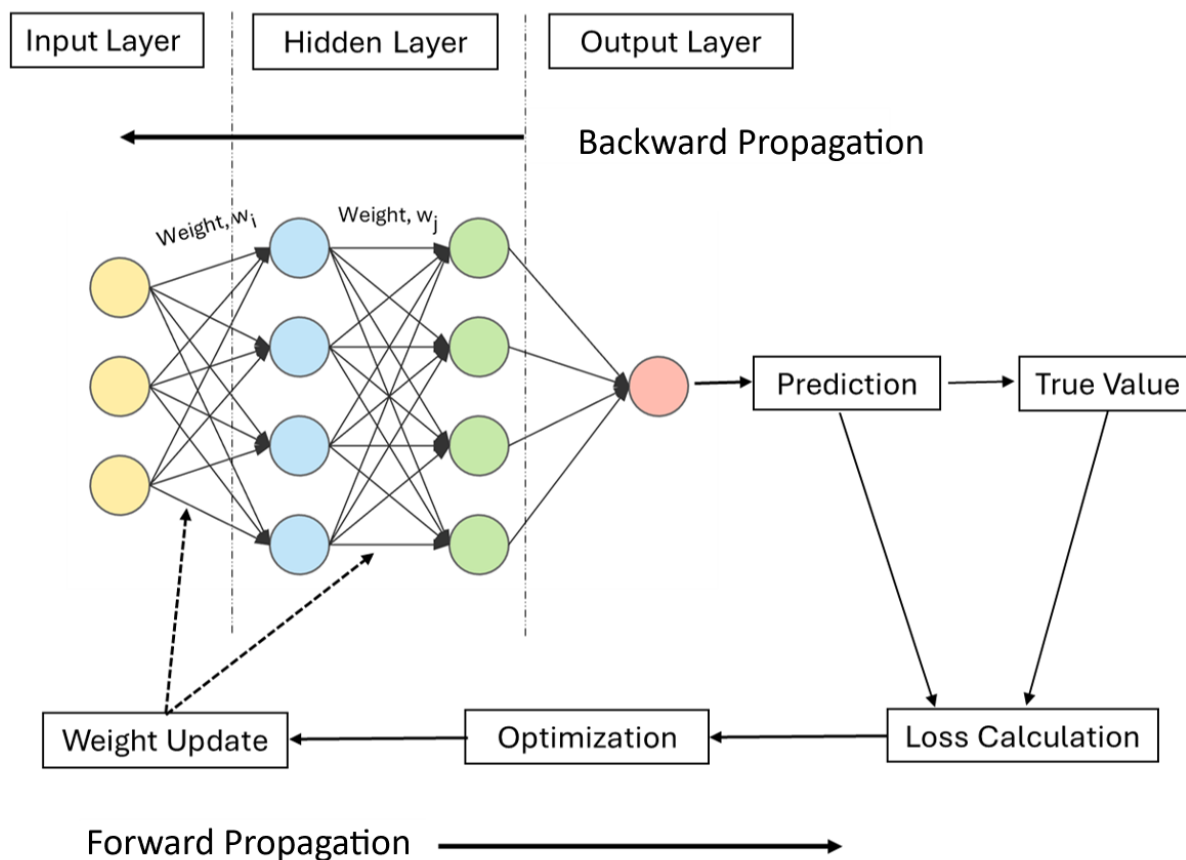


Figure 3.8: Artificial Neural Network Algorithm

Deep learning has demonstrated high effectiveness in the analysis of medical data, particularly when dealing with high-dimensional or unstructured datasets such as images, clinical notes, and genomic information. Some CNN-based research has highlighted the successful implementation of the algorithm in detecting neurological com-

plications like Alzheimer’s and brain tumors by identifying subtle patterns and disease progression in neuroimaging data that human eyes may overlook [177, 77]. Even for predictive modeling, deep learning can be utilized for automatic analysis of pathology data, and patient’s electronic health records to predict outcomes, such as readmission risks or disease progression. There are also studies that report that deep learning algorithms can achieve performance levels comparable to or even exceeding those of human experts in the identification of various diseases from medical data [105]. However, there are also some limitations of using deep learning algorithms such as high training time and resource requirements because of the complex model architecture. In machine learning research, deep learning is considered a powerful tool for various medical applications, with the potential to advance diagnosis, treatment, and patient care.

3.2 Real Time Applicability

The efficacy of machine learning algorithms in patient health diagnosis is highly dependent on their ability to perform effectively with real-world data in real-time environments. Real-time deployment of ML models facilitates quick decision-making in milliseconds and adaptive responses, offering significant advantages in improving outcomes and operational efficiency. Specifically, models designed to recognize health conditions or recommend treatments are expected to operate in actual rescue scenarios, necessitating minimal latency in predictions. This imposes two critical challenges for the algorithms developed:

- Are the algorithms suitable for deployment in real-world rescue operations?
- Does the inference time of the algorithms hinder the performance of the rescuers?

The implementation of these algorithms can be realized through two primary approaches: software-based applications and hardware-accelerated systems. In contemporary health-focused ML research, the primary emphasis often lies on software solutions and their performance with clinical datasets. However, in the context of emergency healthcare, it is equally critical to develop hardware prototypes capable of delivering high performance and reliability under real-world conditions. This doctoral research investigates both approaches and evaluates their effectiveness in addressing the challenges of real-time applicability. The subsequent subsections provide a comprehensive discussion of each approach, offering insights into their potential for real-world deployment.

3.2.1 Rescue Care Software

. Rescue stations around the world receive millions of emergency rescue calls each year, most of which are due to health complications. Due to the high frequency and necessity of rescue services, there is always an increasing demand for quick, accurate, and coordinated responses from rescue personnel to save lives and mitigate damage.

Rescue care assistance software is a subset of health management software that focuses on supporting rescue personnel in identifying the patient’s health situation and accelerating the treatment path. These software applications serve clinical institutes or hospitals by acting as specialized digital tools or platforms, aimed at enhancing different facets of healthcare delivery, including patient care, administrative functions, and data management. Currently, many medical organizations are also using it as a clinical decision support system to support healthcare providers in quickly accessing patient information, monitoring treatment plans, and tracking health outcomes, which ultimately leads to improved patient care and treatment results [48].

3.2.2 Embedded Accelerators

Contemporary research on embedded accelerators for machine learning models that detect health complications has gained significant traction due to the growing demand for high-throughput, energy-efficient, and real-time ML inference. To use the medical decision support system, rescue personnel need to carry a portable device capable of generating health predictions and treatment recommendations upon request. The core computation unit of such portable devices consists of programmable hardware platforms where AI modules can be accelerated. The commonly used embedded accelerators are ARM processors, graphics processing unit (GPU), FPGA, ASIC, and microcontroller which are optimized for specific types of AI workloads. These features allow for faster computations and less energy use, which are crucial to integrating AI solutions into real-time applications [162, 184]. Generally, the software version of the AI models is developed on local computers with a central processing unit (CPU). Afterward, the developed and trained models are accelerated in hardware accelerators, capable of rapid inference when user data is provided. Deployment on small portable embedded devices for machine learning (ML) in emergency rescue situations presents several significant benefits that enhance the patient survival ratio and the operational efficiency of rescue personnel. The key benefits can be listed below:

- **Real-time data processing:** Embedded accelerators like FPGAs have multiple processing units that can be configured to perform multiple operations simultaneously. This simultaneity facilitates high-throughput data handling, which is essential for real-time implementations [37]. In addition, it is highly beneficial for the matrix and vector operations commonly found in machine learning algorithms. Likewise, a microcontroller has efficient technology such as interrupt handling, direct memory access controllers, code optimization with loop unrolling, and inline functions that reduce the number of instructions and allow preprocessing tasks to occur concurrently with other operations [89, 45]. This capability is particularly beneficial in rescue situations, where quick assessments can significantly impact patient survival [36].
- **Portability:** As the hardware platforms are smaller comparable to CPU and GPU, but still provide powerful performance, they are very suitable to use in handheld devices like smart watch. Also, patients in distressed condition can be primarily supported with sophisticated diagnostics instantly at the point of

care without needing for constant connectivity to centralized data centers or to travel to specialized healthcare facilities, which can be a limitation in rural or underserved areas [95, 118]

- **Low power consumption:** Handheld medical support devices in emergency scenarios must operate consistently and have reliable power stability, as such incidents may occur in challenging environments without access to electrical power. The discussed small size hardware’s fulfill this requirement as they are designed for energy-efficient operation ensuring extended operational time without the need for frequent recharging
- **Scalability and reduced cost:** Accelerating ML models on these hardware reduces operational cost, as it reduces the dependency on traditional server and cloud infrastructure that requires high-speed connectivity [22]. Furthermore, accelerators can be utilized in various medical instruments for numerous procedures, facilitating scalable solutions for comprehensive health monitoring, screening, and diagnostics
- **Support for rescue personnel:** When hardware is accelerated with situation recognition and patient care models, they can automate routine diagnostic tasks followed by treatment suggestions, which reduces the cognitive load on the rescue care provider, allowing them to focus on more complex decision-making. In general, patient care will be better and faster with less human involvement and a broadened care spectrum [68, 111]

Using hardware accelerators for ML models offers significant performance and energy efficiency benefits, but it also presents unique challenges. For instance, employing FPGAs as accelerators often requires expertise in hardware description languages such as Verilog or VHDL, along with extensive hardware-software co-design efforts. This makes the development process both complex and time-intensive. To address these challenges, researchers frequently leverage custom library packages like Conifer or ML compilers such as TVM-VTA and FINN. However, these third-party solutions are not universally compatible with all ML models and can suffer from limited or delayed support when the accompanying software environments are updated. Another common approach to simplify FPGA programming is high-level synthesis (HLS), which enables developers to use higher-level languages like C or C++ to design hardware-specific solutions. Despite these advancements, accelerators, such as microcontrollers and FPGAs, still face resource constraints, limiting their scalability for larger, more complex models, such as GPT-like transformers. To overcome this, model pruning techniques are often employed to reduce the computational and memory demands of ML models while maintaining their performance. There are also challenges for accelerating ML models on microcontrollers as they have more resource limitation and limited tool support.

This doctoral research aims to explore these challenges and solutions by conducting a comparative analysis of various hardware accelerators and their suitability for

relevant ML models used in rescue scenarios. By evaluating resource-constrained platforms and optimization techniques, this study seeks to provide valuable insights into the trade-offs and practical considerations of deploying ML models on hardware accelerators.

4

Rescue System Model

The system model is a structured representation of a system and its environment that describes the flow of information between modules. It explains the interaction between processes and the tasks they execute, but does not go into the specifics of the implementation of the process [149]. When the concern is to improve the performance of a system and identify bottlenecks, it is necessary to understand how the system components interact with each other, as the system architecture is not static, rather varies significantly across different domains and applications.

Rescue stations around the world generally use a structured system architecture that is used to coordinate rescue activities and manage the internal emergency service protocol. Developing a sustainable and efficient rescue system is necessary to reduce casualties and economic losses during the response phase [179]. This emergency system is essential for addressing acute medical needs outside traditional healthcare settings, thus playing an important role in reducing the morbidity and mortality rates associated with emergencies [14]. The rescue station in Siegen-Wittgenstein, relevant to this investigation, established its system as a call center where people can report medical emergencies. Afterwards, the system is expanded to various stakeholders and services to maintain an efficient rescue service. The inclusion of AI in this system model is a new prospect and needs to be investigated to find out how it can improve rescue care by increasing the patient's survival rate. This chapter focuses on explaining the existing rescue system, detailing its components, and assessing the impact of incorporating AI technologies into the system.

4.1 Organization of Rescue Services and Stakeholders

Emergency medical response is important in helping and saving people during a fatal health crisis. There are several types of emergency situations that rescue stations deal with. An overview of the major rescue situations is provided in Table 4.1. Organizing its functionalities for these situations in different rescue stages is a primary requirement of every rescue station. These stations are often part of a collaborative network that includes stakeholders like medical ambulances, emergency rescuers or physicians, rescue accessories, and other support services. Together, they create a balanced ecosystem that ensures efficiency, reliability, and successful outcomes during emergencies. The key stakeholders can be classified into two groups: the rescue service providers and the service receivers. The providers encompass the rescue station, its staff, and equipment, while receivers are emergency patients. Figure 4.1 depicts an illustration of these stakeholders.

Table 4.1: Type of Emergencies

Emergency Type	Description
Medical Emergency	People suffering with different health complications like sudden stroke, asthma attack, etc.
Natural Disaster	Saving collapsed people from earthquake, flood affected area, hurricane, and wildfire
Massive Accident	Massive road accidents, train derailments, aviation or industrial accident
Fire	Evacuate people in the events of residential or commercial fires
Urban Incidences	Save trapped individuals from collapsed buildings, gas leaks, or explosions

In addition to these major stakeholders, there are other external members connected to the rescue system, such as the community and the volunteer organization. A prime example is the Red Cross, which frequently responds to large-scale emergencies by deploying their medical ambulances along with paramedic and physician teams. The Red Cross also provides psychological support and counseling help for people who experience trauma or mental shock due to an accident [110]. All of these stakeholders follow specific guided protocols during an emergency to ensure that the right people receive the right service. During the research timeline of this thesis, some workshops were conducted by the rescue station, Siegen-Wittgenstein, where these protocols were described. Every day, control centers report receiving more than 1000 calls, a significant number of them being fake or unclear emergencies. These unexpected events often hinder efficiency, yet the station adheres to its protocols to identify fraudulent calls and offer assistance to those genuinely in need. The essential measures taken to structure the current rescue system in Germany are as follows.

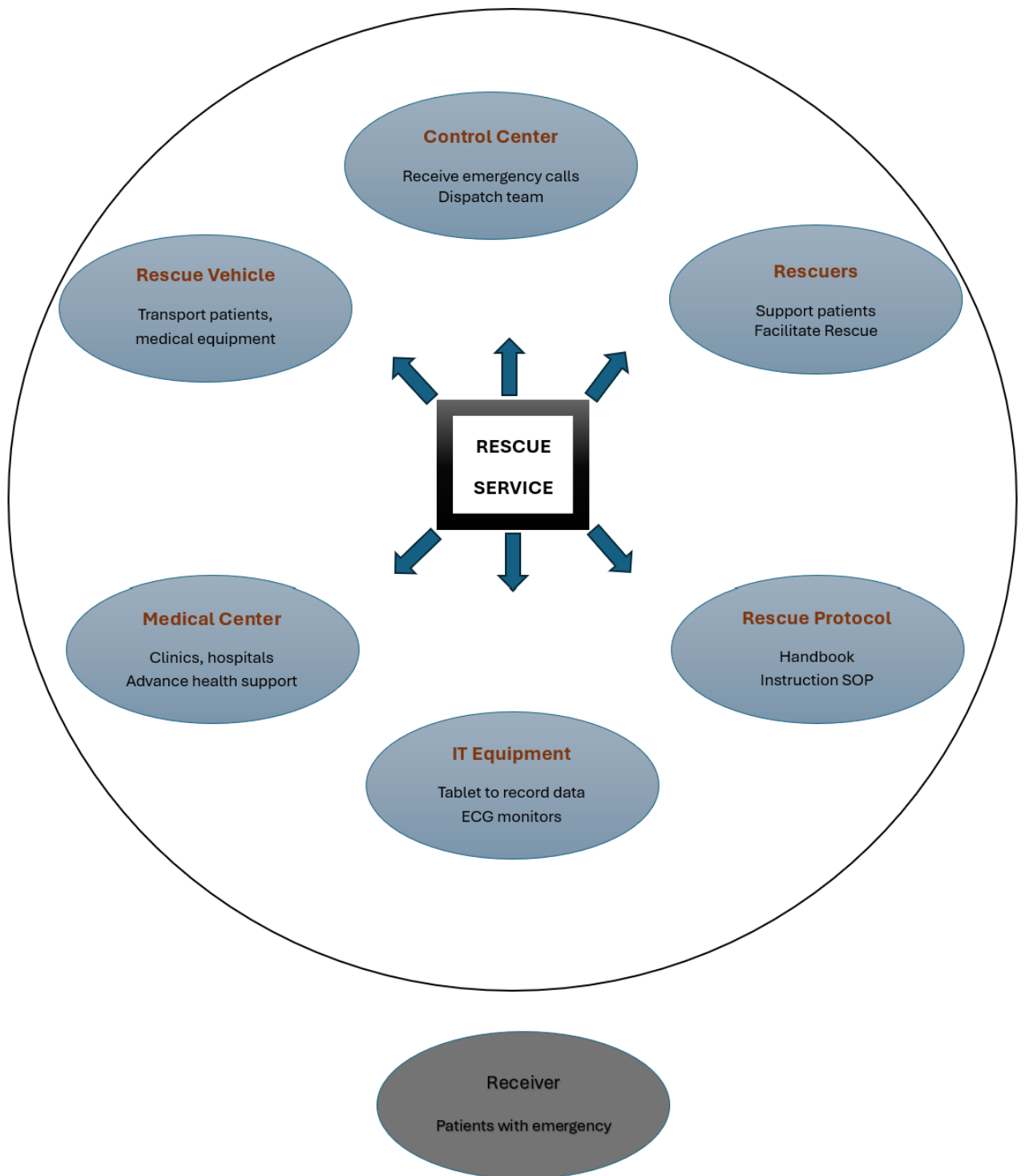


Figure 4.1: Principal Stakeholders in Rescue System

- **Emergency call service:** People use a dedicated emergency number 112 when rescue support is required. It connects them to the regional control center where a dispatcher collects vital information such as patient's location, type of emergency, number of people involved if it is a massive accident. The dispatcher then determines the urgency of the response, generates the required action protocol, and assigns rescue resources.
- **Resource deployment:** Based on the patient's requirement, rescue stations can assign three types of vehicle: Ambulance, emergency medical vehicle, or if required rescue helicopter to bring the patient as soon as possible to medical centers. In addition, vehicles are loaded with the necessary personnel, such as paramedics and emergency doctors, and also with patient monitoring equipment, like electrocardiogram machine, wearable sensors to collect health data, and a capnography device to monitor the level of carbon dioxide in the air. Basic onsite treatment equipment such as defibrillators, infusion pumps, inhalers, and medication kits is also carried.
- **On Arrival Assessment:** Upon arrival, rescue personnel first make an initial assessment of the patient's condition and then collect the necessary information from the patient by attaching body sensors and health questionnaires through different human sources.
- **Treatment:** The interplay between initial assessment and treatment follows a structured protocol within the emergency medical services by rescue stations. Once the seriousness of the emergency is assessed, emergency physicians may be required to perform advanced medical procedures and make critical decisions at the scene. Otherwise, paramedics are trained to provide initial care and first aid. The treatment procedure may include stabilization of vital signs, cardiopulmonary resuscitation (CPR), pain management or medication, immobilization of injuries, and bandage to stop bleeding. Rescuers use a handbook that provides information on the standard treatment pathway for various medical emergencies.
- **Further Transportation:** As rescue vehicles are not equipped with large medical equipment, depending on the patient's condition, he or she may be referred to transfer to a nearby hospital. During transportation, patients are provided continuous medical care by paramedics or emergency physicians. As the process is dynamic, in every step the patient's data is recorded which is handed over to the medical center upon arrival.
- **Post-event Activities:** Once the rescue operation has ended, both the ambulance and the equipment are cleaned and disinfected. In addition, supplies and medications are replenished in preparation for the next mission. A detailed rescue report is generated through a medical pad that contains patient information from the call center to handover.

In the steps described above, the most important part is the diagnosis and treatment recommendation, as it directly impacts the patient’s survival, recovery, and overall outcomes. Currently, many research works are being carried out to improve this process using a medical decision support system which may not only be an additional support to rescue personnel, but will also accelerate the process. The study by Rowland and Adefuye in 2021 [141] revealed that inaccurate diagnoses are frequently cited as the primary reason for claims of malpractice and significantly increase global mortality rates annually. Examining the procedures performed during a mission, it becomes apparent that the sequence of events is generally quite disordered and bound by tight time constraints, which may lead to poor judgment and misinterpretation of clinical signs. The integration of AI in such critical environments may have huge potential, as it has the ability to understand complex data patterns and can provide prediction in milliseconds. In the next section, an updated system model is presented in which AI can be a part of the diagnostic and treatment recommendation steps.

4.2 AI Integration in the Existing Rescue System

Machine learning, a field of AI, works with data and, based on user requirements, can provide predictive analysis, which in a diagnostic system can work as a possibility of having a certain complication. Data recorded from different stakeholders of the rescue system can help rescuers make a diagnosis of the patient in which ML can be an associative component. Below is an overview of the data from different sources that may be utilized by ML models:

- **Control Center:** The first communication source who can provide the initial information about an emergency situation. For medical cases, the control center records the location of the incident, the geographic information of the patient, the existing health status and complaints, etc.
- **Rescuers:** Paramedics and emergency physicians serve as the first responders in the onsite assessment of patients that provide a deeper analysis of the patient’s health. The assessment report contains information on health vitals, the primary impression of the rescuers, patients’ diagnosis, and the treatment path that can be used to train ML models.
- **Health Record:** The health insurance card contains crucial details about the patient’s medical history and any medications currently being used.
- **Patient:** ML models can also use information collected from patients, like location of pain or health abnormality, record of daily activities, incidence time, nutritional habit, and bowel movements. If the patient is unconscious, this information can also be collected from the persons nearby, like friends, family members, neighbors, or passerby.

The way machine learning-based prediction works is that it analyzes historical data and checks their diagnostic outcome to recognize a pattern. As an example, if it was

found in rescue records that patients with chest pain and low consciousness often result in cardiovascular complication, ML models save this pattern for future prediction. Rescue records contain hundreds of information about patients and many of this information have a strong influence on the predictive analysis. The algorithm inside machine learning models calculate the weights for each information grid based on it's relation to past outcomes and then apply them to test data to generate a prediction which in rescue situation can be both complication recognition or medicine recommendation. However, utilizing AI for making medical decisions carries considerable risks. Instead of offering definitive guidance, ML-based decision support tools can deliver probability percentages for various outcomes, potentially guiding rescuers towards improved diagnostic approaches. In Figure 4.2, a AI-based rescue system architecture is drawn which can be employed for better patient management.

A core challenge in building this system is the different modalities of input information recorded from patients. This includes linguistic input from the control center and paramedics, as well as time series of vital signs from medical devices. Therefore, existing machine learning frameworks and architectural approaches cannot be applied directly to the aforementioned problem. Another core challenge is the incomplete and informal information about previous missions in the rescue databases. Therefore, appropriate software tools need to be developed that standardize and formalize the data sets. Fully manual processing is infeasible due to the large number of records. Likewise, fully automated processing is unrealistic. Therefore, interactive tools for semiautomatic processing of the data sets were planned to be developed. Taking into account these challenges, the planned ML development process consists of three main steps: data management, the development of machine learning models, and performance evaluation. Figure 4.3 illustrates the steps in the proposed workflow, including data preprocessing through sorting and filtering, text parsing, feature engineering for building a training dataset, ML model development, and performance evaluation. As mentioned above, the state-of-the-art data processing or ML-based framework is not the right fit to create a high-quality situation detection model. Given the potentially disastrous consequences of inaccurate results, it is obvious to demonstrate high accuracy, sensitivity, and specificity on disease classification. The upcoming chapters in this dissertation extensively explore the innovative techniques and algorithms utilized in this research, including a section that focuses on evaluating performance to demonstrate the results achieved.

4.3 Smart Wearable

Integrating a smart wearable model into the rescue system, specially designed to support rescue personnel, has great potential to improve patient diagnosis and action recommendation. The wearable can help to identify the health situation by evaluating the control center data, the vital signs, and, if necessary, additional input from the rescue service personnel and to output the steps to be taken in accordance with the treatment paths depending on the patient's condition, e.g. the administration of a drug with a dosage adapted to the patient's data. An example of a rescue operation

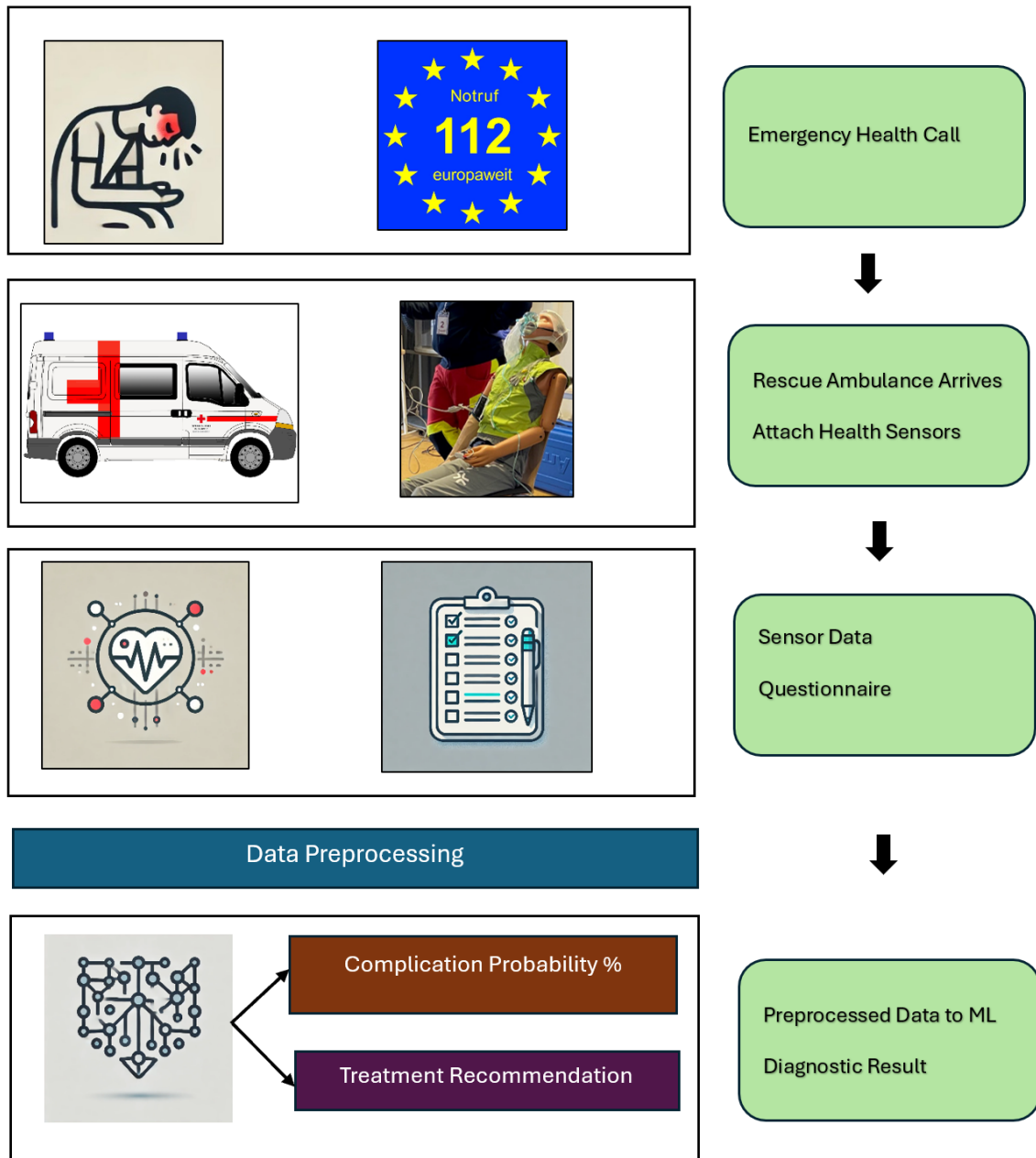


Figure 4.2: Integration of AI in Rescue Activity [35, 132]

involving cardiopulmonary resuscitation can be demonstrated in Figure 4.4. This example is created by integrating the ML-accelerated smart wearable into the rescue system. When a patient is reported to the emergency station, the dispatcher first makes the emergency call, opens the call with the message 'unclear unconsciousness', and dispatches an ambulance and an emergency medical service vehicle. While the vehicles are moving to the scene, another call is made because the patient is no longer breathing and the dispatcher begins telephone resuscitation; at the same time, the

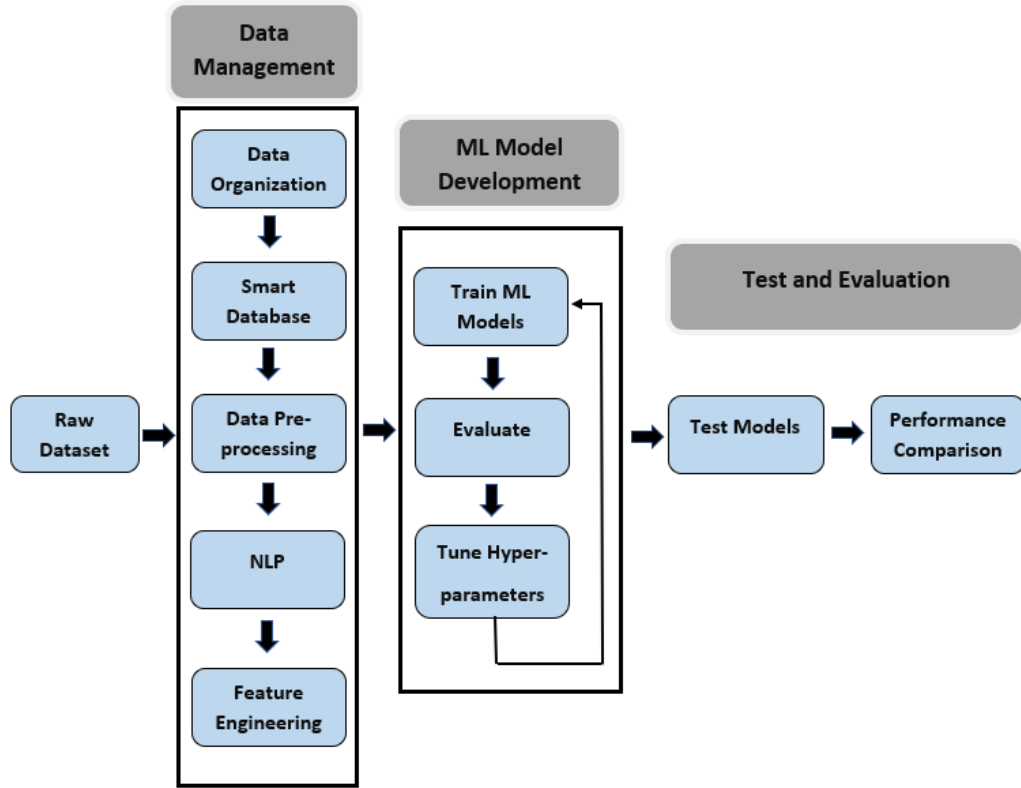


Figure 4.3: Proposed Methodology on Data Analysis and ML Model Development.

dispatcher adjusts the message image to "resuscitation" and sends the mission information to the wearable. With this information, the wearable selects the instructions for resuscitation and displays the necessary steps. The immediate recording of the query information and adaptation of the outputs enables immediate preparation for the expected situation, and thus saves valuable time. Since the system has the alarm data, the otherwise required emergency call is not displayed but provides information on the readiness of the defibrillator, the use of the electrodes, and the thorax compensation by the second helper is shown. The feedback system for thoracic compression provides data on pressure depth, which the wearable then provides with feedback and, if necessary, can also evaluate linguistically, since the rescuer's hands are fixed by the compression itself. Then the automatic analysis by the defibrillator and its evaluation on the defibrillator and on the wearable takes place. Depending on the electrical activity detected, the device indicates the need to initiate defibrillation or to continue chest compressions. Once a ventilation option has been established, these data (e.g., respiratory rate, respiratory volume, oxygen content, and CO₂ content in exhaled air) are also evaluated, and the wearable displays the corresponding recommendations for action.

Modernization of existing rescue systems through the integration of AI-based smart wearables presents a significant opportunity to improve the detection and treatment of complications during emergencies. These smart wearables, wirelessly connected

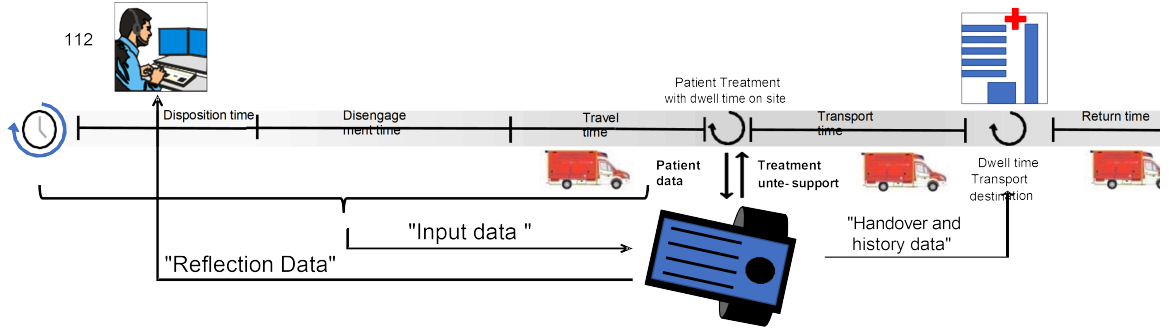


Figure 4.4: Rescue Patient Management with ML Accelerated Smart Wearable Having Interactive Display

with health sensors or rescue equipment, can significantly improve real-time health prognosis and personalized treatment decisions in rescue scenarios. The ability to rapidly track health vitals provides guidance to rescue doctors so that rapid deterioration of health can be prevented. It also allows first responders to prioritize care according to the severity of the patient's condition [114]. This capability is particularly beneficial in chaotic rescue environments where rapid assessments are necessary to allocate resources effectively. Furthermore, the algorithms for recognizing patient situations and generating treatment recommendations examined in the thesis can also be applied to various other applications and health issues. Accepting that, the focus of this doctoral research lies mainly in developing a highly reliable algorithm, adaptable for deployment across any hardware or application software, to aid patients in emergency situations.

5

Rescue Data Analysis and Organization

The digitalization of healthcare has opened up the potential to generate and collect large amounts of biomedical data from a patient using sensors and electronic devices. State-of-the-art data analysis tools and computing devices make it feasible to deepen our knowledge of individual health data, but the availability of such data comes with different challenges for the data researchers, as often it is found that the data are high-dimensional, unstructured, redundant, or ambiguous to be able to draw insight into patient's health. Similar challenges were faced when working with the rescue data set, as the data was often incomplete, redundant, and did not provide meaningful information in raw form. This is not unexpected because during a rescue mission, it is quite challenging to collect data and make a diagnosis in a limited time frame. Hence, noisy artifacts and contaminated information are present in the data set that need to be filtered to extract meaningful information. The effectiveness and precision of ML models are directly influenced by the quality, quantity, and relevance of the data used to train them. This chapter will describe the organization of the raw data received from the rescue station and provide an overview of the pre-processing steps. Throughout rescue operations, data collected from multiple sensor devices are stored in separate files. Therefore, the essential data organization methods are also detailed and need to be implemented to effectively merge and filter this information. The chapter also focuses on the development of a data dictionary that was created to explain each of the filtered features on the rescue dataset.

5.1 Raw Rescue Data

The initial raw rescue data used in this research were collected with the support of partners from the KMU-Innovativ research project-‘KIRETT’. The Siegen-Wittgenstein rescue station provided these data for research purposes, which was recorded from January 2012 to August 2021 in various rescue scenarios in the city of Siegen, Germany. These 10 years of historical databases of previous rescue attempts consist of 273,183 unique cases identified with numerous types of complications and 452 unique attributes. The data recorded were extracted by one of the KIRETT project partners in XML format and later converted to XLSX to facilitate comprehensive data analysis. At first, the data were held in separate files according to their types and subsequently consolidated into one file to centralize all the information. Both quantitative and categorical information was present in the database. Due to privacy issues, the data given were anonymous with a nondisclosure agreement. During data exploitation, the data were categorized into different subgroups as can be seen in Table 5.1.

Table 5.1: Categorization of Rescue Information

Category	Information
Patient information	Patient’s demographic information, height, weight, BMI
Pre-conditions	Patient’s pre symptoms, events before emergency, health complaints, pre illness, medication
Time of incidents	Time of emergency incidents, performed tests, recording events
Basic health state	Basic health condition when observed initially
Consciousness	Responsivity of patients, basic conscious level
Body sensitiveness	Sensitivity condition in arms, legs, and eyes
Blood pressure (BP)	Methods of measuring BP, where measured, capillary refill, systolic and diastolic values
Health vitals	Collected health vitals such as heart rate, glucose value, blood circulation, respiratory rate, etc
Cardiac condition	Possible stroke information, ECG status, arterial pressure value, catheterization status
Respiratory condition	Airway obstacles, lung sounds, breathing status, requirement of intreavenous (i.v.)
Neurological condition	Face, Arms, Speech, and Time (FAST) test result, Glasgow Coma Scale (GCS) assessment, body sensibility, initial neuro-assessment

Continued on next page

Table 5.1: Categorization of Rescue Information (Continued)

Psychiatric condition	Patient’s mental status
Injury status	Possible injury location, current condition, National Advisory Committee for Aeronautics (NACA) score, bleeding condition, severity
Diagnosis	Initial assessment by rescue personnel, illness categorization, verdict if possible
Intervenous(i.v.)	Cannulation, inserted fluid/medication, amount, Oxygen status
Treatment path	Type of medications, amount of dosage, other treatment means
Death record	Log if patient is already dead

5.1.1 Data on Health Vital Information

Health vitals, also known as vital signs, are objective measurements that provide essential information on the physiological status of an individual. These measurements are crucial for assessing a patient’s overall health condition and are often used to diagnose a patient’s overall well-being, detect medical problems, and monitor the progress of treatment. Primary vital signs include body temperature, pulse rate, blood pressure, respiratory rate, and oxygen saturation [122, 40]. In emergency situations, additional vital signs are documented to enhance the depth of diagnosis because traditional vital signs may not be sufficient to recognize all forms of patient deterioration, especially in critically ill patients [40]. The significance of monitoring vital signs in an emergency scenario can be described in the following ways:

- Baseline assessment of patient’s physiological status
- Early detection of diseases such as high blood pressure, infections, or respiratory issues
- Monitor effectiveness of treatment path provided to a patient by continuously detecting deviations
- To assess the severity of health complications for guided emergency treatment
- Help to decide if a patient needs to be transferred to hospital

The vital signs information in rescue situations was generally collected through multiple sensors and portable devices such as Corplus C3, ZOLLX series, LifePak15, etc. available in rescue wagons. These sensor-based devices allow rescue teams to make quick informed decisions, provide accurate care in transit, and communicate vital information to hospital staff before arrival. A short overview of the health vital information found in the rescue record is presented in Table 5.2.

Table 5.2: Common Health Vitals Recorded in Rescue Event

Vital	Information
Pulse rate	Also known as heart rate is the number of time heart beats per minute. Normal rate 60-100 beats/minute
SpO2	Oxygen saturate rate is a measurement of how much oxygen blood is carrying as a percentage of the maximum it could carry. Normal rate >95%
Respiratory rate	Number of breaths taken in a minute. Normal rate 12 to 20 breaths/minute
RRsys	Known as Systolic blood pressure, it measures the pressure blood is pushing against someone's artery walls when the heart beats. Normal rate for adult <120 mmHg
RRdia	Known as Diastolic blood pressure, it measures the pressure in the arteries when the heart rests between beats. Normal rate for adult <80 mmHG
EtCO2	Known as end-tidal carbon dioxide that indicates the level of carbon dioxide released at the end of an exhaled breath. Normal range 35-45 mmHg
Blood glucose	Measure of sugar found in an individual's blood. Normal range 70-100 mg/dL
Body Temp.	Temperature of human body. Normal range 97.7–99.5 °F

5.1.2 Text Data on Rescuer's Impression

During rescue events, rescue personnel record information on their smart devices, such as tablets. The information collected is necessary to deduct the treatment path and subsequent medical care. For the diagnostic analysis of rescue patients, not only is the patient's demographic information or incident details important, but also the primary assessment from rescue workers. The assessment is carried out through investigation of patient pre-illness information, health complaint, visible status, checking level of consciousness, medication history, and many others. For this research, some of these text attributes recorded as the rescuer's impression were a significant source of predictive analysis. Table 5.3 provides a brief overview of the textual attributes of the rescue data considered.

Table 5.3: Text Information Recorded by Rescuers in Rescue Cases

Attribute	Recorded Information
Circulation comment	Initial analysis of blood circulation, skin temperature, heart beat
Pulse rhythm assessment	Regularity or irregularity of pulse
Limbs assessment	Movement pattern and sensitivity of legs and arms on both right and left sides
Pupil assessment	Pupil position and condition, sensitivity
ECG status	Any possible AV block, arrhythmia, abnormal heart beat, heart failure, or if ECG was conducted
Injury	Assessment of injury condition, possible bleeding, size
Consciousness	Glasgow Coma Scale (GCS) values based on verbal, eye, and motor responses. Also a summary of alertness
Complain narratives	Patient’s or nearby people’s comments on patient’s health problems
Anonymous text	Rescuer’s note on patient or event as a summary of event observation
Medication	Provided medication with dose amounts
Pre-illness	Patient’s medical record of previous illness
Psychological assessment	Patient’s mental condition, such as signs of depression, suicidal tendencies, or fear
Neuro assessment	Neurological status based on FAST and GCS tests indicating communication status, drug or alcohol consumption, awareness of surroundings
Respiratory assessment	Patient’s airway condition, respiratory status, possible dyspnoea, thorax trauma, etc.
Rescuer’s impression	Initial health complication assessment
Illness assessment	Main or suspected diagnosis of patient’s health complications

5.1.3 Data on Diagnostic Information

The rescue cases represent the history of rescue operations for different patients suffering from complications such as cardiovascular, respiratory, neurological, psychiatric, abdominal, and digestive. Each of these cases was recorded with rescue parameters in information form, geographic information of the operation, patient medical his-

tory, health diagnosis, vital signs, first impression of the complication, medications administered and treatment performed. The attributes of the health diagnosis were represented in three separate columns with information such as:

- **Type of illness assessment:** If the patient has an injury or suffers from specific health problems
- **Illness type:** Type of illness the patient has, for example if it is a normal fever, breathing problem, heart disease, etc
- **Diagnosis:** This column provides the diagnostic information of a patient, mentioning if it is only a suspicion or a confirmed diagnosis of a disease.

Based on this information, each of the rescue records was labeled with one of the eight major health complications. However, many times the rescue personnel only have to rescue people from unnatural events without any health problems. Sometimes, no information could be recorded due to the patient’s reluctance or environmental issue. In addition, if the patient is already dead, diagnostic information is generally not recorded. Finally, with the available information, 157,677 records were labeled. In Figure 5.1, an overview of patients with different types of complications is shown based on the diagnostic information labeled.

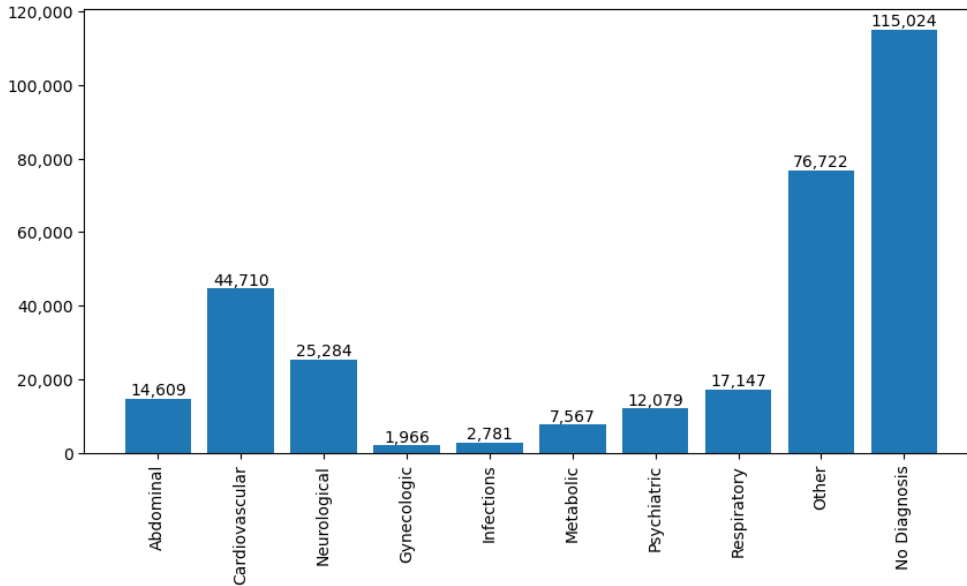


Figure 5.1: Number of rescue patients with different types of health complications as per the 10 years record of the rescue station, Siegen–Wittgenstein. Many rescue events were found without proper diagnostic information, and were thereby uncategorized.

5.2 Data Organization

The data from the rescue station were recorded in an emergency environment where information collection is quite difficult and sometimes gets distorted. Hence, it was

not surprising to find misspelled text information and abnormal values of health vitals like blood glucose, respiratory rate, and pulse rate. Many of the data columns were found without viable information. As a consequence of these data anomalies, it became necessary to implement data organization and management techniques to render the data suitable for machine learning applications. The data organization process was performed in several steps described in the following subsections.

5.2.1 Data Merging

Data Merging is a fundamental process in data management and analysis that involves combining data from different sources into a single unified dataset. This process is particularly important in medical data-based analysis, where patient health information can be distributed in different systems, databases, or formats. The merging of data can improve the clarity and usability of health information, allowing more effective analysis and decision making [11]. The database received through the KIRETT project consisted of 83 CSV files, each containing unique information about complete rescue events, including patients' health assessment. In addition, some records had multiple rows with information from the same patient because the rescue process was dynamically recorded to check how the patient is evolving. Therefore, to further process the data, these files and user information were merged to obtain a holistic view of the patient's medical history, treatments, and results. The key to unify the data set was the case number assigned for each rescue event. The merging process was done by the following steps:

- All the 83 files were stored in a unique folder without changing any content inside them.
- A Python program went through each file, identifying duplicate cases, and then merged them to a single row. If the information is the same in different records, then the particular cell contains only one data point. Otherwise, all different data are stored with a comma separator. Generally, duplicate columns were found to exist only in the files that contain time series information on health symptoms such as respiratory rate, pulse, and SpO2. In that case, the health information with the time stamp was recorded.
- All files were merged using their case ID in one single XLSX file. The reason for using the XLSX format is its user-friendly interface with better built-in tools for data analysis and presentation.

5.2.2 Data Pre-processing

The performance of machine learning models depends mainly on how the data are provided to train the algorithm. Although deep learning may work with raw data by interpolating feature values, it does not ensure good performance as data anomalies and outliers can still be present. Data outliers can only be defined on the basis of the data context and have no definite values. In the field of medical data, sometimes

the patient’s health symptoms can fall outside the average value by many percentages, but it cannot be called an outlier. Rather a deep data analysis is required to understand the type and behavior of the data. That is why for the situation recognition model pipeline, data pre-processing was a crucial step as it transformed the raw data into a flawless and structured format suitable for accurate analysis and model training. Common steps for data preprocessing are: data imputation, data cleaning, data normalization, and data splitting, which was applied to the rescue data set [72] in different phases of the ML pipeline [30, 96]. During the preliminary stage, data preprocessing was conducted to ensure a thorough data analysis, free from any obvious anomalies or missing data points. Another stage of pre-processing was performed during feature engineering steps discussed in the next chapter. The pre-processing task was divided into two stages to align with the context of the individual complication type and avoid a long processing time.

The rescue station data were recorded in an emergency environment often contained misspelled text information and abnormal values of health vitals like blood glucose and respiratory rate. This prompted the necessity to execute data cleaning procedures, converting to data suitable for machine learning applications. The first step was to remove definite outliers, such as a negative value for a health symptom or ambiguous signs or expressions such as quotations and brackets in information columns that had no significance. Many of the rescue records were also not usable as they had no usable information relevant to diagnosis. These records were identified and omitted for final data processing. In addition, there were some columns with duplicate information that were also deleted to shorten data processing times in the next steps. As an example, the Glasgow Comma Scale (GCS) score is calculated by observing the response to eye opening, verbal communication, and motor function of a patient. In the dataset, details of every response were recorded with individual scores. It was decided to work only with the total GCS score that was adequate to indicate the level of consciousness of a patient, thus skipping the scripted scores in different columns for the eye, motor, and verbal response.

Some of the empty cells where the values can be interpolated from other columns were filled with data imputation techniques. To do that, relevant columns with similar information were sorted and then an algorithm was developed to predict the value. Natural language processing techniques were applied to identify words in data columns that can be converted to meaningful attributes by assigning values. A good example is to find the status of the blood circulatory information in the body from the columns that indicates the pulse strength on different hands, the time to refill the capillary, and the condition of the skin. But not all data can be filled up with data imputation, e.g. the medication plan considered for a rescue case. For such cases, due to the unavailability of critical data, rescue cases were omitted during the model development phase for treatment recommendation. The next step was data normalization and standardization techniques, which are used to scale the data to a uniform range or distribution. Normalization typically rescales the data to a range between 0 and 1, while standardization transforms the data to have a mean of 0 and a standard

deviation of 1. These techniques were applied when the features relevant to the individual complications were confirmed to have a common understanding of the scale for each of them. Often, it was found that the health vital values have a different average for different complications, prompted to split the data for individual complication.

Data splitting was done with multiple steps to segregate the data to individual complication as ML models will be developed separately for each complication. This is a horizontal data preprocessing technique in which information on a patient's diagnosis was compiled from different columns to identify diseases. To do that, a list of diseases under each complication was considered as a base for data splitting. The disease list was generated based on the knowledge of medical experts working in the Siegen rescue station. Then a Python program was written that checks if the diagnostic columns of the rescue record contain any of the diseases grouped under a complication. Once a positive match was confirmed, the rescue case was categorized in the column of that specific complication. For example, cardiac arrest was listed under heart disease and asthma as respiratory complication. Some diseases were found to have incorrect spelling or written with synonyms, making it difficult to list a rescue case in a complication group. As a solution, natural language processing tool kits were used to make a comprehensive list of words referring to common diseases. Then each content of this list was looped through in the diagnostic information columns to categorize the complication of the data set. A manual review was also performed to improve the categorization. During the creation of the modified data set, it became critical in many rescue cases, as the diagnosis referred to multiple types of complication. In such a case, the rescue records were placed under multiple list to later verify whether the planned situation recognition model can identify all the complications.

5.2.3 Data Dictionary

A data dictionary is a centralized repository that defines and describes data elements, structures, and relationships within a database. Essentially, it serves as a reference guide for understanding the data used in a system, including metadata that provides context and meaning to the data. In the data organization phase, the creation of a data dictionary for the rescue database was significant because many of the attributes inside it were not understandable for non-medical persons. In addition, some of the information was presented in acronyms that need to be elaborated on for clarity. Understanding the source of the information is also essential for accurately interpreting it in subsequent steps. Developing the data dictionary ensured that all the users of the data involved in different parts of the research were using the accurate interpretation and meaning when entering or querying about the data attribute. Eventually, it reduced the likelihood of errors and inconsistencies that can arise from miscommunication of data or incorrect interpretation [180, 27].

The data dictionary for the rescue database was developed using the Tkinter Python standard Graphical User Interface (GUI) library. According to the Python documentation [50], the Tkinter package is a thin object-oriented layer on top of the Tcl/Tk GUI toolkit that comes bundled with Python for most Unix platforms, including ma-

cOS and Windows systems. It is a user friendly and resourceful tool for developing a data dictionary because of its ease of usage and comprehensive widget set such as buttons, labels, text boxes, menus, toolbars, dialogues, etc. The rescue database dictionary was developed with a total of 340 filtered attributes after pre-processing in previous steps. It was a simple dictionary as the goal was only to create a tool that can define and elaborate the meaning of attributes usable for machine learning. The GUI of the dictionary software is demonstrated in Figure 5.2.

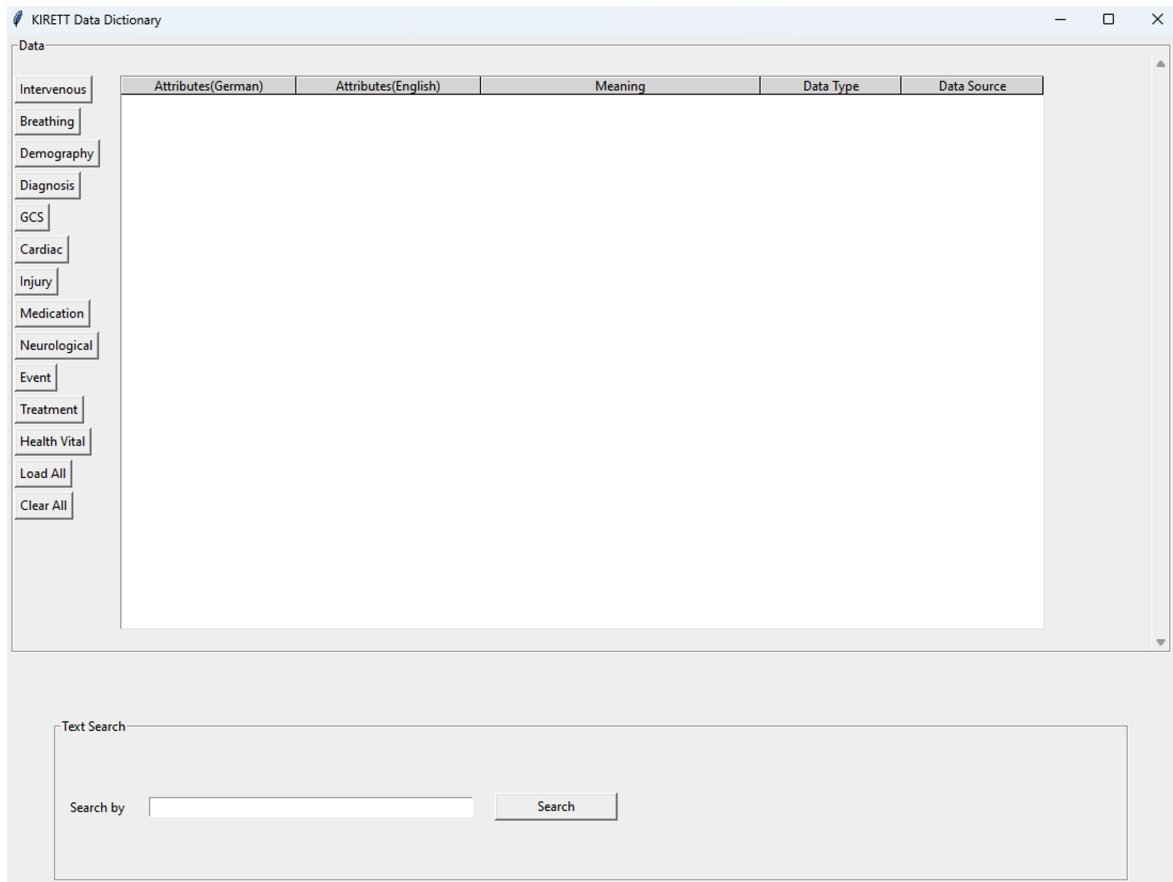


Figure 5.2: GUI of Rescue Data Dictionary

The GUI was designed with buttons representing the categorized information group on the left side. After pressing any of the buttons, the relevant information in that group will be shown. As rescue records were also planned for German users, a separate attribute column was created with German translation. The 'Meaning' column explains the attribute in medical terms. The columns 'Data Type' and 'Data Source' provide an overview of the data characteristics and extraction source. In Figure 5.3, a demonstration is provided on how the GUI looks when a button is pressed (e.g., 'Health Vital'). A search frame was also included in case specific attribute information is inquired.

	Attributes(German)	Attributes(English)	Meaning	Data Type	Data Source
Intervenuous	HR	HR	Heart bit rate	Continuous	Sensor
	OPQRST-SAMPLER Symptom	OPQRST-SAMPLER Symptom	Onset- Provocation-Quality-Radiation-Severity-	Qualitative	Manually by Rescuers
Breathing	Blutzucker	Blood sugar	Measure of Sugar in blood	Continuous	Sensor
Demography	SpO2	SpO2	O2 saturation	Continuous	Sensor
	EtCO2	EtCO2	CO2 concentration	Continuous	Sensor
Diagnosis	Blutzucker ausserhalb Messbe	Blood sugar out of range	Answer if blood sugar out of control	Discrete Binary	Sensor
	EtCO2 Einheit	EtCO2 unit	CO2 concentration	Continuous	Sensor
GCS	Systolischer-BD	Systolic BD	Blood Pressure measurement	Continuous	Sensor
Cardiac	Diastolischer-BD	Diastolic BD	Blood Pressure measurement	Continuous	Sensor
	Puls	Pulse	Pulse rate of patient	Continuous	Sensor
Injury	CO2	CO2	CO2 concentration	Continuous	Sensor
	Temperatur	temperature	Body temperature	Continuous	Sensor
Medication	COHB	COHB	Carboxyhaemoglobin - Indicates CO poisoning	Continuous	Sensor
	METHB	METHB	Methemoglobinemia - can cause blood disorder	Continuous	Sensor
Neurological	Rekapillarisierungszeit > 2 Sek.	Recapillarization time> 2 sec.	Recapillarization	Continuous	Manually by Rescuers
	Temperatur Ort	Temperature place	Body part where temperature was measured	Continuous	Sensor

Figure 5.3: Health Vital Information in Data Dictionary

5.3 Attributes on Health Complications

The rescue personnel follows a specific protocol to collect data from the patients during emergency events. During an emergency call, they are equipped with specialized EMS software applications for real-time data entry and transmission. Many rescue stations utilize mobile applications and electronic health records (EHRs) to facilitate data collection and sharing. These technologies assist in real-time data entry and access to patient information, improving the efficiency and accuracy of collected data [124]. Considering the patient's health condition, the rescue team attaches sensors to the body to monitor health metrics such as pulse rate, oxygen saturation, and blood glucose levels. In addition, they manually assess various bodily signs, including blood circulation, skin condition, and respiratory status. The patient assessment process in each country adheres to a standardized protocol known as CABCADE, as detailed below:

- **Critical bleeding:** Perform examination to find out presence of life threatening bleeding

- **Airway:** Assess if the airway is free or there is a presence of foreign body
- **Breathing:** Evaluate the patient's breathing condition
- **Circulation:** Check for pulse, bleeding condition, and skin color to find out if there is a sign of poor perfusion, internal bleeding
- **Disability:** Assess neurological and altered mental status using tools like the Glasgow Coma Scale (GCS)
- **Exposure:** Expose the patient to identify injuries while preventing hypothermia

After exploring the rescue data received from the rescue station, Siegen, it was found that the data collected from sick patients followed these protocols and had a higher data dimension helping to create a broader and in-depth patient assessment. The high-dimensional data were significant for the situation recognition modeling as it gave scope to choose meaningful attributes for individual complications. In the following subsections, the data categorized for each major complication is discussed.

5.3.1 Heart Complication

Heart complications, also known as cardiovascular complications, refer to a variety of health disorders that affect the heart and blood vessels, often arising as a consequence of other medical conditions, lifestyle factors, or surgical interventions, and can lead to serious health problems or even life-threatening situations. Individuals experiencing this complication pose substantial challenges during rescue operations, where rapid diagnosis and prompt action are vital to avoid life-threatening consequences. The most common heart diseases found in the rescue record are: Myocardial infarction, angina, arrhythmia, cardiogenic shock, syncope, pulmonary edema, thrombosis, hypertension, venous, and arterial occlusion. In rescue records, the number of patients with heart disease was the highest.

5.3.2 Respiratory Complication

Respiratory complication refers to disorders that affect the respiratory system, including the lungs and airways, leading to blocked breathing and oxygenation. These complications can greatly impact an individual's ability to breathe effectively, leading to reduced oxygen intake and impaired removal of carbon dioxide from the body. If the patient stays in such an exacerbating condition for a long time, it can be life threatening. Rescue records show that common respiratory diseases are asthma attack, dyspnea, COPD, pneumonia, bronchitis, hemoptysis, pseudo-croup, and epiglottitis. Respiratory complications were the second most common type found in rescue records and required rescue personnel to take certain measurements to check the condition of the airway of a patient, such as inspecting possible obstacles in the airway, listening for breathing sounds and observing chest movement while checking if the patient is exhaling air.

Table 5.4: Diseases Under Neurological Complication.

Nr.	Diseases
1	Transient ischemic attack
2	Intracranial bleeding
3	Subarachnoid hemorrhage
4	Seizure
5	Febrile seizure
6	Status epilepticus
7	Apoplexy
8	Intercostal neuralgia
9	Retrograde amnesia (after trauma)
10	Retrograde amnesia (without trauma)
11	Neurogenic shock

5.3.3 Neurological Complication

Neurological complications refer to diseases that can affect the nervous system of a patient due to various underlying conditions, infections, or injuries. The nervous system of a person is consisting of the brain, spinal cord, and peripheral nerves which basically controls the body function, establishing a messaging system from various parts of the human body to the brain. An adverse effect on the nervous system means that it will damage not only the movements of the body, but also the intelligent functions of humans, such as thoughts, memorizing, learning, and feelings. Neurological complications-related diseases can range from mild and temporary problems to severe and long-lasting impairments, significantly affecting an individual's quality of life. Common neurological diseases found in rescue records can be found in Table 5.4.

5.3.4 Psychiatric Complication

Psychiatric diseases are related to the mental health of a person who can be affected by various medical conditions, treatments, or life circumstances. In the current world, mental illness is a highly discussed topic, as the symptoms can be invisible to other people, but the effect on patients can be severe. A significant number of people commit suicide due to the overburdening effect of mental illness. The multidimensional impact of severe mental illness includes physical health problems like intolerable headache, sleeplessness, lethargy, psychological difficulties, and socioeconomic drift [47]. Among various diseases, the rescue record shows that common mental illnesses are depression, mania, alcoholic intoxication, insanity due to drugs, suicidal

tendencies, high agitation, and psychosis.

5.3.5 Metabolic Complication

Metabolic diseases occur due to disruption or abnormalities in the metabolic process of the human body, often associated with conditions such as obesity, diabetes, and metabolic syndrome. Metabolism is an important body function that helps break down food through catabolism into simpler molecules such as glucose, and generate energy to perform different physiological functions [171]. Another important function is anabolism, which helps in building cellular structures, synthesizing hormones, and producing essential biomolecules such as proteins and nucleic acids [159]. Metabolic complication can lead to several other life threatening factors such as cardiovascular disease, kidney dysfunction, high blood pressure, etc. Common metabolic diseases found in rescue records are hypoglycemia, hyperglycemia, desiccosis, and urmia.

5.3.6 Abdominal Complication

Abdominal complications refer to a wide range of health issues that have an acute effect on organs and structures within the abdominal cavity. These complications can arise from various causes, including surgeries, infections, chronic diseases, trauma, or congenital conditions. They may appear suddenly or over a long duration and their intensity can vary from minor discomfort to life-threatening critical situations. In rescue records, common abdominal complications found are gastrointestinal bleeding, biliary colic, renal colic, urinary infection, appendicitis, pancreatitis, and peritonitis. A very limited number of rescue records were found with patients having infectious diseases, but could not be classified into the above-mentioned groups. Infection-relevant complications are generally caused by pathogenic microorganisms, such as bacteria, viruses, fungi, or parasites, entering and multiplying within the body. They can affect virtually any part of the body and vary widely in severity, from mild and self-limiting to severe and life-threatening. The common infections reported to the rescue station are fever due to infection, meningitis-encephalitis, infectious gastroenteritis, infection with multiresistant pathogens, HIV, hepatitis and Ebola. Some gynecological patients were also found who had disease of their reproductive system. Many of these complications are particularly relevant to pregnancy, which requires the involvement of emergency responders when a woman approaches childbirth. Some other complications found in the rescue record are eclampsia, vaginal bleeding, and bladder spurting.

5.4 Database Management System

Database management system refers to the systematic approach to storing, organizing, manipulating, and managing data in an efficient way so that it is secure and the user can retrieve, update, or maintain it anytime throughout its lifecycle. Generally, interactive software is used that serves as an intermediary between users and databases, facilitating efficient data handling, ensuring data integrity, and allowing

authorized entry. In the context of patient health-relevant rescue data, effective database management is quite significant due to the vast amounts of sensitive medical data generated and used for patient care, research, and administrative purposes. The rescue database contained over 273,000 rescue entries and more than 400 distinct attributes, which must be handled correctly to ensure that end users can easily access the requested data for advanced data analysis. In the future, there are also possibilities to integrate more rescue records. That is why a robust, interoperable and scalable open-source object relational database management system (ORDBMS) was used which stores all the records, allowing multiple users to simultaneously retrieve information about any rescue event by simply providing the event ID. ORDBMS is used because of its ability to query complex applications, preserve the original data structure, and handle large and complex applications, which is a requirement in the management of rescue databases [158]. This database management system can serve as a subsidiary for practically all SQL operations. It also has a reliable architecture with dependability, data integrity, and precision [74].

To perform the SQL operations, PostgreSQL was chosen, which is an open source relational database management system known for its robustness, extensibility, and support for advanced data types and performance optimization features. It exhibits a higher performance quality when the environment requires a precise and structured data model [79]. The major reasons for using PostgreSQL can be described as below:

- Open source and can be modified easily by end users when required
- Supports a wide range of extensions to enhance functionality
- Supports multiple data types such as JSON, XML, and arrays
- Allows multiple transactions to occur simultaneously without locking the database

Although there are several advantages of PostgreSQL, it also has some negative points due to its complexity for a new user who has no experience in database administration. Also, during usage, it was found to be quite resource intensive, requiring more memory and processing power compared to simpler database systems. Saving data after storing was often slower, but can be compensated for by using on a high-performance server.

5.4.1 Software Development

After refining the raw rescue data, the next goal was to create an interface for users to interact with this database. The first step was to migrate the data to a PostgreSQL database using the client software pgAdmin 4. PgAdmin is considered the most popular and feature-enriched open-source administration development tool with a supported graphics user interface (GUI). The GUI can also be used to interact with the PostgreSQL database sessions, both locally and on remote servers. However, direct access through SQL queries using the GUI in pgAdmin 4 was not user-friendly, especially for those without technical expertise. In addition, when required to perform a complete visualization or analysis of the rescue record based on a single patient, the

GUI was unable to do it effectively. The reason was that when the rescue data set was transferred to PostgreSQL, it was grouped into multiple query-enabled tables based on the relevance of the data attributes. It was necessary as the number of attributes was large and cannot be stored in a single table if organized data analysis is required. So, when a record is searched, not all data can be displayed together as they were saved in different tables. To address this, a customized graphical user interface (GUI) was developed using JavaScript and Angular. Angular’s modular and component-based structure enhances code reusability and clarity, with each component consisting of HTML, CSS, and Typescript files to perform specific tasks. For the back end, Node.js (Node JavaScript) was used, an open-source, cross-platform JavaScript runtime environment that allows developers to execute JavaScript code server-side, and Express.js, a minimal and flexible framework for Node.js that simplifies the development of web applications. An application programming interface (API) using the graph query language (GraphQL) was also implemented, as its flexibility allows users to request only the specific data they need, which minimizes unnecessary data transfers to make queries faster. To fetch, cache and modify the application data from the GraphQL server and to build a complex and reactive user interface from the front-end Angular framework, Apollo and the HTTP client were used. The final application architecture is shown in Figure 5.4, offering an interface for users to access patient information, thus optimizing data management for rescue operations.

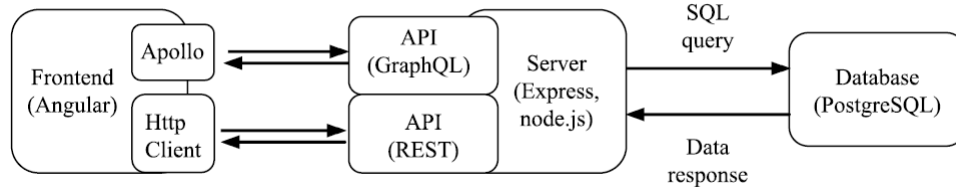


Figure 5.4: Software Architecture of Database Management Software Showing Features Relevant to Central Nervous System (CNS) or Neurological Disease

The graphical user interface facilitates enhanced inquiries about health complication groups and patient-based data analysis extracted from the data set stored on the PostgreSQL server. At the software runtime, the interface presents the main complication categories, as can be seen in Figure 5.5, allowing users to select specific categories to view all diseases related to this complication found in the rescue record. To inquire about a specific rescue event, a search function is also integrated in the GUI, where the user only needs to provide the unique case ID to visualize all the health information of a rescue patient. Figure 5.6 represents an example of a rescue case in which the event was searched with its ID.

Category Selection

Lung disease

CNS disease

Cardiovascular disease

Respiratory disease

Abdominal disease

Psychiatric disease

Metabolic disease

Gyn-obstetrics emergencies

Other diseases

Infections

CNS disease

Relevant features:

- Pupil shape
- Movement of the extremities
- Neurological status and comments
- Stroke status
- Age
- Chief complaint narrative
- Pre-existing conditions
- Pupil reflex to light
- Arm and leg sensibility
- Neurological diagnoses
- GCS score
- Diagnosis
- Risk factors
- Body temperature

Download

Figure 5.5: GUI Displaying Major Health Complications

KIRETT Data Explorer

Complication based selection

Case search

Patient information

Case ID: f2472a60-4f2a-4263-afec-1a09fc64c860

Diagnosis: No diagnosis

Deceased at any point?: Yes

Gender: Male

Age: 37

GCS summary: 12(3/4/5)

Download

Download for this diagnosis

- MENTAL CONDITION -

Inpatient total GCS score: 12	Total GCS score at destination: 3
Intoxicated (drugs): ✓	Intoxicated (alcohol): ✓
Psychological status: None noted	Unconscious: ✗
Drowsy: ✗	Communication difficult / not possible: ✗
Summary: Abnormal	

Figure 5.6: GUI Displaying Patient's Record

5.5 Data on Treatment Path in Rescue Events

The most important step in the rescue operation, specifically in health emergencies, is to establish a treatment path that follows a structured approach to ensure the best

possible outcomes for the patient’s survival. A treatment path outlines the protocol that rescue personnel should follow to assess, stabilize, and, if necessary, take steps for hospital transportation. If the necessary treatment protocol in medical emergencies is not followed in a timely manner, it can have profound consequences on patient outcomes, often leading to increased morbidity and mortality. Research on delayed treatment in medical emergencies like myocardial infarction [175] or acute ischaemic stroke [41] highlighted that every minute of delay in treatment can increase the risk of morbidity of the patient. Based on rescue situations analyzed from Siegen, Germany, it was found that although there are certain limitations on providing adequate treatment for critically ill patients, the ambulance used in such situations is always equipped with urgent medical equipment such as electrocardiogram tracker, defibrillator, vital health monitor, and sphygmomanometer. Several crucial medications such as Aspirin, Epinephrine, Diazepam, Midazolam, and others are maintained similarly in the ambulance inventory and are administered by emergency physicians or medically certified personnel. Each rescue record contains health emergency attributes that indicate the path of treatment and specially administered medication. An overview of these attributes can be found in Appendix A.

5.5.1 Medications

The medications used in rescue situations are generally common over-the-counter drugs that help stabilize patients and address immediate life-threatening conditions until they can receive full medical care. Paramedics or rescue personnel administer medications stored in their vehicle, after a patient diagnosis is performed. The common causes for which medications were used according to the rescue record can be seen in the following.

- Pain management
- Cardiac emergencies if heart attack is probable or patient already had a heart attack diagnosed
- Breathing support if patient’s airway is blocked or for some medical reasons, patient cannot breath
- Mitigate allergic reactions
- Blood sugar management supplement if the patient is falling into hypoglycemia
- Control seizure attack to avoid further neurological damage
- Intoxication support due to drug overdose
- Infection or sepsis shock
- Other emergency situations like stop bleeding from injury, acidity pacification, etc.

Generally, the goal of these medications is rapid onset to provide relief or stabilization as fast as possible. Sometimes, patients require fluids and electrolytes to help with dehydration administered intravenously, intramuscularly, inhaled or through intranasal routes, depending on the situation. Before administering medications, the patient's medical history should be carefully checked because the chaotic nature of emergencies increases the risk of medication errors, including incorrect dosages or incorrect medications being administered. Details of the most common medications for the four main health complications in the rescue record can be found in Table 5.5.

Name	Usage
Analgesia	Used for relieving pain, effective for chest pain
Crystalloid	Used as fluid for dehydration, electrolyte imbalance correction
Antihypertensive	To manage high blood pressure
Bronchodilation	Used in breathing problem, can relax muscles around lungs
Antiemetic	Used to prevent and treat nausea and vomiting
Sedatives	To keep patients calm, relaxed, or induce sleep

Table 5.5: Common Medications used in Rescue Situations

6

Feature Engineering

Choosing the right features is important to create an efficient machine learning model. The more meaningful and relevant the features are, the better the performance of the ML model is. That is why feature engineering is highly significant in the ML pipeline and needs to be designed in a way so that all significant attributes can be extracted, transformed, interpolated, or extrapolated from the data set. The concept of feature engineering lies in the process of using domain knowledge to select, modify, or create features that can highlight important patterns and relationships in the data. To effectively handle features, one must have a thorough understanding of the dataset, the objectives of data processing, and the functionality of individual machine learning algorithms. In this doctoral research, the aim was to create multiple ML models to recognize individual health complications and predict the path of treatment with different feature sets. So, the processes of creating features did not follow a unique pattern. Rather separate strategies have been considered for different complications and predicting the correct medications. The common strategies followed are as follows:

- Handling missing features by data imputation
- Encoding categorical variables by label or target encoding
- Scaling and normalization
- Binning variables into diverse categories to create discrete features
- Explore different NLP techniques to convert textual features to categorical ones
- Create domain specific features

- Reduce features based on their importance score

This chapter provides a comprehensive overview of the feature engineering steps taken to develop training data for all machine learning models in this thesis.

6.1 Feature Identification

Identifying the right features for training the machine learning model is critical to building a robust and high-precision health complication prediction model because the features directly contribute to the model's ability to learn patterns and make correct predictions. Generally, features in the training data for a machine learning model are chosen from existing raw data with necessary modifications. When it is relevant to health complications, identifying the correct features also requires medical expertise to understand the signs and symptoms of patients related to complications. In this study, before selecting the features of each individual complication in the filtered data set, the necessary knowledge was acquired by exploring previous research works and discussing with medical experts. Subsequently, it was found that many of the features required to identify health complications are relevant for lab tests like pathogen, blood, or X-ray. However, since such health examinations are not carried out by rescue personnel due to resource constraints, only attributes available in the rescue database had to be considered for the situation recognition models, which certainly was not extensive. Initially, the features determined were mainly the health vitals available in the database and were decided with the empirical knowledge of previous biomedical studies. Some of the medical experts and rescue station employees were also consulted about complications to validate the initial selection. Some features were not directly available in the dataset but were created using one hot encoding. Certain features, such as respiratory rate and oxygen saturation, were consistently identified in all health complications. Psychiatric complications were not influenced by health vitals and therefore were omitted in the training phase of the prediction models.

Features like circulation cannot be extracted directly from the rescue database, but were interpolated from multiple columns describing pulse strength, bleeding condition, capillary refill time, and skin color. All features were converted to numerical values for the compatibility of the ML algorithm, because ML algorithms rely on mathematical computations (e.g., distance calculations, gradient descent) that require numerical input [83]. In addition, algorithms like decision trees, support vector machines, linear regression, and neural networks are designed to process numerical matrices or vectors to learn patterns from data.

6.2 Feature Generation from Text Data

Identifying health complications with limited features like health vitals is challenging, especially when dealing with complex diseases. Sometimes an individual unexpectedly falls ill with no apparent physical signs, despite their bodily vital signs appearing normal. The rescue database was useful in this case, as it does not contain only patient health information but also textual remarks that indicate historical health

information, relevant complaints, or the evaluation of rescue personnel. The columns that were considered important text data are:

- Anonymous data - This data represents initial impression of rescuers on the event containing information of how the patient was found, level of consciousness, how bad the injury is if available, any major health condition reported by patient.
- Complaints - Information on the health complaints done by patients on different topics such as the type of complaint, where it started, how long it persists, any recent change, which area or time of the day it became severe is recorded in this text data.
- Assessment Text - These text data contain textual information written by rescue personnel on different types of patient assessments, e.g., if a patient was recommended for a certain treatment path by rescue physicians, if specific health patterns are normal, if patients should be transferred to hospital.

To extract information from those columns, natural language processing, a subset of artificial intelligence, was used. NLP is known as a collection of computational techniques for automatic analysis and representation of human language that focuses on the interaction between computers and human (natural) languages. It encompasses a range of computational techniques that aim to enable machines to analyze, interpret, and generate human language in a valuable way. Notably, it proves to be highly effective when the complexity of a set of textual data makes it challenging for humans to extract the knowledge within a specific time frame [33]. For an unstructured database like rescue records, NLP techniques play an important role in handling raw texts and extracting semantic meaning to understand the relevant context and intent. As discussed in the previous section, the quantitative features present in the database were not enough to model situation recognition models for neurological, abdominal, or psychiatric complications. To better understand the patient's health status, it was essential to gain insight into how the body responds to specific complications. NLP played its role on generating new features by parsing the text columns, identifying mainly repeated words, and checking if certain words related to a complication are present in the columns. Text parsing, a process belonging to NLP, was used for that purpose, which involves analyzing a sequence of words or symbols in a text to understand its grammatical structure and extract meaningful information. Parsing transforms unstructured text into a structured format that machines can easily process, allowing various downstream applications such as machine translation, information extraction, sentiment analysis, and more [84]. The technique was applied separately for individual health situation recognition models to group relevant words under selected categories explaining that complication. This is called semantic analysis of text data to extract information from it and structure it for further NLP applications. The whole text parsing pipeline for an individual complication can be explained as follows:

- At first, all text columns relevant to the complication of the rescue database were merged into a single file. This was done for all 273,183 rescue cases.
- A program called 'WordCounter' was written that uses the text data file and checks how many times a word appeared in that file. The Natural Language Toolkit (NLTK) library was used to design this algorithm which covers both symbolic and statistical natural language processing, and is interfaced to annotated corpora [25]. It is one of the most popular platforms in the Python environment that allows users to access a wide range of functionalities through a consistent interface. Common tools and processes that come with NLTK are tokenization, parsing, classification, stemming, tagging, and semantic reasoning. [46]. The rescue text contains many repeated regular expressions and ambiguous words such as patient, since, and found, which need to be avoided. So, a stop-word dictionary was created that helped ignoring those words during the word counting process. Finally, the words most encountered were saved in an XLSX file and manually checked again to enlist the meaningful keywords.
- The initially selected keywords were then grouped into different categories where the categories are nothing but certain health conditions common in a health complication. Each category is equivalent to a custom feature created from the text data. The list of these interpolated features for each complication can be found in Appendix C.
- A Python-based algorithm was developed to parse all text data from rescue cases to identify a match for the specified keywords. If the text data of a rescue case contains one of the keywords, then its corresponding feature value was assigned to the matched keyword text in the feature column. It also meant that the feature was relevant for the rescue case. If no match was found, the feature value of the corresponding column was kept as none.
- The appearance of a certain keyword does not ensure that the rescue case has relevance to that keyword, as negation words can also be present before the word in text columns to null out the possibility of any relevance. That is why the parsing algorithm was also included with the exceptions and assign keywords only if no negation before the word is present. As an example, if a certain health condition like 'A' is considered as a keyword, then the following situations can happen when parsing through the text database:

Case 1: the patient shows symptoms A - the rescue case is acknowledged with the presence of the keyword and the corresponding feature value is assigned to 'A'.

Case 2: The patient shows **no** A symptoms - the rescue case is not acknowledged with the presence of the keyword, and the corresponding feature value is kept empty.

This process is repeated for all complications to find an initial record of rescue cases that have relevant keywords related to the specific complications. To fully under-

stand the process, an example can be provided considering psychiatric complications. For the psychiatric complication, the number of keywords initially selected was 69 and were grouped under: preillness, drug intoxication, abnormal characteristics, psychiatric condition, alcoholism, current condition, and possibility of death. Certain keywords were assigned to each category, such as suicidal, and depression was assigned to pre-illness, panic was assigned to abnormal characteristics. Then, the text database was parsed to check the presence of these specific keywords. Suppose that in rescue case 1, the keyword depression was found, and in rescue case 2 panic and suicidal were found. Then the updated list will look like this:

Case	Current Condition	Psy. Condition	Abnormal Character	PreIllness	Alcoholism	Drug Intox	Posibility of Death
Case 1				depression			
Case 2			panic	suicidal			

Finally, when a feature value contained a keyword, it was replaced by 1, otherwise 0. Converting it to binary numbers was necessary as the classical machine learning algorithms used will only work with numerical data. To apply this feature collection technique in real rescue events, a questionnaire was created that represents the features extracted from the text. In rescue events, rescue personnel can use this questionnaire to record information that will help to collect the value of the features as 'yes' or 'no'.

6.3 Feature Selection and Elimination

Initially chosen features comprise health vitals and text data supported by NLP, which require further pre-processing for the final feature selection. The primary objectives of the feature selection were to improve model performance, reduce overfitting, decrease training time, and improve model interpretability. In rescue modeling, it is not only about selecting the right features that are relevant to a health complication, but also removing features that are related to multiple complications. Because those common features will confuse the model during the training phase and will provide incorrect prediction results. In addition, eliminating unnecessary features will simplify the model, improve generalization capabilities, and reduce the computational burden associated with training and inference, as fewer features mean less complexity in the model [127].

There are three popular filter selection methods used by researchers: filter methods, wrapper methods, and embedded methods. Among them, the filter and wrapper method were selected which are occupied in two stages of feature selection: pre-training and post-training. Before developing the ML model, the filter method is used to evaluate the relevance of features by their intrinsic properties, independently of any machine learning algorithm. An Excel-based validation algorithm was developed that evaluates the feature relevance score to estimate whether a keyword is more relevant to the specific complication than to other complications or no complication. To accomplish this, an average was calculated for each feature, representing the mean

count of occurrences of that feature in both patients and non-patients for a particular complication. Then, a relative deviation was measured between these two averages to find the relevance score. Relative deviation measures the size of the deviation between two values, expressed as a fraction or percentage of one selected value. It helps to understand how significant the deviation is relative to the size of that reference value. Here, the selected value was the average appearance value for the patient. The equation for relative deviation is as follows:

$$\text{Relative Deviation} = \frac{|x - y|}{|y|} \times 100$$

A small relative deviation indicates that the observed value is close to the reference value, while a large relative deviation indicates that the observed value has a high variance from the reference value. After observing these deviation values, for each individual complication, different threshold scores were decided to finally select a feature. For instance, if the validation score for a feature meets or surpasses the threshold score e.g. 6, it indicates that the feature occurs six times more frequently in rescue events for a patient compared to non-patients or patients with different illnesses.

The wrapper method was applied during model training that evaluated subsets of features by training a model on them and assessing the model’s performance. This approach can lead to better feature selections since it calculates the interaction between features. However, it was found that when this method was applied, it was computationally quite expensive, especially when the training data set was large, due to its multiple iterations of model training [169]. One of the techniques used to incorporate this method was recursive feature elimination with cross-validation (RFECV). It is an advanced feature selection technique that combines recursive feature elimination with cross-validation to select the best subset of features. RFECV automatically optimizes the number of features to keep by evaluating the model’s performance with different feature subsets. But for some models like support vector machine with radial basis function kernel, k-nearest neighbors, or naive Bayes, this method will not work. Because these models do not provide inherent feature importance metrics like `coef_` (used in linear models) or `feature_importances_`. This lack of a feature importance score makes it impossible for RFECV to assess which features to eliminate or retain during the recursive elimination process. Even when the importance getter attribute is selected as ‘auto’ in Python, RFECV expects the estimator to have a method of determining feature importance. Unfortunately, many models do not have intrinsic properties like that. That is why another technique called Sequential Feature Selector (SFS) is used, which is a popular feature selection method developed by scikit-learn that helps identify the most important features for a machine learning model through a sequential approach: forward selection and backward elimination. When the feature set and a base model are passed to the algorithm, it selects or removes features one at a time sequentially, making it an iterative process. This method does not require any feature importance score as it calculates the importance of a feature itself through sequential processing. In our research, RFECV was found to be faster than SFS because it works only with the elimination of backward features, but SFS was easy

to handle and offers more flexibility with model selection. Utilizing cross-validation for both methods guaranteed that the selected features functioned well with new datasets, thus preventing the model from overfitting to a particular feature set.

6.4 Outlier Detection and Removal

In Section 5.2.2, removal of definite outliers from the rescue database was discussed. Those outliers consisted of mostly ambiguous signs, e.g., commas, brackets, and negative values that carry no significance in data analysis. But after confirming the required feature, another type of outliers need to be eliminated, which can deteriorate the model's performance because of its extreme value. Those outliers can be defined as data points that deviate significantly from the majority of the feature data and can lead to overfitting, poor generalization, and inaccurate predictions. A good example may be the distribution of respiratory rate data for patients with a cardiovascular complication depicted in Figure 6.1.

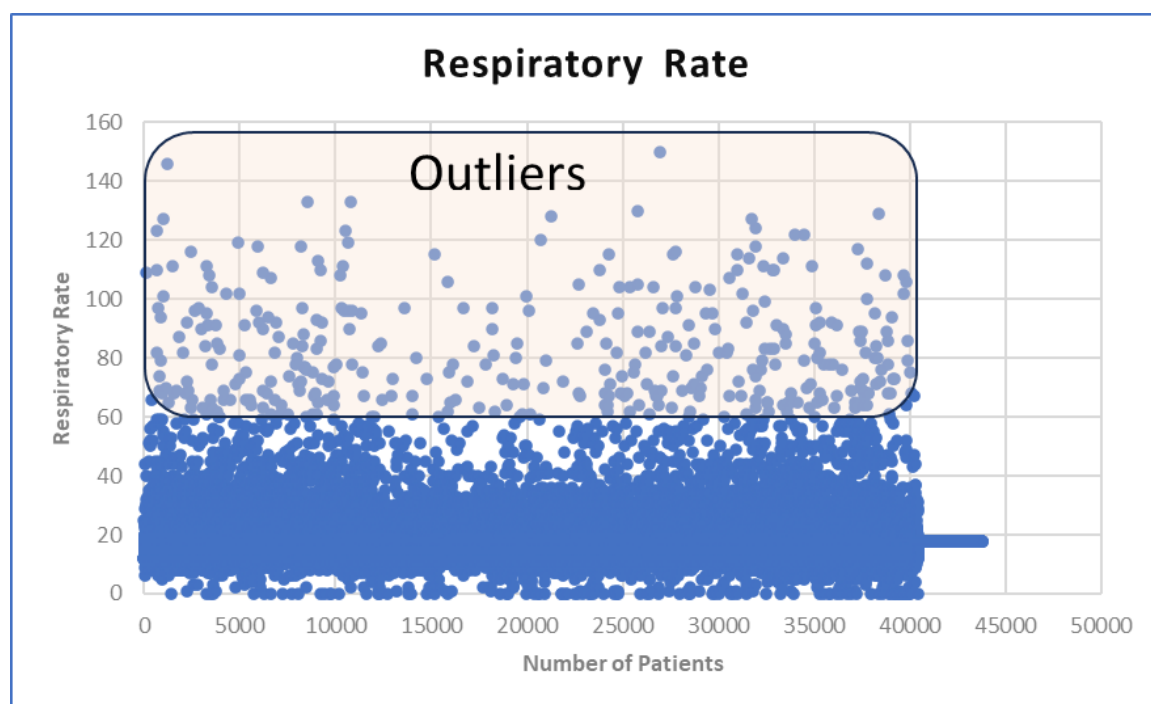


Figure 6.1: Outlying Respiratory Rate Values in Cardiovascular Complication

As can be seen in the figure, most heart patients had a respiratory rate in the range of 5-35, but some records indicate a respiratory rate much higher than or below that range. The normal respiratory rate for adults and children is in the range of 12-20 and 25-40, respectively. People during exercise may have an elevated respiratory rate between 40-60. But the scattered graph shows that some people have respiratory rate more than 80 which were considered outliers. The existence of such outliers in ML models significantly skews the results and leads to misleading interpretations. Various methods have been developed to identify and handle outliers, among which

the Interquartile Range (IQR) was adapted for outlier elimination. This method calculates the IQR, which is the difference between the 75th percentile (Q3) and the 25th percentile (Q1). Outliers are identified as data points that lie outside 1.5 times the IQR of the lower or upper quartiles. While using the IQR method to filter the data, the method was modified to avoid too many data deletions. For complications like cardiovascular or respiratory complications, if the IQR method is applied, a significant number of cases were eliminated. This is not practical considering that rescue cases deal with patients who may have out-of-range health vital values. For example, in diabetic patients, the blood sugar level can even be up to 500 mg / dl, which puts this value in Q4. So, the method was modified by considering that the characteristic values should be between 10% and 90%. A Python program was designed, which first takes all the rescue cases with selected features. Each of the features representing health vitals were considered sequentially and, upon finding an outlier completely removed. Following this method, fewer rescue cases were deleted and the data became noise-free.

Machine Learning in Rescue Event

Machine learning, a subset of AI, has become a strong transformative force in the healthcare industry, revolutionizing the way medical professionals diagnose health complications, predict treatment requirements, and optimize operational efficiencies in emergency care. The strength of ML algorithms lies in their ability to work with a large amount of data and find patterns within. This chapter will investigate machine learning in rescue operations and subsequently outline the implementation pipeline for ML development carried out in this thesis.

7.1 Advantages of ML in Rescue Situations

In emergency medical situations, like rescue events, a patient's health check is not just a check for common health vitals. In more intricate situations, patients require diagnosis from multiple perspectives and with foresight to determine whether there is simply one straightforward issue or if multiple complications are arising simultaneously. For humans, looking at all those health aspects and making an accurate judgment is difficult. This is a scenario where machine learning, leveraging its robust computational capabilities, can effectively evaluate a patient and offer additional guidance to ensure his recovery. In healthcare, machine learning can offer certain benefits as follows:

- **Early Detection and Diagnosis:** Machine learning algorithms can identify subtle patterns and anomalies in medical data that might be missed by the human eye, leading to earlier detection of diseases or complications. Some diseases which are indicated by multiple symptoms can be assessed more reliably with ML based prediction models, which is particularly significant for rescue cases. Since medical professionals are not consistently available to evaluate all symptoms when rescuers attend to patients, ML predictive models can be of sig-

nificant assistance in these situations. Moreover, given the frequent monitoring of health vitals to track the patient's condition in real-time, ML algorithms can effectively analyze patterns and offer updated information about the patient.

- **Dynamic Patient Monitoring:** In dynamic rescue scenarios, it is essential to continuously evaluate patients to determine their current treatment needs. Sensors attached to patients consistently capture health vital data, allowing ongoing monitoring of their condition. By incorporating this data into machine learning algorithms, it becomes possible to efficiently follow evolving patterns and provide insights into the patient's health condition.
- **Improved Accuracy:** ML models trained on large datasets can achieve high levels of precision in diagnosing various health conditions. They can reduce errors and variability in diagnosis, providing more reliable outcomes. Techniques such as ensemble learning, where multiple models are combined, can further enhance the accuracy of health complication detection.
- **Personalized Medicine:** ML can analyze unique factors of the patient like the history of the disease, current health patterns, and lifestyle factors to tailor medical treatments and interventions. Numerous studies have demonstrated significant potential in genomic-based machine learning algorithms, where patients can benefit significantly from customized treatments or medications, improving the effectiveness of therapy, and minimizing side effects.
- **Clinical Decision Support:** Predictive algorithms can also work as decision support systems that help healthcare professionals with evidence-based recommendations, improving quality of care, and reducing the risk of errors in critical emergency conditions. Generative AI technology can provide context-based recommendation so that even a non-professional person can follow the suggestions to take first aid actions.
- **Continuous Learning and Adaptation:** ML models can continuously learn from new medical records, allowing them to adapt and improve over time. This adaptability ensures that the detection models remain up-to-date with the latest medical knowledge and practices. This continuous learning can lead to more accurate detection and prediction of health complications as the system gains experience in real-world cases. This is particularly advantageous for rescue support, as integrating more rescue records to the model training path will ultimately improve the algorithm's efficiency and result in better care.

As the field of machine learning continues to evolve, its impact on the detection of health complications is likely to grow, contributing to better patient outcomes and more efficient healthcare systems. However, there are also certain risks to using machine learning in the situation recognition or medication recommendation process. For example, neurological and psychiatric complications are difficult to assess during rescue time, especially when rescuers do not have historical patient data. Data privacy

is also a big concern, as many data providers are concerned about the leakage of personal data due to breaches of privacy and security in data collection. Applying a new method replacing years of traditional approaches can create reluctance and put strain on users, which in this case is rescue personnel. It should be considered that not all users of AI models are technically proficient people and require proper training to adapt to new methods. Training should also handle the challenges of establishing clear accountability for the outcomes of AI applications in rescue events. Therefore, it is crucial to create a robust algorithm and carefully plan the execution process before deploying it in real-world settings. If the risks associated with AI in healthcare are not adequately addressed, it can undermine public confidence, impede scientific progress, and worsen digital and health disparities.

7.2 Training Data

After confirming the features and rescue cases for each complication, the first step in the ML development pipeline was to formulate training data. In Chapters 5 and 6, horizontal and vertical data management techniques were discussed to understand the structure of training data. Separate training data was created for each specific complication, tailored for both situation recognition and treatment recommendation models. In the following subsections, details on the training data for both types of models are discussed.

7.2.1 Training Data for Situation Recognition Model

As these data will guide the machine learning algorithm on the pattern of patients and non-patients, it should contain data not only from the specific health complication, but also from patients with other complications and healthy persons. To maintain this data balance, an equivalent amount of data from other patients was incorporated for training purposes. In Table 7.1, the final size of the training data can be seen for the situation recognition models.

Table 7.1: Distribution of Data for Patients and Non-Patients for Situation Recognition Models

Complication	Total Cases	Patients	Non Patients
Cardiovascular	74,222	34,716	39,506
Respiratory	33,830	16,514	17,316
Neurological	45,870	22,780	23,090
Psychiatric	18,978	10,220	8,758
Metabolic	14,505	6,102	8,403
Abdominal	23,437	12329	11,108

Upon observing the data, it was found that feature values in many rescue cases were missing, which can hamper ML model training. In machine learning, the algorithm learns from the data it receives, and the algorithms can only learn effectively if the data is clean and complete. Models like SVM stop training and show errors in

the presence of null data fields. As a solution, data imputation was performed on the training data to fill the null cell with legitimate values. The data imputation process was conducted in two ways as follows.

- For features such as health vitals, the mean value was determined for both patients and non-patients and then null cells were filled accordingly. The main reason for mean imputation was to keep the feature pattern symmetric to a specific complication and to avoid a high bias to the overall mean. It ensured that the imputation did not significantly alter the distribution of the data while preserving the scale and variance of health vital information. Mean imputation also has some disadvantages, such as it ignores feature relationships or does not work well for a skewed distribution. But the health features in the rescue data are neither identical nor dependent on each other. In addition, the data were not highly skewed except for the outliers and the vitals always follow a range of values.
- The features which are binary or categorical, specially created using NLP, could not be filled up logically with mean imputation. Rather, it was assumed that if a feature value is not recorded for a rescue case, it means that the feature was not visible in the patient's record. So, the value was assumed to be 0. This assumption was made after discussing with the rescue personnel.

7.2.2 Training Data for Medication Recommendation Model

The training data for the medication recommendation models contained the same features as the situation recognition model, with the addition of medication information. For each of the four medication recommendation models, the three most frequent medication information was selected. As mentioned in Section 5.2.2, there were rescue cases with no available medication plan and therefore cannot be used for model training. Some of the medications were repeatedly used for multiple complications, like crystalloid. In Table 7.2, the final shape of the training data set can be seen after removing the records without adequate information on the medications. The training data is focused on predicting the correct medications based on health symptoms of patients, so segregating medication requirements for patient and non-patient was not the requirement here. Nevertheless, a diverse set of data was utilized to build a dependable and resilient model.

Table 7.2: Number of Patients Considered to be Used for Treatment Recommendation Models

Complications	Total Patients	Patients	Non-patients
Cardiovascular	18,479	13,020	5,459
Respiratory	11,742	7,224	4,518
Neurology	9,208	4,660	4,548
Psychiatry	2,359	1,158	1,201

After completing the training data, a common problem appeared as data imbalance. As providing medications is not the primary job during rescue situations, many patients were just transferred to nearby hospitals for further treatment without giving medication. In addition, the medications considered as output labels for the training data are more than one. In many incidents, only one medication was provided, so the field of other medications was empty. That is why some of the medications were used for less than 20% of the rescue cases, which means that they were significantly underrepresented compared to others, leading to a high data imbalance. When an ML model is trained with imbalanced data, it gets biased toward the majority classes, while failing to generalize well for the minority class, leading to high false negatives or missed detections. Verifying the model with test data often yields high accuracy on standard evaluation matrices, as these data typically include more instances of the majority class that the model predicts effectively. However, there are fewer instances of the minority class, which, although not predicted well, does not lower the overall performance. Such a skewed performance works poorly in real scenarios, which is a concern especially in the medical field. Generating synthetic data to increase the minority class incidences is an effective solution used by many researchers for data imbalance. In this research, the multilevel synthetic minority oversampling technique(MLSMOTE) was used, which greatly helped to achieve the expected performance of the treatment recommendation model.

MLSMOTE is an efficient data augmentation technique that is used to create synthetic instances associated with minority labels. It is particularly useful for multilabel algorithms where class imbalance needs to be fixed. In the model for recommending medications, the three labels that highlight the top three medications are not represented in a balanced manner, as not all medications are used equally for each case. MLSMOTE was instrumental in recognizing these medications with minimal use and generating synthetic data related to them. Its working principle can be divided into the following steps:

- Identify "tail labels" within the dataset. Tail labels represent classes with a significantly lower number of instances, making them candidates for oversampling. It is done by calculating the imbalance ratio per label by summing up the occurrences of the label 1 in each column. Next, it returns the columns where the imbalance ratio is higher than in other columns.
- Identify the feature vectors for the tail labels with value 1. These are called minority instances.
- Find out the nearest neighbors based on user-specific parameter of the minority instances. Generally, it focuses on finding neighbors that share the same or similar label sets to maintain label correlations in the synthetic data. Euclidean distance was used to identify the nearest neighbors for each minority instance. It is a crucial step as the number of neighbors to be selected needs to be optimized based on data type and label occurrences.

- MLSMOTE creates new synthetic instances by interpolating between a minority instance and the selected nearest neighbors. The target label of the synthetic instances was determined based on the class distribution within the nearest neighbors, ensuring that the generated samples reflect the target minority label set.
- Finally, the synthetic samples and the original data were combined to produce a balanced data set to train the multi-label algorithm.

7.3 Tuning ML Models

Tuning a machine learning model means selecting the hyperparameters of the model in a way that maximizes the model's performance without overfitting, underfitting, or creating a high variance. Hyperparameters are the configuration parameters used to control the learning pipeline and model architecture, which are not learned from the data itself, but are set prior to training. As contemporary data analysis workflows become more intricate, hyperparameter tuning is playing a highly significant role. The quality of a predictive ML model depends primarily on the hyperparameter settings used by the developer. However, the interaction between various hyperparameters and their collective impact on the quality of the model remains unclear [100]. When using a programming language such as Python to perform an ML-based classification with packages like Scikit learn, it generally uses the default parameters stored in the library. However, setting these default parameters cannot ensure the expected performance of the prediction models and additional attention should be paid to this critical step [148]. That is why choosing the right hyperparameters is essential because it not only enhances model performance but also reduces computational costs and training time by optimizing the learning process. There are several methods available for hyperparameter tuning, including grid search, random search, and more sophisticated techniques like Bayesian optimization and meta-learning approaches. In this research, grid search and random search tuning methods were used for both situation recognition and medication recommendation models as described in the following subsections.

7.3.1 Grid Search

Grid search is a brute-force exhaustive search approach that runs through a predefined subset of the hyperparameter space in all feasible settings to identify the combination that yields the best model performance based on chosen evaluation criteria. It is a widely accepted model tuning algorithm because of its simplicity and effectiveness in terms of finding an optimal solution. The grid search is a deterministic algorithm because when it receives the same grid of hyperparameters and their specified values, it always produces the same result each time executed [56]. An outstanding advantage of this algorithm is its ability to be parallelized, which significantly accelerates the search process, particularly with computational assets such as multi-core processors or distributed systems [98]. However, the grid search algorithm can also be computationally expensive because of its exhaustive approach. As the number of parameters

and the range of each parameter increase, the computation time increases exponentially, resulting in substantial computational costs. That makes it ineffective for a model that has a large number of hyperparameters. However, the features selected for the ML models in this research did not contain a large number of hyperparameters which led to the decision to use grid search for tuning. The following methodology was used to integrate the algorithm into the ML models:

- First, hyperparameters were selected that contribute highly to the model’s performance. There is a list of hyperparameters for each ML algorithm to be considered.
- A dictionary of the parameters was defined that includes all possible values of each hyperparameter. These values were defined based on historical research work and empirical evidence.
- The evaluation metric ‘Accuracy’ was chosen to determine the performance of the ML model with a subset of hyperparameter values defined in the grid.
- StratifiedKfold and Kfold cross-validation were chosen for situation recognition and medication recommendation models, respectively, to ensure that the model’s performance is robust and not dependent on a particular data split. In this folding method, each set of hyperparameters includes a similar proportion of samples from each target class as found in the overall dataset. The chosen number of folds was 3.
- The grid search algorithm was executed with the function GridSearchCV from Sci-kit learn library package of Python. All available processor cores were utilized for fast computation.

Once all parameter combinations have been evaluated, the results were aggregated to identify the hyperparameter combination that yielded the best performance metric e.g., accuracy. The cross-validation result was also obtained for each fold highlighting the mean test score and standard deviation of the scores. The best estimator was chosen to develop and validate the final model.

7.3.2 Random Search

Random search is also a popular method for hyperparameter tuning, where a random combination of hyperparameter values is selected from a predefined range or distribution. Unlike grid search, which exhaustively evaluates all possible combinations of specified hyperparameters, random search randomly samples a specified number of hyperparameter combinations from the defined parameter grid [139]. As an example, if the search space consists of three hyperparameters, each with three possible values, the random search might evaluate only 9 random combinations instead of all 27 combinations as performed by the grid search. This technique makes it highly efficient in high-dimensional spaces saving computation time.

Because of its efficiency it was additionally utilized for fine-tuning ML models. Random search also allows different types of distribution for hyperparameters, such as uniform and logarithmic uniform, enabling better exploration based on the nature of each hyperparameter [23]. Another advantage of this tuning algorithm is the parallelization of each iteration, like grid search, which speeds up the search process. Although random search requires fewer computations than grid search, at the same time there is a penalty for accuracy [97]. As the hyperparameter subset is chosen randomly, there is no guarantee of finding the optimal set. The methodology for integrating random search was exactly the same as for grid search, described in the previous section.

For the developed ML models, the parameter grid size was relatively small, making grid search a more efficient method for identifying optimal parameter sets in the majority of cases. However, both algorithms were tried for better practice. In Appendix A, the parameter grid and the best hyperparameter set found for each complication are presented.

7.3.3 Neural Network Tuning

Classical machine learning algorithms such as support vector machines, random forests, and gradient boosting, eliminate the need for users to manually construct the model structure, whereas the neural network requires it before compiling the model. Every layer starting from the input layer needs to be designed by the user with proper parameters. That is why tuning neural networks is essential to ensure that the network has the right number of layers, can quickly learn data patterns, and adjust weights and biases accordingly. For the situation recognition model, three tuners were used to tune the hyperparameters: Random search, Hyperband, and Bayesian optimization. Grid search was not used to tune neural network models as it takes indefinite times due to large hyperparameter space. The parameters chosen to be optimized were:

- Number of layers
- Number of perceptrons in each layer
- Dropout rate
- Activation functions
- Learning rate
- Number of epochs

A callback function was also integrated to monitor the loss value with value 3.

Hyperband is an advanced hyperparameter optimization algorithm that combines the concepts of random search and early stopping to efficiently identify the best hyperparameters for machine learning models, including neural networks. This algorithm uses the successful halving strategy for the optimization of hyperparameters as mentioned in [100, 73], combined with multiple rounds of resource allocation to systematically

explore the hyperparameter space. The methodology followed by this strategy starts with allocating a grid of parameters with a range of values, then evaluating the performance of all settings, removing the worst half and repeating until one set remains. This makes the hyperband algorithm quite popular, especially in neural network tuning which has a significant number of parameters defining the layout structure, and may take a long time if algorithms like grid search are used. It is always smarter to allocate more computation budget to more promising hyperparameter settings while quickly eliminating the less promising ones [99]. There are also some limitations of this algorithm, like parameter sensitivity. It has its own parameters, which need to be carefully setup, as they influence the tuning performance. Another limitation is that it treats all hyperparameter configuration points as being independent of each other. However, generally the parameter space is smooth and the performance of a model using nearby points is quite similar.

In this research, to execute the hyperband algorithm in Python, a hyperparameter optimization library called KerasTuner was used. The following arguments were provided to define the hyperband function:

- Hypermodel: It is a basic architecture of a neural network model designed with TensorFlow/Keras that contains the hyperparameter search space.
- Optimization Objective: Validation accuracy was considered as objective.
- Epochs: It refers to the number of complete passes through the entire training dataset during the training process. Values between 20 and 80 were considered.
- Factor: It is the reduction factor for the number of epochs and the number of models for each bracket. The default value 3 was considered.

Afterwards, the algorithm was executed with a callback function that had a patience value of 3 while monitoring the validation loss.

Bayesian optimization is a powerful hyperparameter optimization technique, used mainly for complex models like neural networks. It is best suited for optimization on continuous domains of less than 20 dimensions and tolerates stochastic noise in function evaluations [51]. The goal of this algorithm is to find the extrema of a scalar objective function on some hyperparameter settings, which is a subset of a defined parameter grid. The unique aspect of Bayesian optimization compared to other algorithms is its creation of a probabilistic model during function value evaluations. This model is then used to decide the next regions in the hyperparameter space for function evaluation, simultaneously managing uncertainty [153]. The steps followed to implement Bayesian optimization in this research can be summarized as follows:

- The parameter grid was defined as it was done when the hypermodel for rescue situation model was created
- A Gaussian process model was developed as surrogate for the objective function which was validation accuracy

- An acquisition function was chosen to find out the next hyperparameter set to fit in the Gaussian model for approximating objective function. There are many acquisition functions like expected improvement, probability of improvement, upper confidence bound among which expected improvement was chosen which aimed to choose points expected to improve over the current best
- The optimization process was iterated until the convergence criteria are met

Bayesian optimization is advantageous in high-dimensional spaces as it requires less iterations compared to Grid Search or Random Search, and hence it reduces the computation time. But it requires more effort, as its performance depends heavily on the right selection of surrogate model and acquisition function. The implementation of the optimization technique in Python was similar to that in hyperband. The only difference was the use of a new argument `max_trial`, which indicates the maximum number of different hyperparameter combinations to try. The value taken for it was 10.

Some other research studies [76, 3] have demonstrated the effectiveness of manually tuning ANN models, highlighting how this approach allows model developers to incorporate domain-specific insights into the design and explore custom architectures. The observations from this research suggest that manual tuning enables greater control over parameter selection, offering an opportunity to fine-tune the model for improved performance and adaptability to specific applications.

7.4 ML Model Development

For developing ML models, classical machine learning algorithms were initially emphasized for this research with a single deep learning model like the artificial neural network, as it is assumed to be more suitable for health-based decision support systems, particularly where time and resource limitations are significant factors [4]. To evaluate the effectiveness of various machine learning algorithms for recognizing situations and recommending medications, individual non-GUI-based standalone applications were created that focus solely on the back-end infrastructure that includes the ML models. The development of these models utilized Python libraries such as TensorFlow, Keras, and Scikit-learn. Both software were designed with a common architecture consisting of procedures such as selection and preparation of training data, ML model development, and comparative analysis of algorithms to identify the best performance in terms of the selected evaluation matrices. User flexibility was incorporated into the design to select different parameters. Figure 7.1 provides a procedural overview of how the ML-based software functions. In the following subsections, the complete model development pipeline is described with an architectural block diagram.

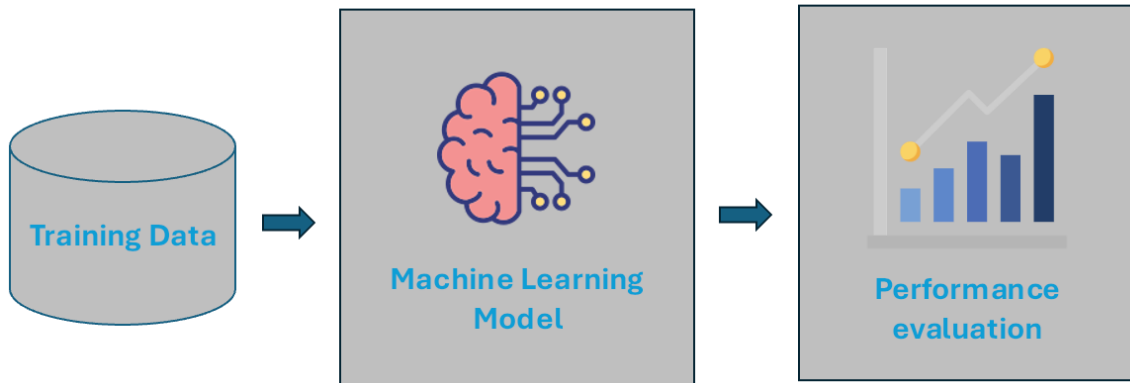


Figure 7.1: Basic Procedures Inside Machine Learning Based Software

7.4.1 Situation Recognition Models

The goal of the complication detection model was to define ML models that can provide the best performance in detecting the health situation of a patient. In this comparison, six classical machine learning algorithms and one deep learning algorithm were considered. The software was developed with Python object-oriented programming and can be used by anyone with no prior knowledge of machine learning. In addition, it is designed in such a way that the user does not need to provide any additional input except for selecting the required training data files for each complication to view the performance. For a complete understanding of the software architecture and the interconnections among its components, refer to Figure 7.2.

Training Data Storage: This block contains all the training data prepared for each complication that contains the required features. Six tables are stored which will be selected one by one in the ML builder process for model development. Users have the option to put multiple records based on different feature sets and use the inherited class 'Selectdata' and the method 'SelectFile' to consider the file of choice. The file selection method used the Python library Tkinter, which opens a window for file selection when users run the software.

Test Data: This block simply stores a table containing a test record that was never used during the training of the model.

Data Preprocessing: Before processing the data in the ML model builder, data pre-processing is done to split the main data into train and test categories. Python class 'DataPreparation' was developed with 2 methods 'DataStat' and 'split_data' to facilitate the preprocessing steps process. When calling the 'DataStat' method, it shows complete statistics of the complication data with a number of positive cases. The 'split_data' method splits the data based on the user-provided ratio and then uses recursive feature elimination with cross-validation (RFECV) or sequential feature selector (SFS) method to select the optimal number of features. Finally, the selected features were scaled using the standard scaler or the MinMax scaler according to the user definition.

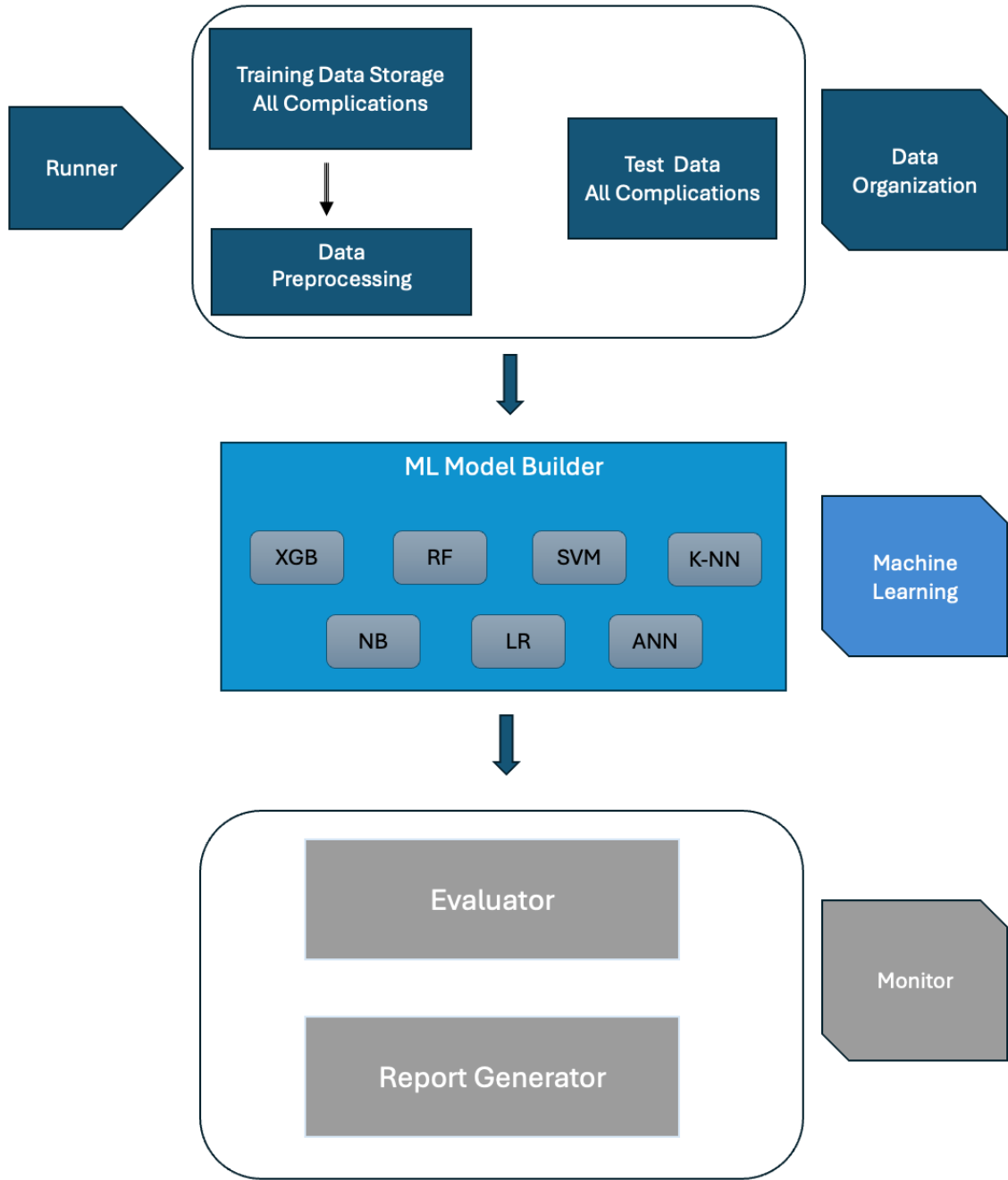


Figure 7.2: Software Architecture- Situation Recognition Model

Model Builder: This is the core block of the software, which uses the scaled data from the previous log to build the pre-selected seven ML algorithms for recognizing the health situation. The inherited class `ModelRunner` takes the splitted training data as class variable and then, with the method `BuildModel`, starts training the models for each complication. Inside the block, for each ML model, there is a separate

Python file with its own class and method 'TuneModel' for tuning the hyperparameters. The tuning method considers all the tuning options provided in the method like grid search, random search, Bayesian optimization, or hyperband and outputs the best estimator for each tuner with parameters to consider. The estimator offering the highest accuracy is automatically chosen and displayed for users' review. Afterward, the estimator is trained with the training data with the cross-validation option. Three-fold cross-validation was used to maintain the class distribution across folds. Afterward, with the Scikit learn package, the cross-validation score for each fold was calculated with metrics such as accuracy, precision, recall, and F1_score.

Evaluator: This block contains the Python class 'ModelPerformance' to use different performance evaluation techniques to identify the best model for each complication. With the methods 'ConfMetrics', and 'draw_confusion_matrix' it calculates the evaluation matrices when test data from data organization block are passed to it. Then it draws the confusion matrix for users to observe the accuracy and misclassification rate of the model. A performance matrix using the Pandas data frame was also included, which appends the performance values such as accuracy and precision of each model.

Report Generator: This block simply generates the report calculated in the evaluation block using the ClfReport method.

Runner: The top file in the software, which wraps all other blocks and runs them sequentially, is the 'Runner'. This block calls all classes and finally shows the performance of the model for each health complication as an output after users define the values of the required parameters.

7.4.2 Medication Recommendation Models

The models designed to recommend medications were more complex compared to those designed to recognize situations, as their goal is to predict three of the most commonly used medications rather than simply making one binary prediction of a health situation. There are two common ways to develop such a model: multiclass and multilabel. Multiclass models address classification challenges where the output class of available feature vectors can be varied to one out of multiple potential categories, unlike binary models, where the output is limited to one or zero. This type of model is particularly useful in various applications where the outcome can belong to multiple categories, such as image recognition, medical diagnosis, and natural language processing. Training of such algorithms can be complex due to the high computational cost, when the number of labels is very high. Data imbalance can be another challenge as some instances may occur fewer times than others. Multilabel models likewise manage diverse outputs, but specifically, through assigning multiple labels to a single vector instance. Each label may have binary instances, but can also be like a multiclass model with varied output. For example, a medical record might be associated with multiple diagnoses or an image might contain several objects. The medication recommendation model in this study dealt with three output labels relating to medications. Each label had binary values one or zero that indicated whether the medication can be recommended or not for a specific health

condition. As there were records in which the medications were mutually inclusive, the multi-label model was a better choice for designing the machine learning models. The software structure for the medication recommendation model is very similar to the situation recognition model as depicted in Figure 7.3. The software architecture showed that in the data organization part, a new block is integrated for data balancing. After creating the training data with essential features, the data needed to be balanced with MLSMOTE as discussed in Section 7.2.2. To do that, a class method MLSMOTE was created inside the data preprocessing block. The number of closest neighbors defined inside the method ranged from 5 to 10 for different complications. Users will also have the flexibility to choose the neighbor size. To set the target label '1' for generated synthetic data, at least 50% of the neighbors needed to have the same label. The models chosen for the recommender are based on suitability with the multi-label algorithm. Unlike the situation recognition model, the model builder for this is not created directly with the main ML algorithms. Rather each model was wrapped with Python Scikit-learn library 'MultiOutputClassifier'. It is a meta-estimator used for multilabel classification problems where each label is treated as a separate classification task. The model builder needed to be provided with all the suitable single-label classifiers as can be seen inside the block, and then for each target label in the multi-output dataset, the MultiOutputClassifier wrapped around a base classifier to create one classifier for each output. When a user executes the Runner block, it provides a data frame with comparative analysis of the base classifiers so that users can decide to choose one for the next steps.

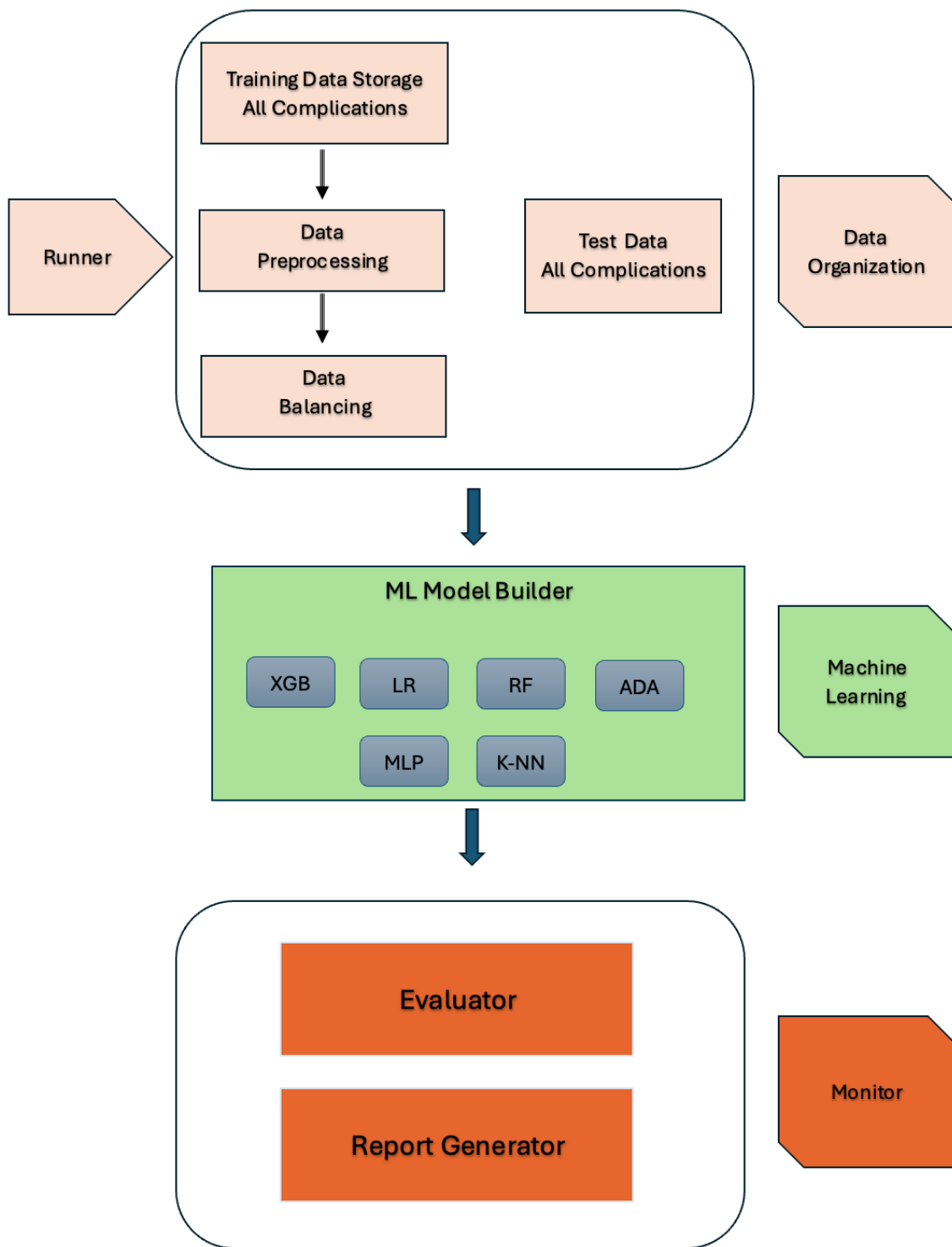


Figure 7.3: Software Architecture - Medication Recommendation Model

8

Embedded Software in Rescue Situations

Rescue emergencies are generally quite strenuous and challenging because they involve human lives in a situation where it is difficult to accurately assess the health status of a distressed patient due to personal and work limitations. Often rescue personnel face high levels of mental and physical stress, trauma, and emotional stress during rescue operations, especially when dealing with injured and vulnerable victims. This certainly impacts their decision-making abilities and overall well-being. According to recent statistics from healthcare organizations and emergency response agencies, a high percentage of patient mortality is attributed to delays in the identification and early treatment of patient complications. An additional limitation lies in resource constraints, as unlike hospitals, rescue vehicles lack modern medical equipment to assess the underlying health complications patients may have. Taking into account these challenges, real time support in patient care through software and smart wearables can play an important role in emergency events by not only providing quick suggestions but also reducing the risk of errors associated with manual decision-making. In previous chapters, algorithms required to detect patient's health situation and corresponding medication recommendation was discussed. This chapter will focus on how the algorithms can be deployed in rescue scenarios through a rescue care assistance software. Also, the possibility of accelerating situation recognition models in hardware platforms is discussed.

8.1 Rescue Care Assistance Software

The rescue care assistance software developed in this research is a suitable solution for a rescue situation as it requires only anonymous health data of patients to support

the identification of the disease in milliseconds using artificial intelligence. Artificial intelligence techniques ranging from classical machine learning to deep learning are widely used in health care software for disease diagnosis, treatment path recommendation, and patient health management [91]. The purpose of using machine learning in the health sector is to reduce the workload of physicians, minimize errors and diagnostic times, and improve performance in the prediction and detection of various diseases. However, it is equally crucial to understand that the interpretability of AI-based disease detection systems can also come with considerable risks. The algorithms might not be transparent in their decision-making processes, which can hinder healthcare professionals from comprehending and trusting the algorithmic outcomes [123]. Therefore, the software developed in this study is intended only to assist rescue teams in their decision-making process, but not to serve as a primary solution. The focus was to design reliable and robust software that can track dynamic changes in patient health data and support decision-making. The rescue care assistance software discussed here includes all the best performing ML modules developed for individual health complications and is quite flexible to be deployed on different computing devices. The upcoming subsections provide an in-depth discussion on the software requirements and the development strategy.

8.1.1 Software Requirement Analysis

Software requirements are specifications that define the functionalities, constraints, and attributes of a software system. According to IEEE, requirements are similar to features or nonfunctional constraint that software must provide to meet the needs of users and other stakeholders [70]. Generally, in the development stage, the requirements convey the user expectations of designing, developing and testing software solutions. The progression of methods for developing software systems has typically mirrored the advancement of concepts for code implementation. Some of the common software requirements that were found relevant to the rescue care assistance software are:

- **Functional Requirements:** Identify the components and methods that are necessary for the software. The requirement must describe the interactions between the system and its environment, independently of their implementation process.
- **Non-Functional Requirements:** These requirements specify the characteristics of the system, including performance, usability, reliability, flexibility, and scalability. The performance of the software can be verified with two criteria: response time and the ability to interpret the outcome of machine learning algorithms apprehensively.
- **System Requirements:** These requirements are to ensure that the software is compatible with smart devices and operating systems used by rescue personnel.
- **User Requirements:** The most important requirement is intended for specific users, which in this context is rescue personnel. The software should be

designed considering rescue situations, the rescuers' work procedure, and the environment. It should not add additional workload; instead, it should shorten the work load with quality improvement.

8.2 Software Design and Implementation

The software was developed using Python version 3.11 and Pycharm interactive development environment(IDE) in windows operating system. The Python object-oriented programming paradigm was used, which is basically a computer programming design philosophy and methodology that organizes the software architecture around data or objects, rather than functions and logic [140]. Python project template cookiecutter was used and customized to design the high-level template of the software architecture [17]. For low-level design, the classes and methods intended to integrate the ML algorithms and required test data were outlined in a virtual environment. The complete software architecture can be seen in Figure 8.1. The SW architecture is defined with five main blocks. The data storage block contains six test case data with each having 32 health characteristics of the patients. The features represent patient's health vitals, pre-illness information, injury or trauma, communication or consciousness disorder, and presence of complication-specific abnormal signs. As the test data cannot be used directly to test the algorithms, it needs to be pre-processed by converting to tensors for the PyTorch model and scaling. For this purpose, the DataModification block was designed. A method for manipulating the health vital information of the test data is also defined in this block. The method will modify the health vital values according to the user specification and evaluate the impact of these dynamic changes on the performance of the ML models.

The ML-algorithms block saves all the models i.e. ANN models and the best performing models that have been created for each complication in terms of situation recognition and medication recommendation. The ANN models were created using both the Keras and PyTorch frameworks to flexibly support future deployment at different hardware platforms. Another sub-block there was the 'PredictionModel' which is the soul of the software and runs the major operations i.e., predicting test data modified from the 'DataModification' block using the saved ML algorithms. The prediction methods in this block were designed to output the probability percentage of all six main complications based on the test data. The Test & Evaluate block contains a software test script and is intended to present the prediction results in a structured manner, making it easier to comprehend the software output and decide on subsequent steps. When test data are input into the ML algorithms, this block displays the percentage likelihood of all six complications, derived from the algorithms. The medication recommendation block contained the method 'MedRecommend' to recommend medications after the complication of a patient is identified based on the high probability percentage. Here, the probability percentage data was considered from the results of the best performing model generated from the MLPredict method inside the ML Algorithms block.

If a user wants to implement the software, the following procedures are required to

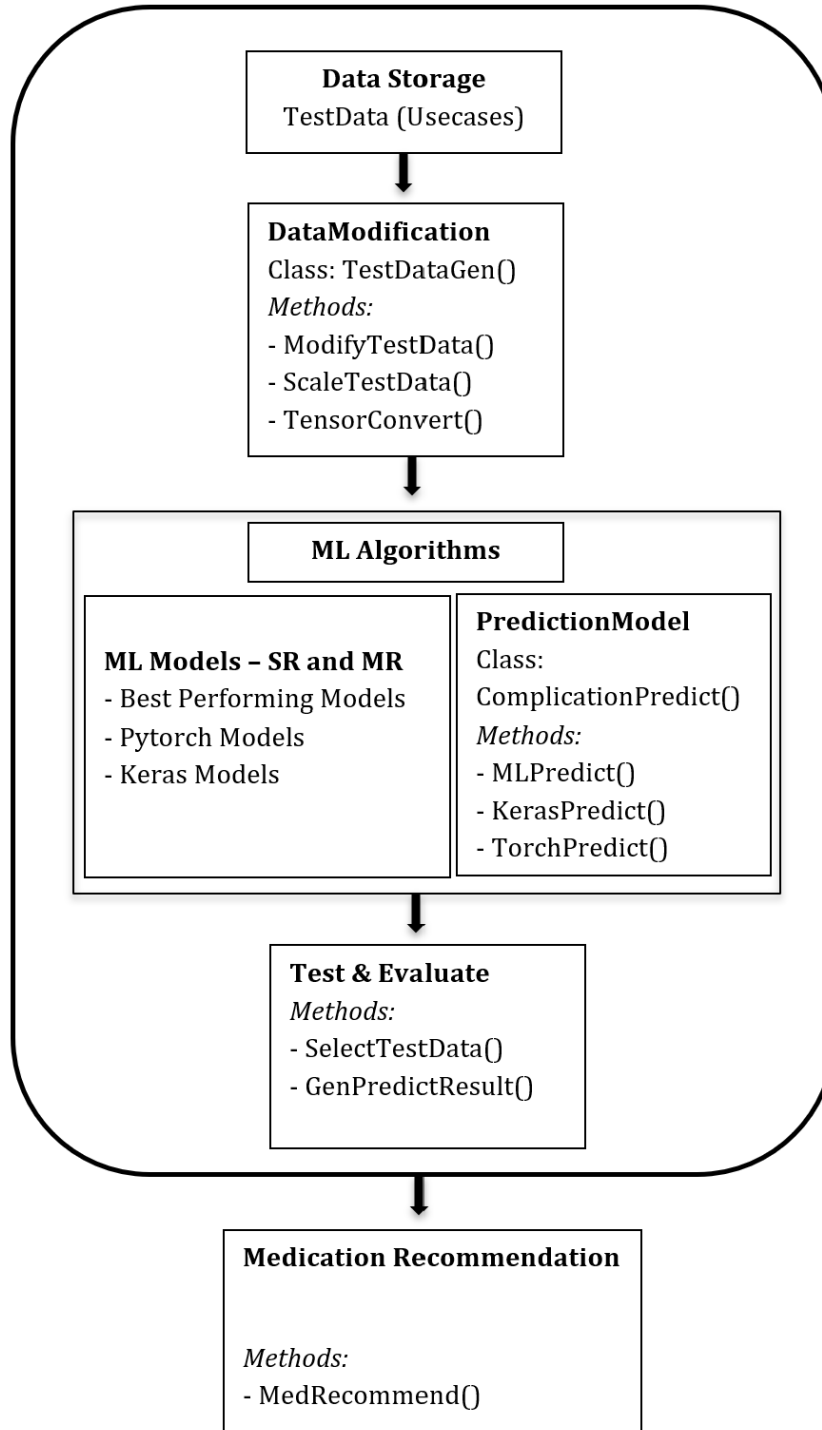


Figure 8.1: Rescue Health Management SW Architecture to Support Patient in Emergency Situations

be performed:

- Run the Demonstration.py file through the Python terminal. It will ask for the test file where the use cases were saved.
- Select the test file and a test case
- An option will be provided to the user if a change in test data is intended. The user needs to type yes or no based on his requirement.
- If yes was selected in the previous step, the program will ask how much deviation in percentage is intended
- Finally, the software will show the percentage probability of each complication for the selected test data. If modification was done, then the results based on modification will also be displayed in parallel.
- The recommended medications will also be included in the result for the complication that showed the highest probability

During the software design phase, after defining the classes and methods, each of them was carefully tested so that any flaw can be identified at the beginning. Detecting issues in the primary stage helps developers minimize the chances that these issues will be affected in later phases. Furthermore, tackling and fixing problems early is simpler, more economical, and reduces the need for redundant work [26] [178].

8.3 Hardware Acceleration

The machine learning algorithms developed in previous research can be significant and resourceful in rescue missions. To use the algorithms in real-life rescue scenarios, hardware acceleration is necessary as the rescuers need to carry a handheld energy-efficient device that can provide the diagnostics of a patient after providing the patient's data. The core of such a device can include the use of Graphics Processing Units (GPUs), Field Programmable Gate Arrays (FPGAs), microcontroller or embedded CPUs like Raspberry Pi that have the power to execute ML algorithms within a short inference time and with lower energy consumption than traditional central processing units (CPUs) [115, 58]. The choice of hardware was complicated by the fact that accelerating the modern ML models requires substantial computational resources due to their complexity and the volume of data they process.

In the following subsections, three different accelerators and their working method with ML are discussed further. The focus of hardware acceleration was on the use of portable wearables in emergency medical situations rather than the detailed analysis of hardware deployment. The ML models used for acceleration are the best performing ones in situation recognition modules.

8.3.1 Acceleration in FPGA

FPGA is a type of hardware accelerator that is often used to accelerate machine learning (ML) models due to their useful characteristics that combine flexibility, performance, and energy efficiency. Unlike traditional CPUs and GPUs, which have

a fixed architecture, FPGAs can be customized to optimize specific tasks, making them particularly well suited for the evolving demands of machine learning workloads. FPGA in its basic form is an array of programmable logic blocks such as flip-flops and look-up tables (LUTs) along with programmable interconnects that gives it the flexibility of reconfiguration as much as the user wants. There is also an input/output block that facilitates interfacing the FPGA with external networks/hardware. The following are the advantages of using FPGA for ML acceleration.

- **Reprogrammable:** FPGAs are constructed with logic gates that allow users to create custom architectures that can be optimized for specific ML workloads. This adaptability implies that, as machine learning algorithms progress, the underlying hardware can be reconfigured to meet new requirements, eliminating the necessity of acquiring new hardware [185].
- **Parallel Operation:** FPGAs can execute multiple operations simultaneously, making them particularly well-suited for the ANN algorithm used for the situation recognition and recommendation models. In addition, it reduces the time to inference, making it highly efficient in rescue situations.
- **Energy Efficient:** In emergency situations, where there is no guarantee of continuous power supply, FPGAs offer certain advantage by consuming low power by optimizing its computational operations using clock gating [69]. In addition, in remote areas where stable electricity is scarce, FPGAs can also be operated using batteries.
- **Adaptability with External Devices:** FPGAs can help alleviate input/output bottlenecks by directly interfacing with external equipment like sensors and other data sources, enabling faster data ingestion and processing. Different wireless communication modules like Bluetooth and Wi-Fi can also be integrated with FPGA to reliably receive vital health data of patients.

As a result of these benefits, recent research on accelerating machine learning (ML) on FPGAs has paved the way for adapting a range of tools and frameworks to optimize ML models while maintaining their accuracy. In this research, software framework like PYNQ, also known as Python Productivity for Zynq, was selected. The primary reason for selecting PYNQ was its high-level abstraction, as it simplifies the process of implementing machine learning acceleration and eliminates the need for in-depth knowledge of hardware design. PYNQ is an open source, Python-based framework with prebuilt libraries and overlays that can be deployed to interact with any Xilinx Zynq SoCs and MPSoCs. Its high accessibility and user flexibility reduce the entry barrier for embedded developers lacking proficiency in conventional hardware description languages such as VHDL or Verilog. By allowing users to write high-level code using just Python, PYNQ streamlines the development process for ML applications [173]. The key features of the PYNQ framework are listed below.

- Linux or Ubuntu-based fast operating system

- Provides pre-built overlays that simplify the implementation of various functions such as image processing, machine learning, and digital signal processing. This feature enables users to quickly switch between different hardware functionalities without needing to reprogram the FPGA from scratch. Overlays have the ability to encapsulate intricate hardware designs, thereby facilitating quick prototyping and testing in diverse applications [157]
- Supports integration with popular Python libraries, such as NumPy, Pandas that makes it easier for developers to implement scientific and machine learning applications
- Includes support for Jupyter Notebooks, enabling interactive development, data visualization, and real-time code execution within a browser-based environment

At present, there are many FPGAs that are pre-configured with the PYNQ framework and a Yocto-based tool flow, like the low-cost Zynq architecture PYNQ-Z1 and PYNQ-Z2. Both are low-cost development boards designed explicitly for the deployment of the FPGA-based ML models. After carefully examining the specifications of these two FPGAs, it was decided to use the state-of-the-art PYNQ-Z2 FPGA, as shown in Figure 8.2. The Z2 board offers a highly versatile and customizable platform for the deployment of neural networks, with the aim of enhancing performance, improving accuracy, reducing inference time, and optimizing power consumption. It has 1 GB DDR3 RAM, which makes it more versatile and capable than the PYNQ-Z1 with 512 MB DDR3 RAM. The biggest advantage it brings is that it comes with the Jupyter Notebook which is a custom IDE for Python programming language. With this IDE, trained ML models from the local computer can be deployed directly to the Z2 dedicated processor ARM Cortex-A9 embedded within the Zynq SoC to run inference. Accelerating directly on the Zynq processor without utilizing the FPGA fabric renders advantages to users who have expertise in Python or C++ rather than in hardware description languages. The iteration cycles are found to be faster in this way, since changes in code do not require FPGA resynthesis or reconfiguration. In addition, users can enjoy a rich software ecosystem and optimized libraries that can be easily implemented to update the ML model. However, ARM processors typically exhibit slower performance than FPGA logic gates for functions such as matrix multiplications or convolutions. This is primarily due to their absence of the fine-grained parallelism required to accelerate the operations handled by the FPGA programmable logic components. There are some other research works found on the compilation and acceleration of ML models in FPGA using acceleration tools such as TVM and VTA. But their support is limited to artificial neural networks, while there is evidence that other classical machine learning models like Random Forest and XGBoost perform better with health data. However, they are not suitable for TVM / VTA design because tree-based algorithms like RF or XGB involve irregular memory access patterns and branching logic, causing significant performance degradation on hardware [144, 150]. Given that the PYNQ Z2 FPGA comes with a pre-installed

located within the SoC. Step-by-step processes followed are described below:

- **Convert to C++:** Conifer open source framework was used to convert the chosen ML models into high-level synthesis code written in C++. The framework is compatible with tree-based models like random forests and gradient boosting and can be programmed with Python. The user needs to provide the trained ML model and fixed-point data type that represent model weights, thresholds, and activations to the framework for compilation. Afterward, the HLS-compatible code is generated in the specified output directory for the next step.
- **High-level Synthesis:** The Vitis HLS version 2021.1, a Xilinx development tool was used for high-level synthesis using source files generated by Conifer. This tool eliminates the need to write a low-level hardware description language such as Verilog or VHDL, simplifying the process of creating custom accelerators for FPGAs and SoCs. The user needs to provide the FPGA board configuration and apply directives like pipelining, unrolling, and array partitioning to the source files for synthesizing, which converts the high-level design into register transfer level (RTL) code to be exported as the intellectual property (IP) core of the ML project. IP core is a pre-designed, pre-verified, reusable unit of logic or a block of RTL code that implements a specific functionality within an FPGA design, which in this case is the ML computations.
- **Hardware Design:** The generated IP core was imported to the Vivado 2021.1 version, which was used to design the block diagram of the intended ML based embedded system. The block design contained components like: Zynq SoC, generated IP core, advanced extensible Interface (AXI) for general purpose input/output (GPIO), processor system reset and AXI interconnect as can be seen in Figure 8.3. The reset block was used to manage and distribute reset signals for IPs during power-up or when required by the application. The AXI GPIO is a programmable interface that allows FPGA-based designs to communicate with external or internal devices using simple GPIO pins through AXI interconnect. In the block design, they were used to communicate between the PS and PL for: - Providing test data written in PS to PL. - Exporting prediction output generated by the PL to the PS part. Depending on the number of input and output ports in the ML IP, GPIO blocks were placed. There were always 2 output ports representing the binary classes of the prediction value while the input port referred to the length of the data used to train the model. An additional GPIO was used to set up the ap_start signal which functioned like a control signal to start the computation of ML IP when test data were provided. Once all blocks were placed and their interconnections were routed, the design was validated to check if there was any misplacement of ports or signals. The validated design was later synthesized and used to generate the bitstream containing the configuration and functional behavior of the system developed.

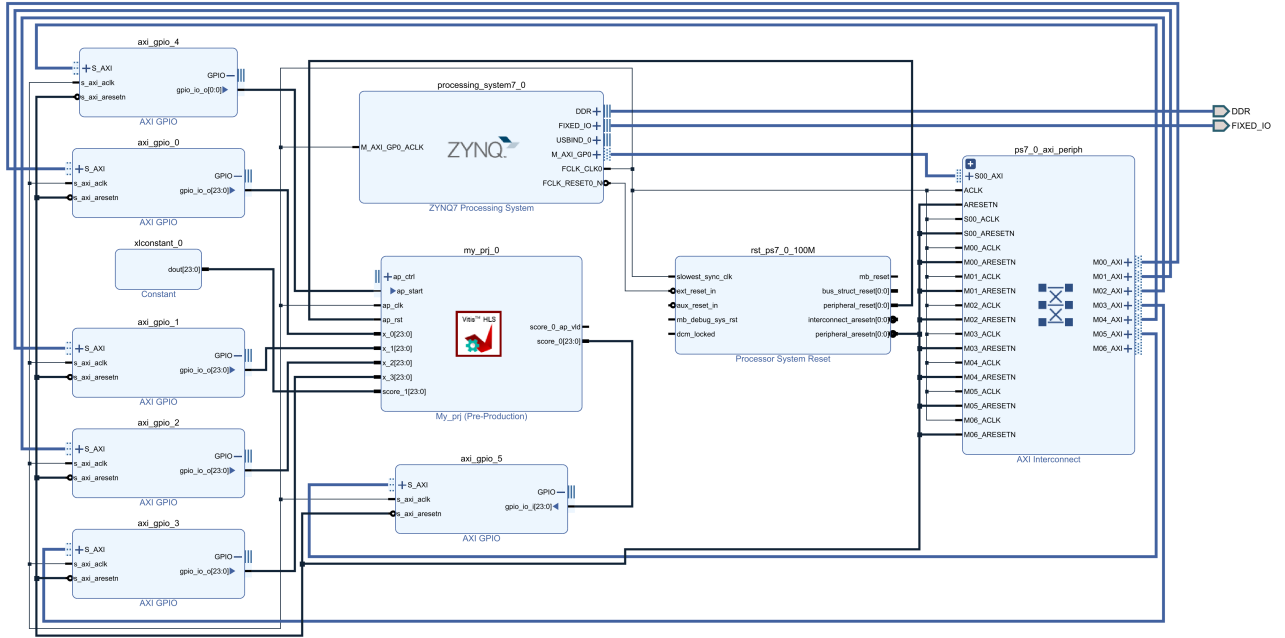


Figure 8.3: A Block Diagram Consist of Zynq 7000 Processor and Custom PL Core to Accelerate Decision Tree-based Machine Learning Model

- Creating Overlay:** From the Vivado block design, two important files- bitstream and hardware handoff were exported to the local computer that was used to create an custom overlay for the PYNQ Z2 FPGA. First those files were re-exported to the FPGA memory and instantiated through Jupyter Notebook. The overlay was checked if it contained all the IPs and ports designed before. The input and output ports were assigned with variables so that they can be easily called when data exchange is required.
- Test Inference in PYNQ:** An array of ten test vectors which was already separated from the training data was created and inserted to the input ports one by one after converting them to fixed-point numbers. Once the `ap_start` signal was initiated, the time was recorded till the prediction result of all the test vectors were available. The results were available to the output ports and could be read directly through their memory address in the notebook. As there were two output ports, only one was read which contains the prediction probability of one test case being 1. The other was connected with a constant IP block and was not required to be read. Finally, the total time was divided by ten to calculate the average inference time in milliseconds.

8.3.2 Acceleration in Microcontroller

TinyML is a field of machine learning that focuses on developing models and frameworks that can run on low power, resource-constrained devices such as microcontrollers, sensors, and edge devices [1, 29]. It enables the execution of machine learning

models directly on edge devices, allowing fast processing without the need for constant connectivity to cloud services. In addition, for real-time inference, TinyML is a reliable solution that can be deployed across a wide range of hardware accelerators [82]. The best performing machine learning and deep learning models developed in this study for individual complications were accelerated to the microcontroller using the TinyML platform to verify the suitability of the microcontroller-based embedded device in rescue situations. The steps to deploy ML models in Tiny ML can be described as following:

- The original ML models were trained on the local computer, as microcontrollers are resource-constrained devices and lack the capability to perform heavy tasks like model training. The trained models were saved in pickle format with '.joblib'/''.pkl' extensions
- A Python script was written to convert the models in C code as header file '.h'. Libraries like 'EverywhereML', 'TinyMLGen', and 'MicroMLGen' were used that reduced the size and complexity of ML algorithms to fit with the limited memory and processing capabilities of microcontrollers. The processes involved were quantization, pruning, and compression. The header files contains the model architecture such as the ANN model's weights, layers, and connection type, while for tree-based models it contains information on thresholds, feature indices, tree structures, prediction values associated with leaf nodes, etc.
- Deep learning models like ANN developed with TensorFlow and Keras cannot run directly in the microcontroller, rather a light and optimized version of ANN was used with the support of TensorFlow-Lite library. After conversion to a light version, the models were quantized to reduce their size and improve inference speed, making them suitable for rescue situation requirements. Then the quantized models were converted into C arrays, making it embeddable in microcontroller firmware
- At Arduino IDE, two individual scripts for ANN and classical ML models were written that include the library file associated. A separate script for ANN was required, as the TensorFlow model manages memory differently from traditional models, so a statically allocated memory buffer needs to be declared that can manage all tensors and their intermediate states during inference. Additional library like Adafruit-TFLITE was installed on the board which can handle TensorFlow models
- Ten test vectors were added from the original test data of the particular ML model to calculate average inference time

Currently, there are not many microcontrollers that can efficiently run ML models due to a limited resource. The microcontroller chosen in this research was BLE Sense 33 which is powered by the Nordic Semiconductor nRF52840 system-on-chip (SoC) and includes a 32-bit ARM Cortex-M4 processor running at 64 MHz [92]. For the

deployment of ML models, the Arduino provides support for accelerated and efficient floating point arithmetic operations with its built-in floating processor unit. Besides, the it's Cortex-M4 core includes DSP extensions that accelerate operations like multiply-accumulate, filtering, and Fourier transforms, benefittable for ML models inference calculation. Figure 8.4 below shows the microcontroller image with it's block diagram to understand it processing steps.

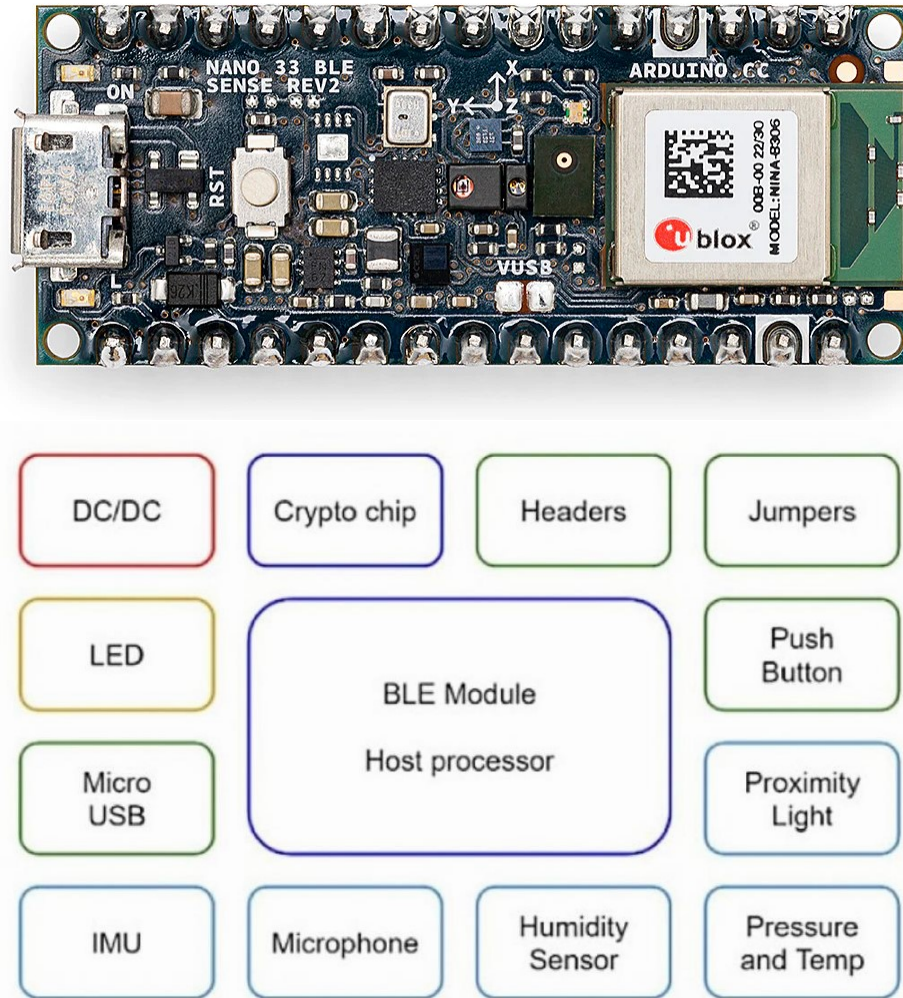


Figure 8.4: BLE Sense 33 Microcontroller with Block Diagram [64]

8.3.3 Acceleration in Raspberry Pi

Although raspberry pi has limited computational power like microcontrollers, light-weight machine learning models can still be accelerated there by using different frameworks like TinyML, TensorFlow Lite, ONNX runtime, etc. In this research, the raspberry pi chosen was the 5B model, which has advanced processing power, extensive memory, and adaptability to edge-based machine learning tasks. The model features

a powerful quad-core CPU and an improved GPU, making it capable of handling complex computations needed for machine learning inference without overwhelming the device. Compared to its predecessors, the increased RAM capacity on the 5B model allows for smoother handling of larger models and more efficient data processing, critical for maintaining real-time performance in applications with low latency. The models used to deploy on other hardware was also chosen for raspberry pi. The deployment process included the following steps:

- Like microcontrollers, the original model was trained and saved in Pickle format to load in raspberry pi later
- The trained models were transferred to the micro SD card installed to the pi. The libraries and environments necessary were created in the accelerator to load the saved file.
- The same 10 test vectors used for other accelerators were used to perform inference to calculate the average time to predict the health complication of a patient.

Figure 8.5 shows a picture of raspberry pi 5B that highlights its architectural blocks.

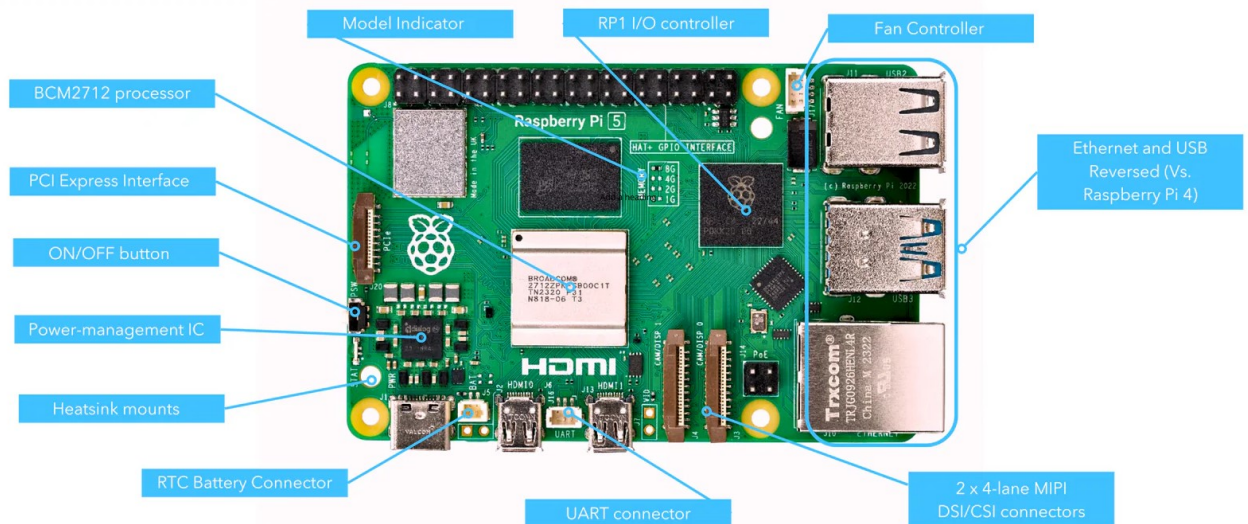


Figure 8.5: Raspberry Pi 5B [170]

The deployment of ML models on the Raspberry Pi proved to be more straightforward compared to microcontrollers or FPGAs due to the following reasons [160]:

- The processor inside Raspberry Pi is stronger compared to the other accelerators (FPGA and microcontroller) and gives more flexibility on arithmetic operations.
- Compatibility of library versions was not an issue as, like other resource-constrained accelerators, it does not limit the installation of required versions of library packages in Python.

- Raspberry pi has larger storage with SD cards.
- Hardware-level optimization is not required.
- Python scripts could be used directly.

Results and Discussion

The performance of the machine learning models for situation recognition and medication recommendation is evaluated not only based on their accuracy, but also on how effectively they can support rescue personnel when they need to make quick decisions. The performance of hardware accelerators for medical emergency situations was also evaluated to apprehend their usability in practical scenarios through portable devices. Given the high stakes in rescue care, it is crucial to use rigorous evaluation methods and clearly interpret the results to ensure reliability, safety, and real-time support. Considering these aspects, the chapter is divided into multiple sections to discuss not only the performance of individual algorithms developed for each health complication but also three use cases of real world scenarios. Examining use cases aids in understanding the impact of the model's predictions on decision making and whether they enhance patient outcomes in emergency scenarios. Developing ML algorithms for diagnosis and patient care is not enough, unless they provide actionable information and guidelines for emergency healthcare personnel in the practical field [133].

9.1 Performance Matrix

The performance of the models was assessed using several standard evaluation metrics, which are widely used in classification tasks. These metrics provide a multidimensional view of the strengths and weaknesses of the model, allowing a comprehensive evaluation that goes beyond simple precision. To measure those performance metrics, there is a requirement of calculating another matrix, called confusion matrix. It is a foundational component that shows the counts of true positives (TP), false positives (FP), true negatives (TN) and false negatives (FN) when compared with the original data. TP signifies instances that have been correctly identified as belonging to a cer-

tain category, whereas TN stands for instances that have been accurately classified as not belonging to a particular category. FP refers to instances that have been incorrectly classified as belonging to a category when they do not. FN denotes instances that have been incorrectly classified as not belonging to a category when they actually do. These components helped to evaluate the performance metrics discussed in the following.

- **Accuracy** (Acc): Accuracy measures the overall effectiveness of the model, representing the ratio of correctly predicted activities to the total number of predictions. It is formally defined as:

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

where TP is the number of true positives, TN is the number of true negatives, FP is the number of false positives, and FN is the number of false negatives.

- **Precision** (P): Precision quantifies the model's ability to correctly identify positive instances. It is defined as the ratio of true positives (TP) to the total number of predicted positives ($TP + FP$):

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

A higher precision indicates that the model has a lower false positive rate.

- **Specificity** ($Spec$): Specificity, also known as the True Negative Rate (TNR), calculates the ratio of actual negatives correctly identified by the model. It quantifies the model's ability to identify negative cases. The equation for specificity is as below:

$$Spec = \frac{TN}{TN + FP}$$

- **Recall/Sensitivity** (R): Recall evaluates the model's aptitude in identifying all relevant instances. It is the ratio of true positives (TP) to total actual positives ($TP + FN$):

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

High recall indicates that the model has a lower false negative rate.

- **F1-Score** (F_1): As a balance between precision and recall, the F1 score computes the harmonic mean of these two metrics. It provides a single metric that balances both precision and recall, which is particularly useful when there is an uneven class distribution. The F1 score is given by:

$$F_1 = 2 \times \frac{P \times R}{P + R}$$

A higher F1 score indicates that the model performs well in terms of both precision and recall.

To identify the best-performing algorithms, relying solely on evaluation matrices is insufficient. In cases where there is a class imbalance, metrics like accuracy can be misleading because they can reflect the model’s ability to correctly identify the majority class. A receiver operating characteristic (ROC) curve can be used to assess whether a data set has such critical points. It provides a more reliable measure in these situations, as it focuses on the model’s ability to differentiate between positive and negative classes, rather than being influenced by the majority class. ROC curve is a graph representation of a binary classification model at various classification thresholds, while the Area-under-Curve (AUC) tells us the degree of measure of separability between binary classes. The two parameters used to draw the curve are the false positive rate and the true positive rate of predicted classes, also known as the sensitivity. Generally, ROC-AUC curves can judge the discrimination ability of various statistical methods that combine various clues and test results for predictive purposes [65]. The value of AUC indicates the classification performance of a model. When it is 1, it means that the classifier can correctly distinguish between all the classes. A value of 0 indicates that the classifier is always making incorrect classifications.

9.2 Model Performance for Situation Recognition

The performance of the situation recognition models for each health complication will be described considering three aspects: performance metrics and their comparison for each ML model, the ROC-AUC curve, and explaining how the selected features impacted the performance. To start observing performance, users need to run the situation recognition software described in Section 8.1 and select data from a patient. After that, the software will output all relevant statistics for each ML algorithm, including cross-validation performance. In Appendix B, the detailed cross-validation result can be found for each machine learning combination.

9.2.1 Cardiovascular Complication

As can be seen in Table 9.1, the performance metrics of the cardiovascular complication detection module showed a strong overall performance for all ML models, but some models performed better in certain areas. The Random Forest model achieved the highest accuracy at 96.77%, closely followed by XGBoost (96.66%) and K-NN (96.15%). This suggests that these ensemble- and instance-based models performed well in capturing general patterns within the cardiovascular data set. Logistic regression showed the lowest accuracy at 93.34%, which indicates challenges in modeling complex relationships compared to nonlinear algorithms. Naive Bayes stood out with the highest sensitivity (95.16%), making it a good choice if correctly identifying complications (minimizing false negatives) is crucial in rescue situations. Random Forest, XGBoost, and K-NN all showed similar high sensitivity, indicating robust performance in capturing true positive cases, but slightly less so than Naive Bayes. ANN achieved the highest specificity at 99.18%, followed closely by XGBoost and

Random Forest. This indicates that ANN was highly proficient in detecting true negatives, making it dependable in situations where it is crucial to avoid false positives. ANN and XGBoost scored the highest precision (98.99% and 98.94%, respectively), making them reliable in avoiding false alarms. However, naive Bayes and logistic regression had relatively lower precision (92.62% and 91.87%), indicating that they might generate more false positives. The Random Forest model had the highest F1 score (96.45%), which balances sensitivity and precision and proved to be a balanced model. XGBoost and K-NN also performed well with F1 scores close to Random Forest, showing their effectiveness in balancing both recall and precision. Logistic Regression and SVM showed the lowest F1 scores, suggesting potential limitations in balancing sensitivity and precision in this specific application.

Classification	Accuracy	Sensitivity	Specificity	Precision	F1 Score
Random Forest	96.77	94.11	99.09	98.91	96.45
XGB	96.66	93.88	99.11	98.94	96.34
K-NN	96.15	94.05	98.01	97.67	95.82
ANN	95.37	91.05	99.18	98.99	94.85
SVM	94.24	93.21	95.14	94.37	93.78
Naive Bayes	94.15	95.16	93.26	92.62	93.87
Logistic Regression	93.34	94.18	92.60	91.87	93.01

Table 9.1: Performance Metrics of ML Models- Cardiovascular Complication. All Values are in Percentage

In summary, Random Forest, XGBoost and K-NN stood out as top performers in most metrics, with Random Forest showing the best balance in sensitivity, precision, and F1 score. ANN was highly effective if needed to avoid false positives. The random forest algorithm appeared to offer the best overall performance, balanced across key metrics, making it suitable for rescue scenarios where both accuracy and reliability in detection are needed. A comparative analysis of different performance metrics can be seen in Figure 9.1. The ROC-AUC curve drawn in Figure 9.2 indicates a similar analysis that shows the Random Forest as the top performer. The ROC curves for XGBoost, RF, and ANN are located close to the upper left corner, indicating high true positive rates coupled with low false positive rates, reflecting their high classification capacity. The XGBoost and RF curves particularly stand out with minimal deviation from the top left, suggesting that they have the highest Area Under Curve (AUC) and superior overall performance in distinguishing between classes. In contrast, models like Naive Bayes and Logistic Regression showed more deviation towards the right side, reflecting lower AUC scores and thus slightly weaker performance in terms of true positive rate versus false positive rate balance. In summary, the ROC curve indicates that both XGBoost and Random Forest are leading models, with Random Forest having a slight edge.

The number of features used by the Random Forest algorithm and their importance score can be seen in Figure 9.3. The figure suggests that mean arterial pressure was

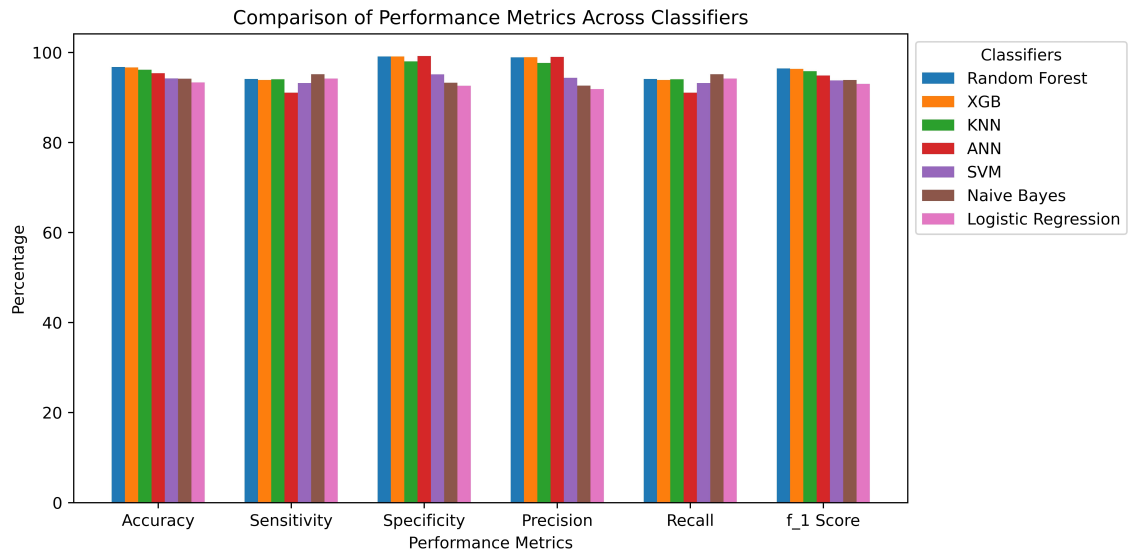


Figure 9.1: Comparative Analysis of Performance - Cardiovascular Complication Detection Algorithms

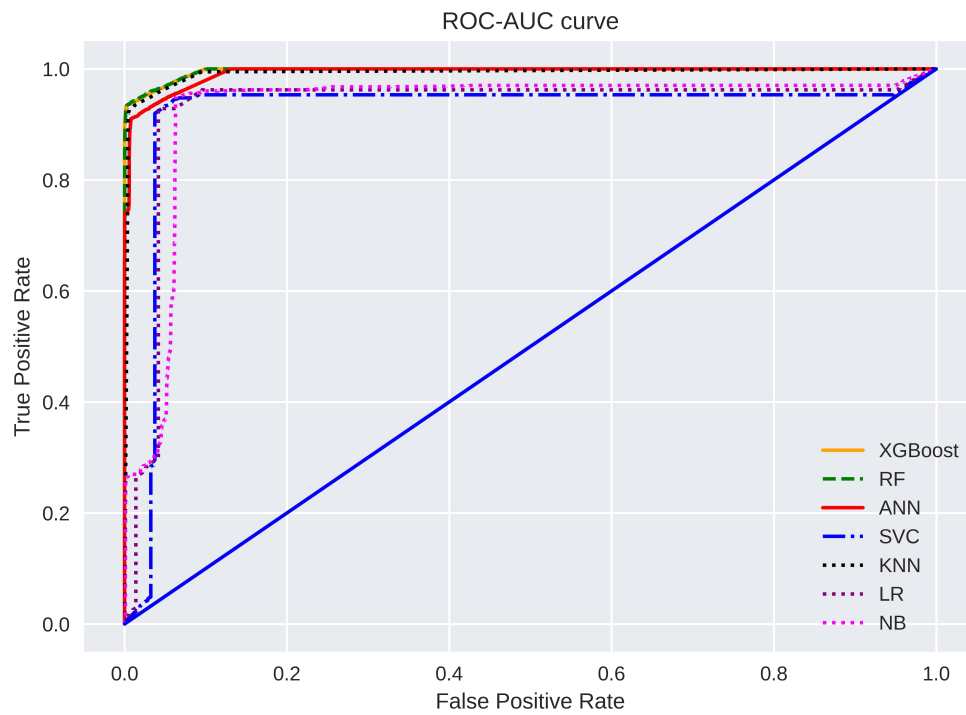


Figure 9.2: ROC-AUC Curve to Depict Performance of Cardiovascular Complication Detection Algorithms

the most influential feature, with a high importance score that exceeded 0.4, emphasizing its critical role in the prediction of cardiovascular problems due to its relevance in the evaluation of perfusion and systemic health. The respiratory rate follows with moderate importance, reflecting the interplay between the respiratory and cardiovascular systems, while the heart-related complaint contributes less, suggesting that objective physiological metrics have more predictive value than subjective inputs in this context. Systolic Blood Pressure (SBP) and Diastolic Blood Pressure (DBP) show minimal importance, likely due to their relevance to mean arterial pressure, which the model efficiently utilizes as a comprehensive metric.

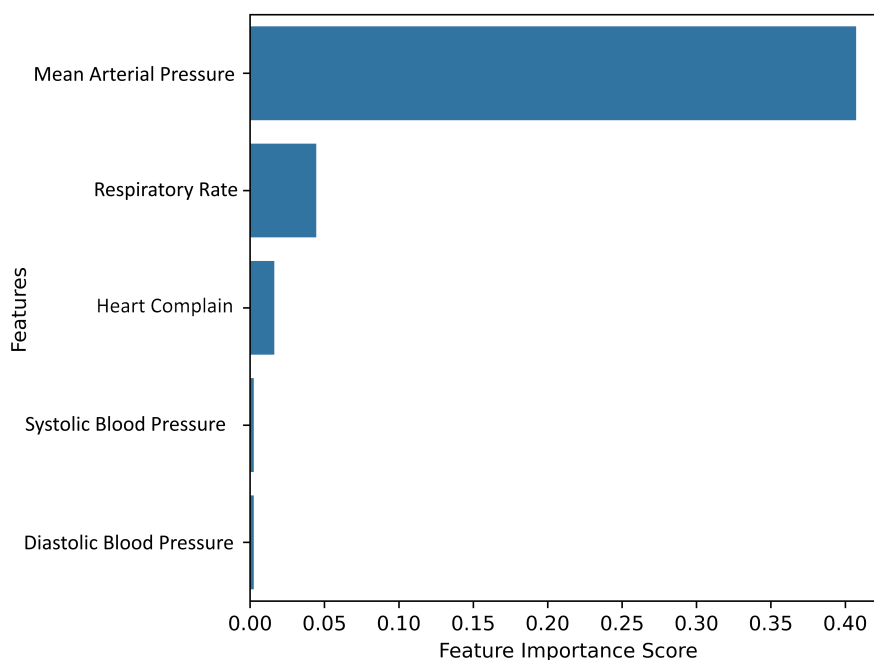


Figure 9.3: Feature Importance Score Used by Random Forest Algorithm for Cardiovascular Complication

9.2.2 Respiratory Complication

For respiratory complications, XGB has excelled in all metrics, as can be seen in Table 9.2. It achieved the highest accuracy at 92.45%, closely followed by Random Forest (92.12%). This suggests that the ensemble methods perform well in the detection of respiratory complications. Naive Bayes and Logistic Regression showed significantly lower accuracy (78.17% and 79.40%, respectively), indicating that these models may struggle with the complexity of the data set. The K-Nearest Neighbors (K-NN) algorithm demonstrated moderate performance in detecting respiratory complications, achieving an accuracy of 89.07% and a specificity of 90.61%, indicating its strength in correctly identifying true negatives. However, its sensitivity was comparatively lower at 87.45%, suggesting a tendency to miss some positive cases. With a

precision of 89.84% and an F1 score of 88.62%, K-NN balanced precision and recall reasonably well, making it a viable option for less critical scenarios where specificity is prioritized. ANN demonstrated a sensitivity of 90.73%, highlighting its efficacy in accurately identifying true positive cases. Such a high recall rate is advantageous to ensure that respiratory complications are detected reliably. XGBoost and Random Forest also had strong sensitivity values, making them reliable for minimizing false negatives, but SVM, Logistic Regression, and Naive Bayes perform considerably lower, indicating potential gaps in capturing all positive cases. Both XGBoost and Random Forest maintained the highest specificity (93.20% and 92.71%, respectively), making them effective in correctly identifying true negatives. This was essential in avoiding false positives, reducing unnecessary alerts for non-complications. Although ANN demonstrated high sensitivity, its specificity was relatively lower at 86.13%, indicating that it could generate more false positives compared to XGBoost and Random Forest. XGBoost also leads in precision (92.67%), indicating a strong balance between true positives and avoiding false alarms. This characteristic is very useful in rescue situations where false positives could carry a risk. Naive Bayes and Logistic Regression have lower precision values, indicating a higher rate of false positives compared to top-performing models. XGBoost achieved the highest F1 score (92.16%), closely followed by Random Forest, making these models better choices in the detection of respiratory complications.

Classification	Accuracy	Sensitivity	Specificity	Precision	F1 Score
XGB	92.45	91.65	93.20	92.67	92.15
Random Forest	92.12	91.51	92.71	92.36	91.93
K-NN	89.07	87.45	90.61	89.84	88.62
ANN	88.37	90.73	86.13	86.13	88.37
SVM	79.80	75.45	83.90	81.54	78.37
Logistic Regression	79.40	73.52	84.94	82.18	77.60
Naive Bayes	78.17	74.92	81.36	79.81	77.28

Table 9.2: Performance Metrics of ML Models- Respiratory Complication. All Values are in Percentage

Figure 9.4 shows the comparative analysis of the selected performance metrics.

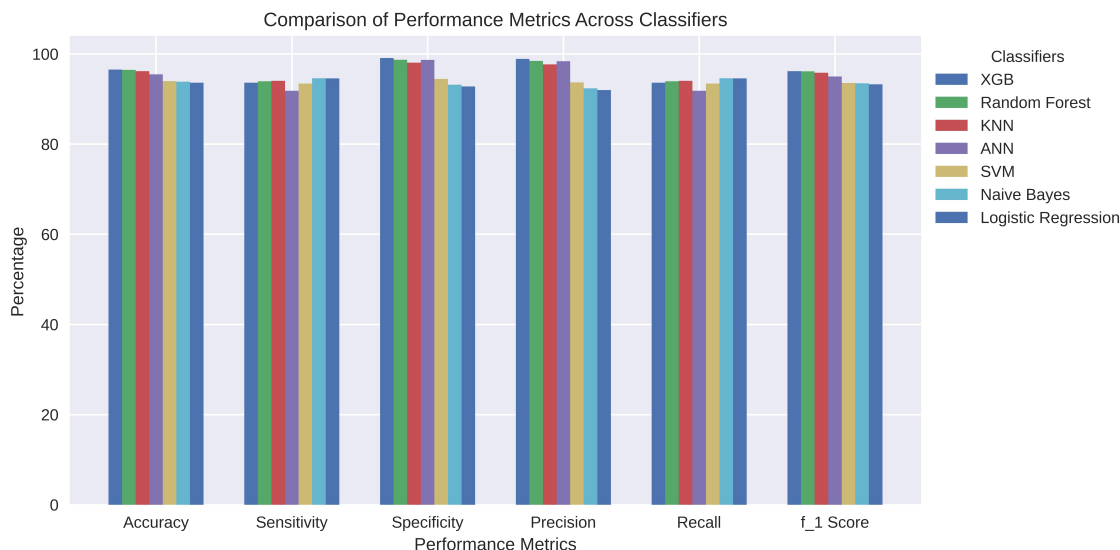


Figure 9.4: Comparative Analysis of Performance - Respiratory Complication Detection Algorithms

Looking at the ROC-AUC curve in Figure 9.5, it shows that the XGBoost and Random Forest models exhibit curves closest to the top left corner, indicating high true positive rates and low false positive rates, which generally indicate better performance. The curves for K-NN and ANN also show reasonably strong AUC values, suggesting good classification ability, although they are slightly below XGBoost and RF. In contrast, models like SVM, LR, and NB are positioned lower, with SVM showing a particularly shallow curve, which indicates a lower AUC and weaker performance in distinguishing between positive and negative cases. In general, analysis of the ROC curve highlights XGBoost as the most reliable model for the detection of respiratory complications as expected, with ANN and K-NN performing moderately well.

The feature importance score depicted in Figure 9.6 indicates that the blood glucose level was the most influential feature, with the highest importance score exceeding 0.16, indicating its critical role in predicting respiratory issues, potentially due to its relationship with systemic inflammation or stress responses. The respiratory rate was followed as the second most significant characteristic, reflecting its direct connection to respiratory health. Body temperature and pre-illness showed moderate importance, suggesting their relevance as indicators of underlying conditions or early symptoms that influence respiratory complications. In contrast, features like SpO2, respiratory abnormality, and pulse rate contributed minimally, probably because their predictive information overlapped with more dominant features or their variations were less critical in the model context. It is quite clear from the scores that the model relied heavily on physiological and systemic indicators to make accurate predictions of respiratory complications.

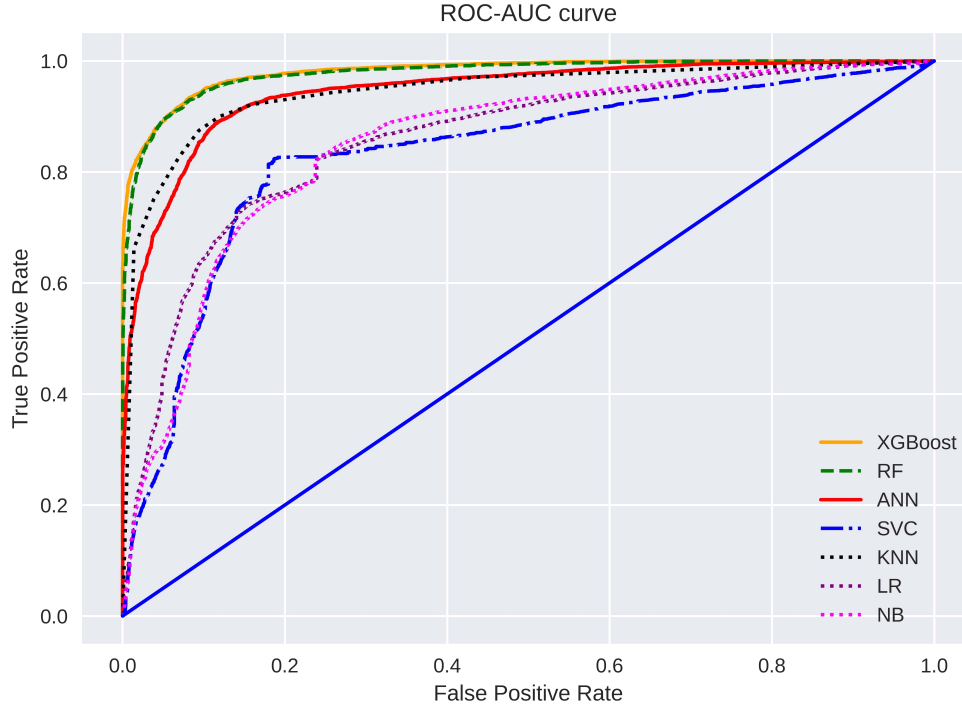


Figure 9.5: ROC-AUC Curve to Depict Performance of Respiratory Complication Detection Algorithms

9.2.3 Neurological Complication

For detecting neurological health complications in rescue scenarios, Random Forest demonstrated the best overall performance, achieving the highest accuracy of 86.09%. It exhibited balanced sensitivity and specificity scores of 88.31% and 83.92%, respectively. Furthermore, its precision of 84.32% and the F1 score of 86.27% reflected its strong ability to identify positive cases while maintaining low false positives. XGBoost closely followed with an accuracy of 85.94%, achieving a sensitivity of 88.09% and a specificity of 83.82%. Its precision of 84.25% and F1 score of 86.13% indicated a slightly reduced performance compared to RF but still made it a highly effective model. Logistic Regression achieved an accuracy of 85.74%, with strong specificity at 84.42% and sensitivity at 87.07%. Its precision of 84.70% and its F1 score of 85.87% showed that it performed similarly to ANN and RF in balanced scenarios. ANN also achieved strong performance with an accuracy of 85.73%, a sensitivity of 87.55%, and a specificity of 83.93%. Its F1 score of 85.88% and precision of 84.27% highlighted its ability to effectively balance true-positive and false-positive rates, making it a competitive option.

Support Vector Machine attained an accuracy of 84.44%, with a specificity of 85.29% and a sensitivity of 83.60%. It achieved good precision at 85.14%, but its F1 score of 84.36% reflected its slightly lower sensitivity compared to the top-performing models.

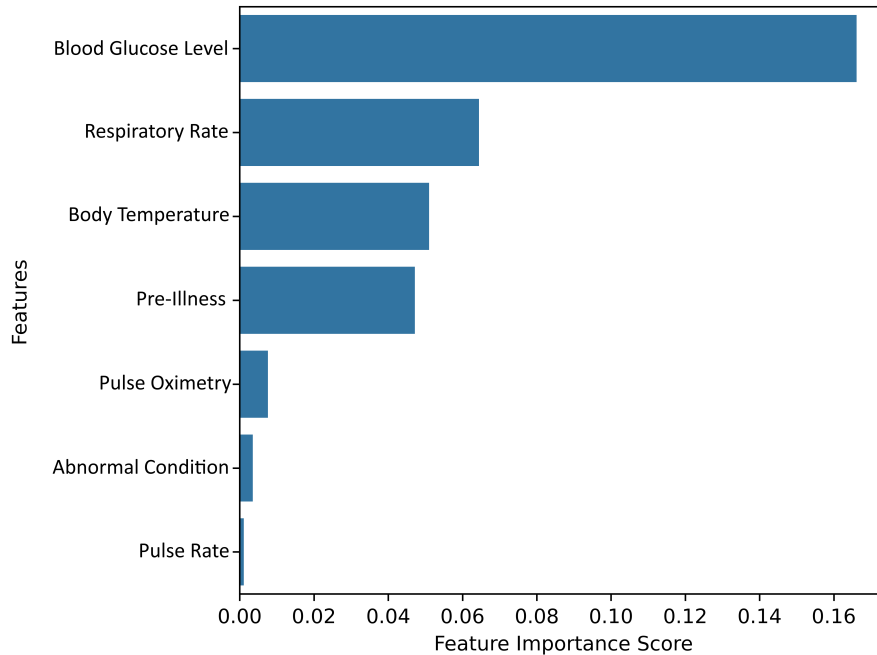


Figure 9.6: Feature Importance Score Used by XGBoost Algorithm for Respiratory Complication

K-Nearest Neighbors showed an accuracy of 84.19%, a specificity of 84.83%, while maintaining a sensitivity of 83.55%. Its precision was 84.30%, and its F1 score of 83.92% highlighted its tendency to prioritize specificity over sensitivity. Naive Bayes recorded the lowest accuracy at 81.27%. Despite its high specificity of 91.88%, its sensitivity was very low at 70.34%, leading to a higher rate of false negatives. Its precision was high at 89.37%, but the low sensitivity significantly affected its F1 score, which stood at 78.72%. This made Naive Bayes less suitable for scenarios where detecting positive cases accurately was critical.

In summary, RF emerged as the model that excelled the best in all key performance metrics. XGBoost and Logistic Regression provided strong alternatives, with ANN offering comparable performance. SVM and KNN were slightly less effective, while Naive Bayes was the least suitable model due to its low sensitivity.

Classification	Accuracy	Sensitivity	Specificity	Precision	F1 Score
Random Forest	86.09	88.31	83.92	84.32	86.27
XGB	85.94	88.09	83.82	84.25	86.13
Logistic Regression	85.74	87.07	84.42	84.70	85.87
ANN	85.73	87.55	83.93	84.27	85.88
SVM	84.44	83.60	85.29	85.14	84.36
K-NN	84.19	83.55	84.83	84.30	83.92
Naive Bayes	81.27	70.34	91.88	89.37	78.72

Table 9.3: Performance Metrics of ML Models- Neurological Complication. All Values are in Percentage

Figure 9.7 provides a graphical demonstration of the discussion above on the performance of the model.

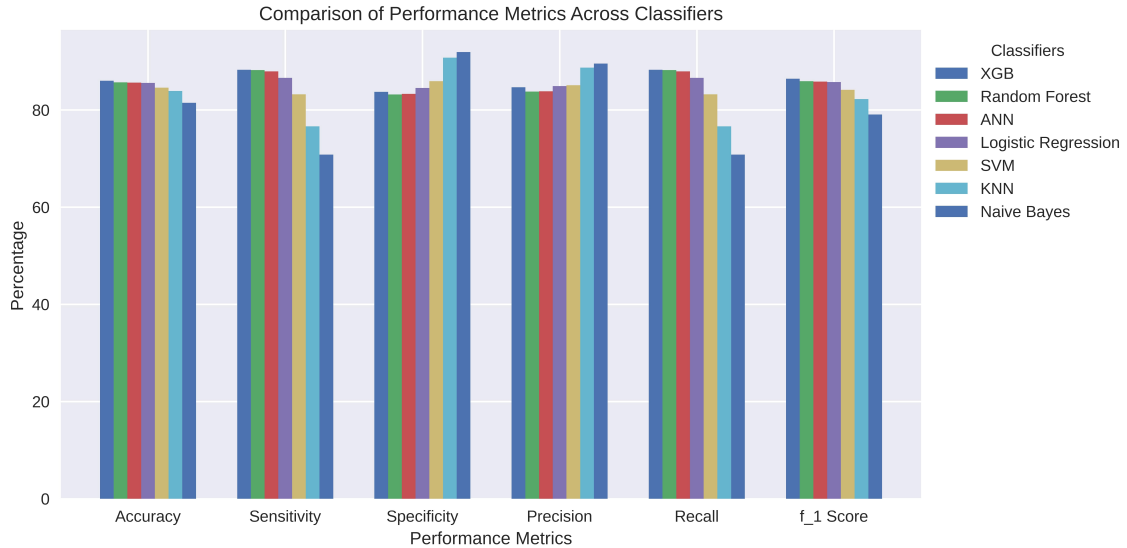


Figure 9.7: Comparative Analysis of Performance - Neurological Complication Detection Algorithms

Figure 9.8 displays the ROC-AUC curve, which shows that Random Forest, XG-Boost and ANN achieved the highest AUC scores. Their curves near the upper left corner suggest an effective equilibrium between high true positive rates and low false positive rates. The AUC curve of LR was also close to the top performing models, except at some points it's true positive rate got deteriorated. SVM exhibited moderate AUC, with curves slightly lower than those of the top performers, suggesting decent but slightly reduced sensitivity. K-Nearest Neighbors and Naive Bayes (NB) showed lower performance, with curves further from the ideal point, emphasizing their inclination for increased false positives or lower sensitivity.

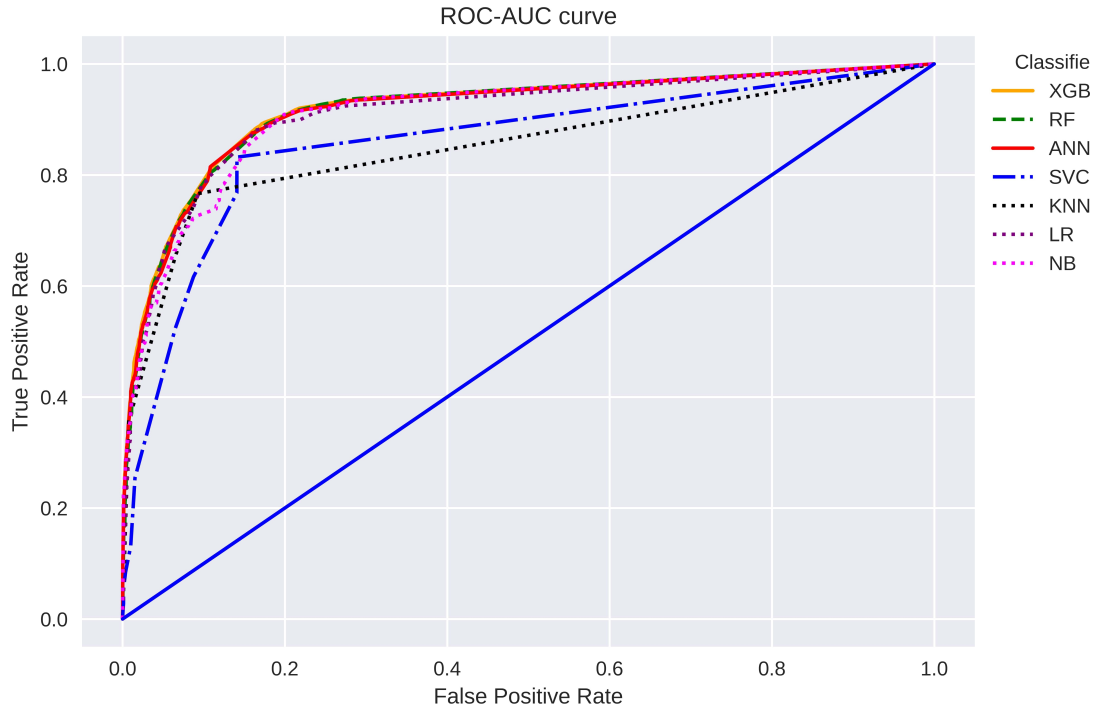


Figure 9.8: ROC-AUC Curve to Depict Performance of Neurological Complication Detection Algorithms

According to Figure 9.9, which highlights the features and their importance score for the detection of neurological complications, it was found that neurological abnormalities had the highest importance score. These abnormalities are directed towards anomalies that originate from the human brain like seizure attack and signs of brain stroke. The features that describe a patient’s communication inability or the presence of head discomfort were also notable contributors. The features indicating trauma and brain injury followed closely, signifying their influence as critical factors in neurological outcomes. However, GCS, neurological pre-illness, and consciousness disorder showed relatively lower importance, suggesting that while they provide value, they contribute less to the overall predictive performance compared to the dominant features.

9.2.4 Psychiatric Complication

Performance metrics for the psychiatric complication detection module, as can be seen in Table 9.4, reveal that Random Forest has the best performance among other algorithms. It achieved an accuracy of 89.27% and the highest F1 score of 90.01%, indicating a strong balance between precision and recall. XGBoost followed closely with slightly lower values, demonstrating strong predictive capability with balanced sensitivity and specificity. The ANN model achieved high specificity (90.58%) and

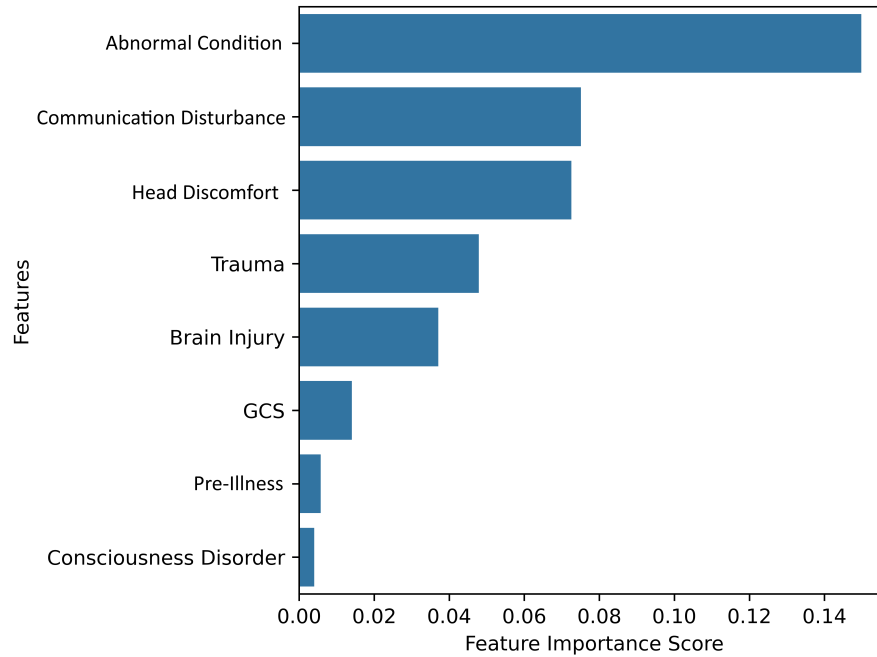


Figure 9.9: Feature Importance Score Used by Random Forest Algorithm for Neurological Complication

precision (91.54%), which highlights its strength in correctly identifying negative cases and its high confidence in positive predictions, although its recall is slightly lower than that of Random Forest.

Naive Bayes, Logistic Regression and SVM had slightly lower scores, with Naive Bayes showing decent sensitivity (83.99%) and high precision (90.11%), suggesting that it is relatively reliable in identifying positive cases but not as balanced as Random Forest. Logistic regression and SVM display similar profiles, with SVM excelling in sensitivity (86.81%) yet showing lower specificity, indicating a tendency towards true positive predictions but with some false positives. Finally, K-NN, while performing well in specificity (89.00%), has a low F1-score of 86.31%, suggesting it is less effective in balancing precision and recall compared to the other classifiers. Overall, Random Forest and XGBoost are the most effective algorithms for this task, while models like SVM and K-NN may serve as secondary options with room for improvement in balancing sensitivity and specificity.

Classification	Accuracy	Sensitivity	Specificity	Precision	F1 Score
Random Forest	89.27	89.44	89.07	90.59	90.01
XGB	87.97	87.17	88.91	90.25	88.68
ANN	87.90	85.65	90.58	91.54	88.49
Naive Bayes	86.30	83.99	89.05	90.11	86.94
Logistic Regression	86.06	83.74	88.92	90.34	86.91
SVM	85.88	86.81	84.81	86.81	86.81
K-NN	85.74	82.98	89.00	89.92	86.31

Table 9.4: Performance Metrics of ML Models- Psychiatric Complication. All Values are in Percentage

A comparative analysis between these metrics is highlighted in Figure 9.10 showing the same stat as described above.

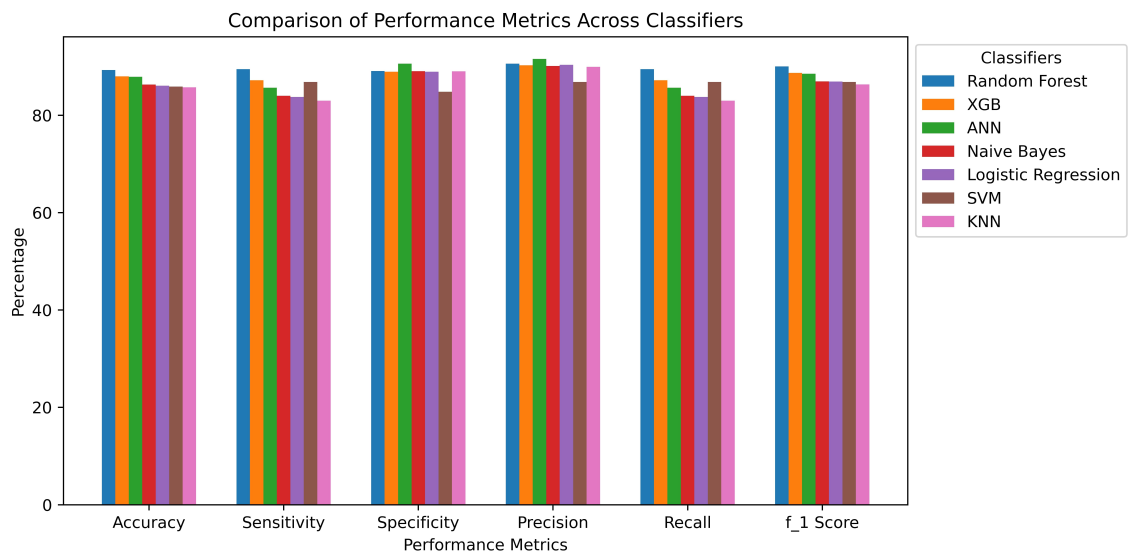


Figure 9.10: Comparative Analysis of Performance - Psychiatric Complication Detection Algorithms

The ROC-AUC curve for the psychiatric complication detection module is shown in Figure 9.11. Based on this figure, XGBoost, Random Forest and Artificial Neural Network algorithms demonstrated the highest ROC values while closely touching the upper left corner. This indicates a strong balance between the sensitivity and specificity of these models. The models showed a high area under the curve values, which meant that they were highly effective in both detecting true positives and avoiding false positives. K-Nearest Neighbors and Logistic Regression exhibited slightly lower ROC curves but still performed reasonably well, indicating decent discrimination capability. SVM and Naive Bayes have comparatively lower ROC curves, reflecting a trade-off where either sensitivity or specificity is compromised more noticeably. In

general, XGBoost, Random Forest and ANN stand out as the top performing models for the detection of psychiatric complication, as indicated by their high ROC curves and potentially high AUC values.

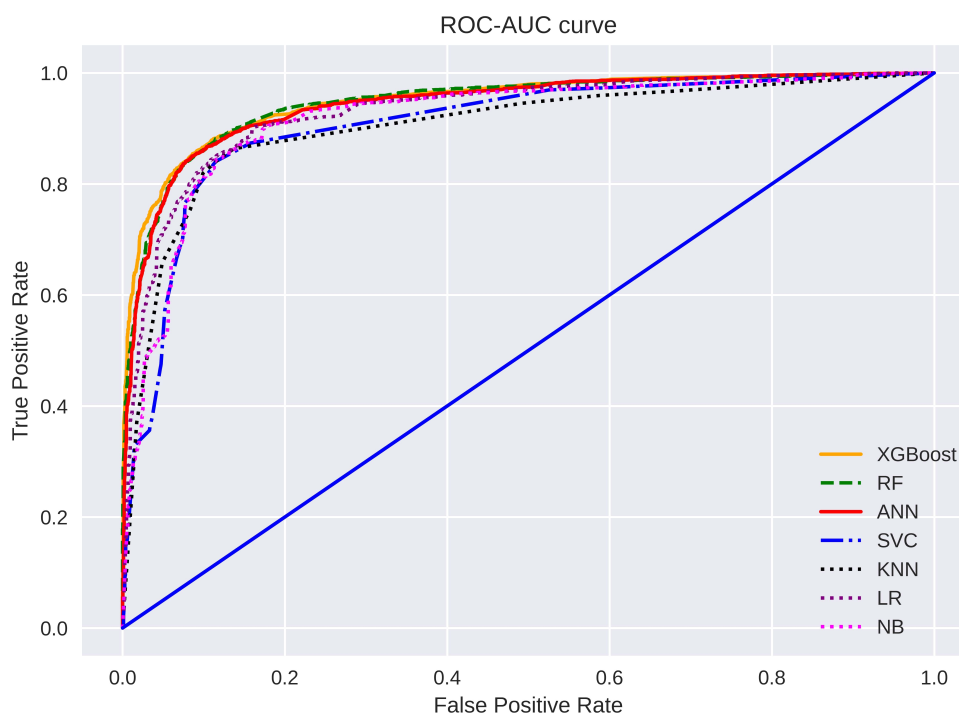


Figure 9.11: ROC-AUC Curve to Depict Performance of Psychiatric Complication Detection Algorithms

The features used by the Random Forest algorithm for the detection of psychosocial complications and their importance score can be found in Figure 9.12. It revealed that psychiatric syndrome was the most critical feature, highlighting its primary role in the detection of psychiatric complications. Alcoholism followed closely, highlighting the strong association between potential alcohol dependence and psychiatric conditions. Mental illness was also of significant importance, reflecting its relevance in identifying the underlying psychological disorders that contribute to the complications. Intoxication showed moderate importance, suggesting that substance-related factors play a significant role, although less so than primary psychological characteristics. Body circulation and systolic blood pressure were less influential, indicating that physiological metrics were secondary in predicting psychiatric issues. Finally, respiratory rate had the lowest contribution, further emphasizing the model's reliance on psychological and substance-related features rather than broader physiological measures for accurate predictions.

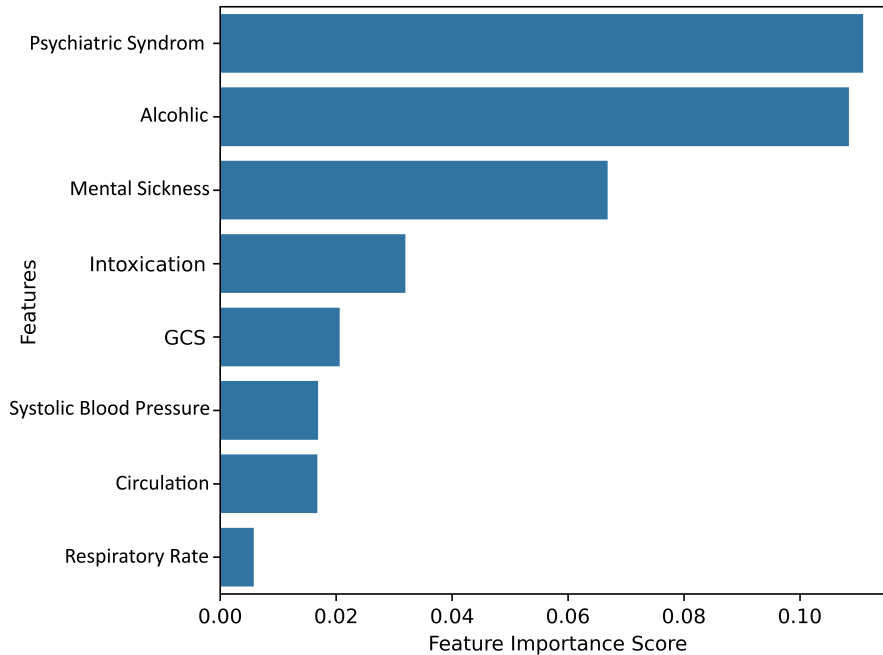


Figure 9.12: Feature Importance Score Used by Random Forest Algorithm for Psychiatric Complication

9.2.5 Metabolic Complication

In evaluating the performance of various machine learning models for detecting metabolic complications in rescue situations, it was found that Random Forest (RF) and XGBoost (XGB) are the models that perform best, offering a strong balance of sensitivity and specificity. Random Forest, with the highest accuracy (87.39%) and a good sensitivity (81.11%) combined with a high specificity (91.97%), shows the ability to reliably detect cases while minimizing false alarms. This reliability makes RF a valuable tool in a rescue context, where accurate early detection is crucial for a prompt, targeted intervention. XGBoost also performed well, with a slightly lower accuracy (86.90%) and sensitivity (78.92%) but still demonstrates high specificity (92.65%), which means that it can effectively minimize false positives, which is critical in high-stress rescue situations where unnecessary interventions could distract from other critical actions. K-Nearest Neighbors algorithm was closely followed, with decent overall performance but slightly lower sensitivity, indicating that it may miss some true cases in critical situations, which could limit its applicability when early intervention is essential.

On the other hand, the Artificial Neural Network, Naive Bayes, Logistic Regression, and SVM models showed weaker sensitivity, especially SVM, which has the lowest sensitivity (36.34%) despite high specificity (92.92%). This suggests that SVM was better at ruling out negative cases but could miss many true cases, making it less suitable in rescue scenarios where sensitivity and prompt detection of complications are prioritized over avoiding false positives. The ANN and Naive Bayes models offered

moderate performance, with ANN achieving balance but with lower specificity, and Naive Bayes underperforming in both accuracy and sensitivity. Logistic Regression's low sensitivity and accuracy also indicate limited reliability in detecting metabolic complications, thus reducing its potential value in rescue contexts.

In general, Random Forest and XGBoost emerge as the most suitable models for the detection of metabolic complications in rescue situations, with their high accuracy, balanced sensitivity, and specificity providing a reliable tool to help in rapid and accurate decision making. The performance metrics of the models are demonstrated in Table 9.5.

Classification	Accuracy	Sensitivity	Specificity	Precision	F1 Score
Random Forest	87.39	81.11	91.97	88.06	84.44
XGB	86.90	78.92	92.65	88.55	83.45
K-NN	85.06	75.79	91.69	86.69	80.87
ANN	80.38	71.68	86.72	79.76	75.50
Naive Bayes	72.73	60.45	81.89	71.38	65.46
Logistic Regression	70.17	51.30	84.62	71.86	59.86
SVM	69.60	36.34	92.92	78.27	49.63

Table 9.5: Performance Metrics of ML Models- Metabolic Complication. All Values are in Percentage

The comparative analysis of performance in Figure 9.13 showed that RF excelled in most metrics, while SVM did not show considerable performance. The reason behind this performance is the complex data pattern and non-linear relationships between the selected features in training data, which is challenging for SVM, even with the Radial Basis Function (RBF) kernel. When the feature space is too high-dimensional or intricate, it causes SVM to perform sub optimally than other tree-based models like RF and XGBoost. The ROC curve for the detection of metabolic complications in rescue situations illustrates the comparative performance of multiple machine learning models, where a steep curve near the top left indicates strong predictive capability. Here, Random Forest and XGBoost displayed the highest AUC, suggesting that they are best suited for this task because of their strong trade-off between sensitivity and specificity. These models achieved high true positive rates with minimal false positives, which is essential in rescue contexts where fast and reliable complication detection is top priority. ANN and K-Nearest Neighbors also showed acceptable performance, although their curves lied slightly below RF and XGB, indicating they may occasionally miss true cases or generate more false positives, which could delay timely intervention. In contrast, SVM, Logistic Regression, and Naive Bayes demonstrated lower AUC values, reflecting less reliable detection that might miss critical cases in high-stakes scenarios.

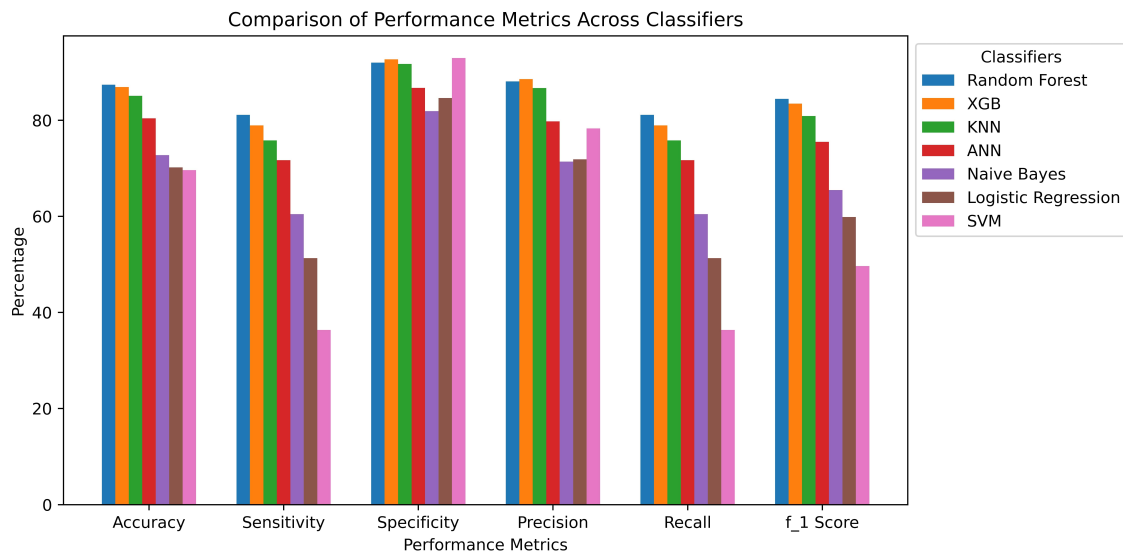


Figure 9.13: Comparative Analysis of Performance - Metabolic Complication Detection Algorithms

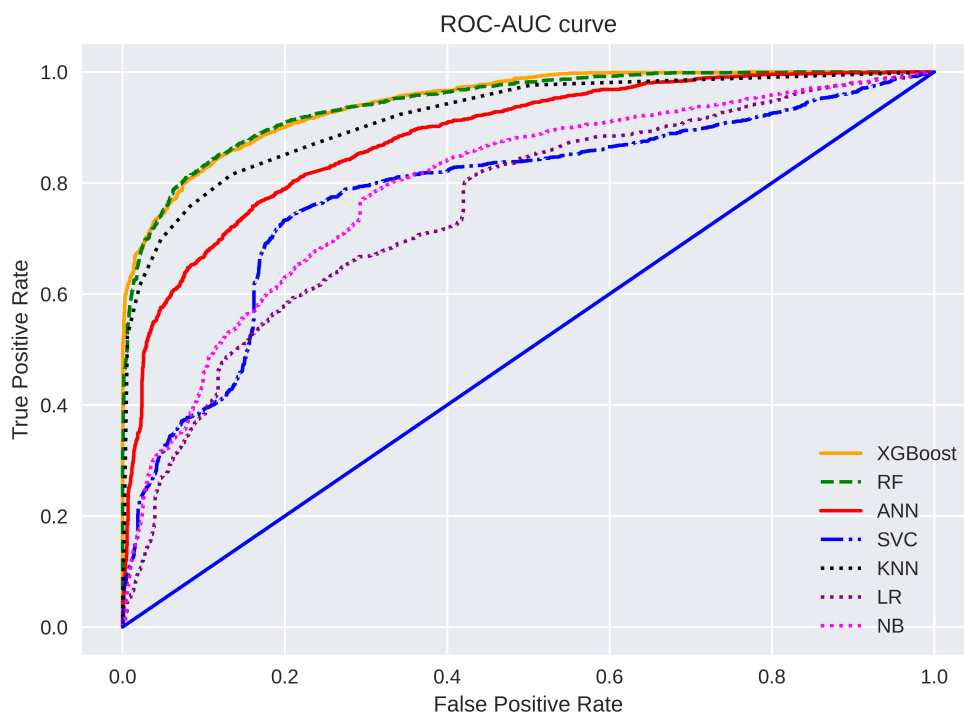


Figure 9.14: ROC-AUC Curve to Depict Performance of Metabolic Complication Detection Algorithms

To develop the Random Forest algorithm for the detection of metabolic complication, a total of seven features was used, as can be seen in Figure 9.15. Blood glucose level was found as the most significant predictor, reflecting its central role in identifying metabolic complications. The respiratory rate was the second most important feature, indicating its association with metabolic stress and systemic abnormalities. Metabolic pre-illness showed moderate importance, highlighting the relevance of prior health conditions in metabolic diagnosis. Metabolic abnormal conditions contributed moderately. However, blood pressure indicators, and pulse rate had lower importance scores, implying that cardiovascular metrics were less critical compared to direct metabolic indicators.

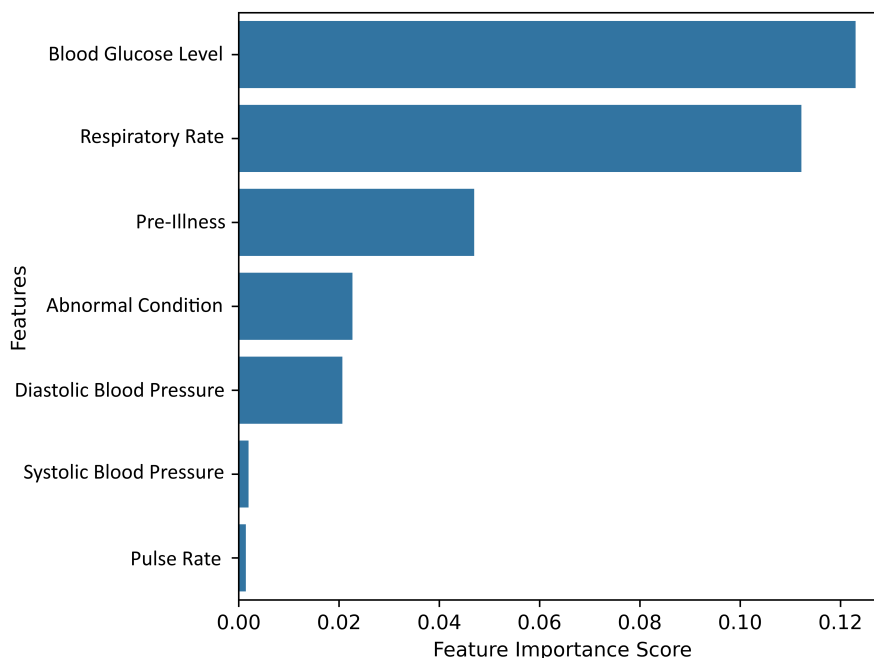


Figure 9.15: Feature Importance Score Used by Random Forest Algorithm for Metabolic Complication

9.2.6 Abdominal Complication

In analyzing the performance of different machine learning algorithms for the detection of abdominal complications in rescue situations, Random Forest and XGBoost again emerged as the top performers, as can be seen from the table 9.6. Random Forest achieved a classification accuracy of 93.49%, with a high sensitivity of 94.54%, indicating that it was highly effective in detecting abdominal complications. This model also showed strong specificity (92.30%) and balanced precision and recall, which makes it reliable in minimizing both false positives and false negatives, an essential quality in urgent rescue scenarios where precision and recall directly impact patient outcomes. XGBoost closely matched the performance of Random Forest, with

an accuracy of 93.47% and slightly better sensitivity (94.60%). This suggests that XGBoost was equally capable of identifying true positive cases while maintaining a similar level of specificity. The high F1 scores (93.88%) reflect their robustness, making them highly suitable for critical decision making required in emergency settings. The K-Nearest Neighbors showed good performance with an accuracy of 91.82%, slightly lower than the top two models. The comparatively lower sensitivity (91.41%) and precision (92.90%) of KNN meant that it may have missed some true cases compared to Random Forest and XGBoost. However, KNN could still be considered reliable, although it may not match the interpretability or speed of decision tree-based models in high-stakes scenarios. The Artificial Neural Network had moderate accuracy (87.34%) and high sensitivity (91.89%) but struggled with specificity (82.21%) compared to other models, suggesting a tendency to produce false positives. Although its high sensitivity could be beneficial in catching more cases, the increase in false positives could lead to unnecessary interventions, which could strain resources during rescues. Support Vector Machine, Naive Bayes and Logistic Regression exhibited lower performance, with accuracy of 82.10%, 81.66%, and 81.41%, respectively. SVM had a relatively high sensitivity (88.03%) but low specificity (75.43%), indicating that it was more likely to misclassify non-abdominal complications as complications. This could compromise the efficiency of the rescue response by increasing false positives. Naive Bayes and Logistic Regression had similar performance metrics but lagged behind the other models in both sensitivity and specificity, making them less suitable for time-sensitive applications.

Classification	Accuracy	Sensitivity	Specificity	Precision	F1 Score
Random Forest	93.49	94.54	92.30	93.24	93.88
XGB	93.47	94.60	92.22	93.12	93.85
KNN	91.82	91.41	92.28	92.90	92.14
ANN	87.34	91.89	82.21	85.34	88.49
SVM	82.10	88.03	75.43	80.09	83.87
Naive Bayes	81.66	83.03	80.11	82.41	82.71
Logistic Regression	81.41	80.84	82.04	82.99	81.90

Table 9.6: Performance Metrics of ML Models- Abdominal Complication. All Values are in Percentage

The comparative analysis in Figure 9.16 also indicates that Random Forest and XGBoost were the most suitable models for the detection of abdominal complication in rescue situations, providing balanced and robust performance that can improve response accuracy and efficiency.

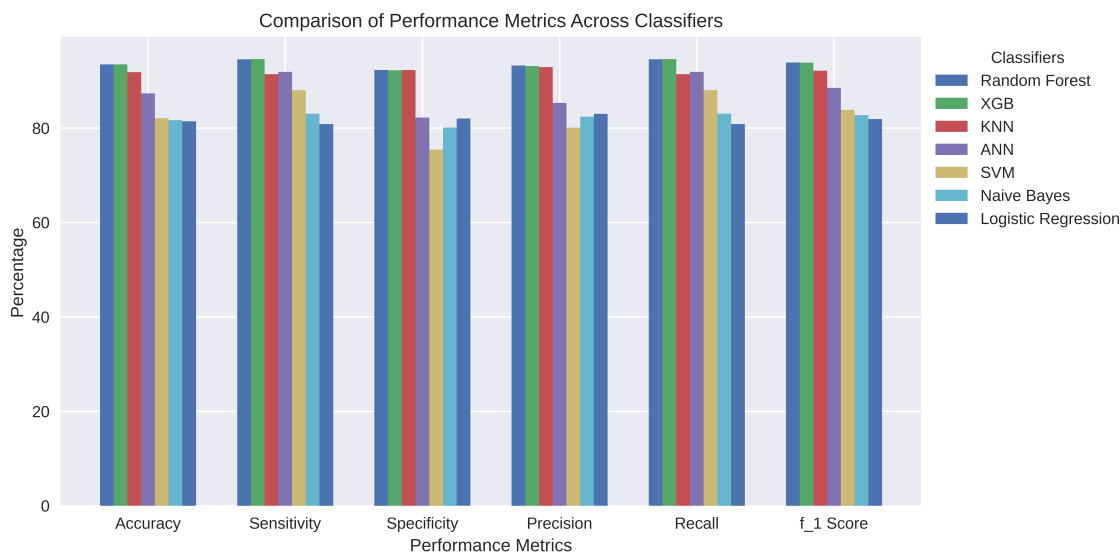


Figure 9.16: Comparative Analysis of Performance - Abdominal Complication Detection Algorithms

The ROC-AUC curves in Figure 9.17 provided insight into the relative effectiveness of the machine learning algorithms used to detect abdominal complication in rescue scenarios. RF and XGB demonstrated consistently high true positive rates in a range of false positive rates, confirming their reliability in correctly identifying complications even under challenging conditions. ANN showed a strong curve, although it was slightly below RF and XGB, indicating that, while effective, it might have been more prone to some false positives in critical rescue contexts. The performance of the KNN algorithm was also commendable, closely following the leading models, but slightly underperformed at certain thresholds, suggesting that it may have missed some cases compared to RF and XGB. Conversely, SVM, LR, and NB showed markedly lower ROC curves, particularly in regions with higher false positive rates, indicating a limited capability to differentiate between cases and non-cases reliably. In Figure 9.18, the features used to train the RF algorithm and their importance score can be seen. The blood glucose level was again found to be the most important feature, possibly due to its connection to systemic metabolic disturbances. Abdominal pain was the second most significant feature, emphasizing its direct relevance as a primary symptom in abdominal complications. Respiratory rate showed moderate importance, reflecting its potential association with abdominal distress. Abdominal abnormality and abdominal pre-illness contributed to a lesser extent, indicating that structural abnormalities and prior medical history had a more limited role in the model's predictive capabilities. The blood pressure indicators, SBP and DBP were the least significant, suggesting that cardiovascular metrics were less relevant in this context. The model relied primarily on metabolic and symptomatic indicators to identify abdominal complications effectively.

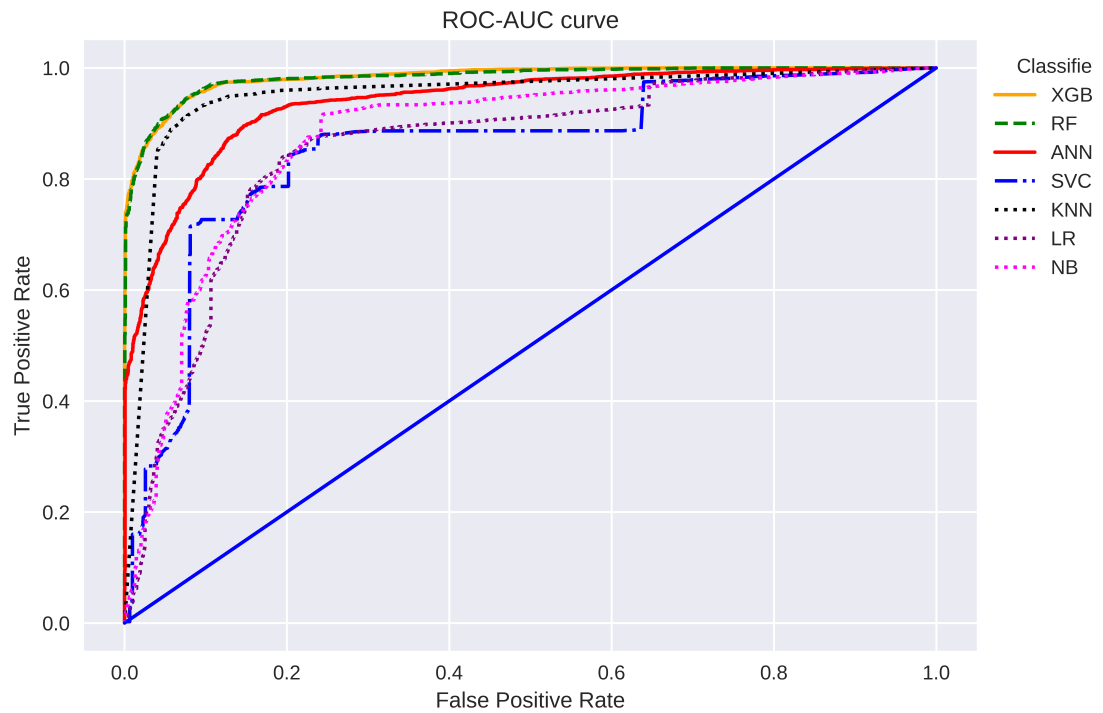


Figure 9.17: ROC-AUC Curve to Depict Performance of Abdominal Complication Detection Algorithms

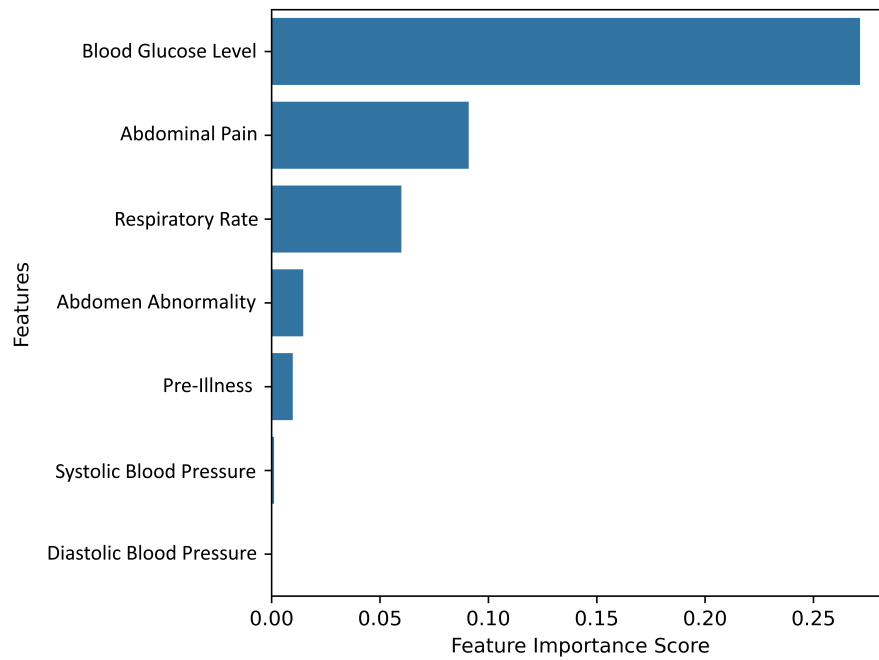


Figure 9.18: Feature Importance Score Used by Random Forest Algorithm for Abdominal Complication

9.3 Model Performance for Medication Recommendation

Evaluating the performance of the medication recommendation model is somewhat different from the situation recognition models. The recommendation models are multilabel models and their performance evaluation deals with all the pre-selected medications used as a label. Earlier discussions in Section 7.4.2 highlighted that the data set for the medication recommendation is highly imbalanced. This is due to fewer occurrences of a particular label and data dependencies among labels, meaning that the presence of one label might increase or decrease the likelihood that another label is present. In addition, unlike single-label classification, where each instance belongs to exactly one class, multilabel classification means that each sample may appear in various combinations, making it difficult to ensure a balanced representation for each unique combination. Using MLSMOTE helped to partially balance the data, but was not able to balance it for all instances when the imbalance ratio was very high. That is why models like XGBoost, RF and AdaBoost were chosen because they allow the incorporation of cost-sensitive learning, which assigns different misclassification costs to classes giving more importance to minority classes during training [138]. A detailed label-by-label performance analysis with confusion matrix was also performed for the strongest and weakest models to verify how misclassification impacted their performance. Cross validation results of the models for individual complications can be found in Appendix B.

9.3.1 Cardiovascular Treatment Recommendation

The models designed to provide treatment recommendations in term of medication for cardiovascular patients in emergency situations showed significant variations, each with different implications for practical use in high-pressure environments. The performance of each model as shown in Table 9.7 was evaluated based on precision, sensitivity, specificity, and the F1 score, reflecting the model's ability to correctly suggest treatment paths while minimizing misclassification that could delay or complicate interventions.

The Random Forest model exhibited the highest overall precision at 87.3% and an impressive sensitivity of 89.3%, indicating its ability to correctly identify patients needing urgent treatment. The balance of sensitivity and specificity of RF (85.1%) suggested that it effectively minimized false negatives and false positives, making it well suited for rescue situations where accuracy and speed are essential for effective intervention. With a high F1 score of 87.9%, RF was the most reliable model for achieving consistent performance across diverse conditions. The Extreme Gradient Boosting model, with an accuracy of 85.9% and a sensitivity of 88.5%, also showed strong potential for detecting critical cases. The slightly lower specificity of XGB (83.0%) compared to RF indicated a higher likelihood of false positives, which could lead to over-treatment in some cases. However, in rescue situations where failing to identify critical cases poses a high risk, the high sensitivity of XGB made it a vi-

able option. Its F1 score of 86.6% confirmed balanced performance, albeit slightly lower than RF. The Multi-Layer Perceptron neural network achieved an accuracy of 82.3% with a sensitivity of 85.7% and a specificity of 78.6%. Although MLP's sensitivity was competitive, its lower specificity could result in a higher false positive rate, potentially overwhelming rescue teams with unnecessary treatments. But its F1 score of 83.3% indicated decent overall performance, making MLP suitable for situations where detecting critical cases takes priority over minimizing false alarms. The K-Nearest Neighbors classifier, with an accuracy of 81.4%, demonstrated reasonable performance with a sensitivity of 83.2% and a specificity of 79.5%. Although KNN's sensitivity and specificity were having a good balance, its reliance on proximity to classify cases could be limiting in dynamic rescue environments where patient conditions vary widely. The F1 score of 82.2% indicated moderate suitability, though KNN struggled with speed and scalability in real-time situations. Adaptive Boosting also achieved a high sensitivity of 89.3% but had very low specificity (67.0%) and accuracy (78.6%), which could increase the risk of unnecessary interventions due to a higher false positive rate. While ADA's high sensitivity makes it suitable for identifying at-risk patients, its lower specificity could present challenges in resource-limited settings. The F1 score of 81.2% highlighted the strength of the model in sensitivity, but also underscored its limitations in achieving balanced predictions. Logistic Regression, despite being the most straightforward model, achieved a sensitivity of 91.9%, the highest among all models, but had the lowest specificity (62.4%) and an accuracy of 77.8%. The model's tendency to overestimate positive cases was evident, and while this may ensure critical cases are seldom missed, it could strain resources with false positives. An F1 score of 81.1% showed that while LR was strong in identifying critical cases, its trade-off in specificity was substantial. In conclusion, the Random Forest model emerged as the most balanced and reliable choice among others, due to its high accuracy, sensitivity, and specificity. The Extreme Gradient Boosting model also demonstrated potential with high sensitivity, making it suitable for urgent identification of critical cases. Models like Adaptive Boosting and Logistic Regression offered high sensitivity but suffered in specificity, making them critical to be used in rescue situations. Each model presented a unique trade-off, underscoring the importance of aligning model selection with the specific needs and constraints of rescue operations. A comparative analysis of the evaluation metrics for all the models is shown in Figure 9.19.

Classification	Accuracy	Sensitivity	Specificity	Precision	F1 Score
RF	87.3	89.3	85.1	86.6	87.9
XGB	85.9	88.5	83.0	84.9	86.6
MLP	82.3	85.7	78.6	81.2	83.3
KNN	81.4	83.2	79.5	81.4	82.2
ADA	78.6	89.3	67.0	74.5	81.2
LR	77.8	91.9	62.4	72.6	81.1

Table 9.7: Performance of Medicine Recommendation Model - Cardiovascular Complications. All Values are in Percentage



Figure 9.19: Comparative Analysis of Performance Metrics for Cardiovascular Complication

The class imbalance affected the performance of the models like LR, especially for the specificity that depends on false positives. In Figure 9.20 and Figure 9.21, the confusion metrics for individual labels are drawn corresponding to the RF and LR algorithms. The 'Antihypertensive' medication category exhibited poor performance in both algorithms due to a severe data imbalance that remained unresolved, even when MLSMOTE was applied. For this category of medications, RF showed a very high TN count and a very low FP count, indicating excellent specificity. However, the low TP count and high FN suggest poor sensitivity, which means it is difficult to correctly identify positive cases. The LR algorithm, on the other hand, had perfect specificity (no false positives), but completely failed to identify any true positives, as shown by the TP count of 0. The false negative count (729) was very high, indicating that the model does not detect positive cases at all.

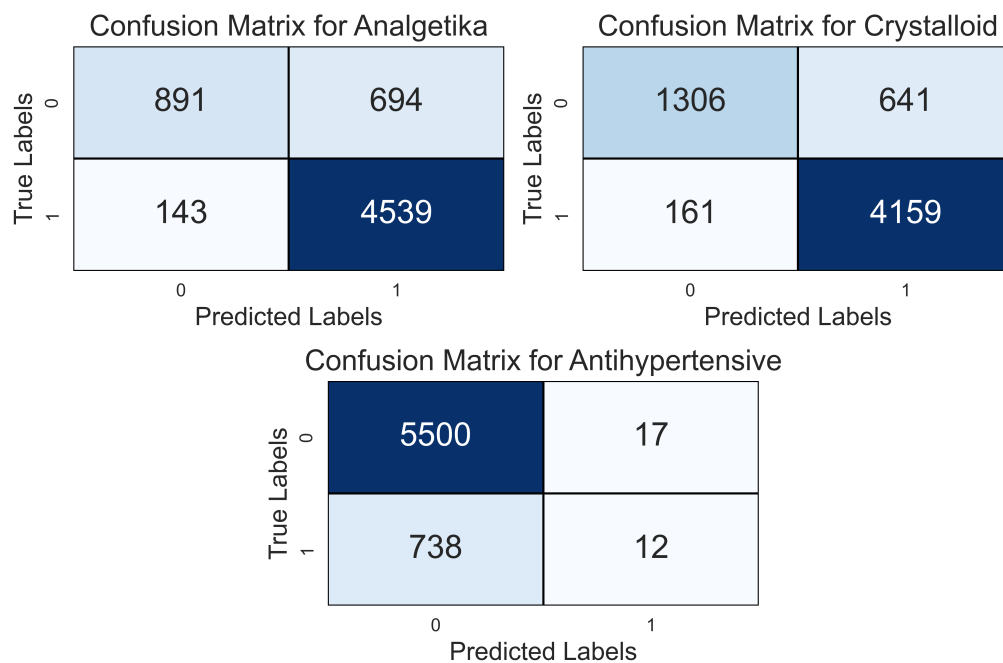


Figure 9.20: Confusion Matrix for Individual Labels with RF Algorithm

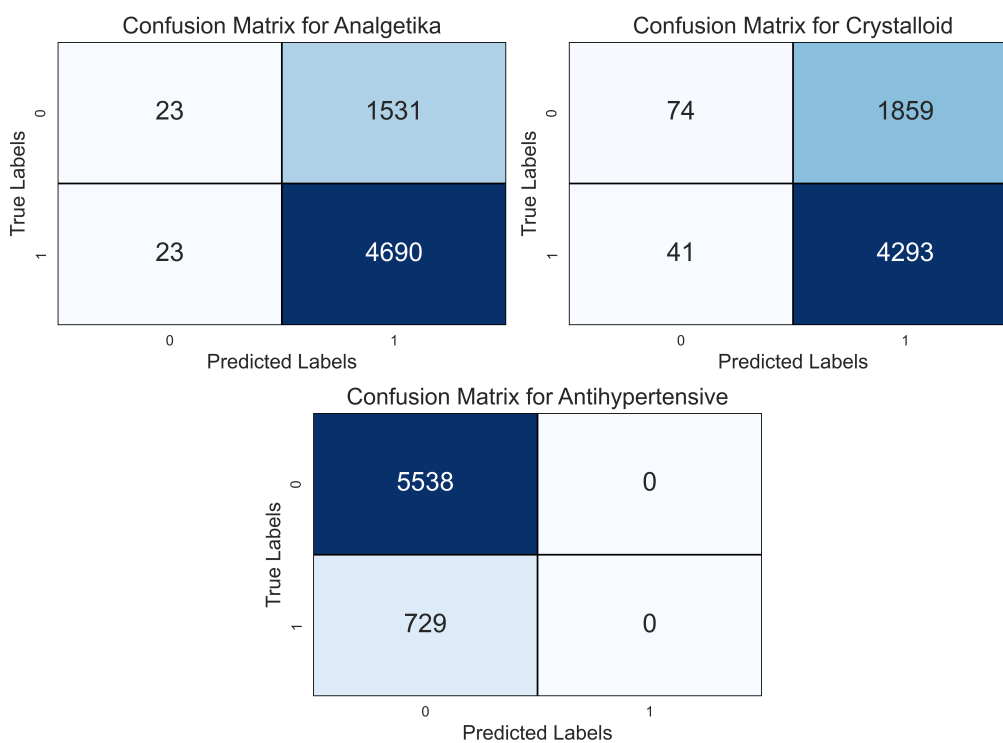


Figure 9.21: Confusion Matrix for Individual Labels with LR Algorithm

9.3.2 Respiratory Treatment Recommendation

The respiratory complication medication recommendation module showed a varied performance for different algorithms as depicted in Table 9.8. XGBoost achieved the highest accuracy of 82.8%, sensitivity of 78.2%, and specificity of 85.6%. These metrics suggest that XGBoost performed robustly in correctly identifying patients requiring treatment (true positives) while minimizing false positives. The precision of 77.6% and the F1 score of 77.8% indicate balanced performance in rescue scenarios, making XGBoost a highly reliable choice for high stake situations. Its superior specificity is particularly important in reducing over-treatment risks. Random Forest also performed well, with an accuracy of 82.0% and the highest specificity of 87.7%, outperforming XGBoost in this aspect. However, its sensitivity (73.4%) was lower, indicating that it could miss more true positive cases compared to XGBoost. With a precision of 79.8% and an F1 score of 76.4%, RF demonstrated balanced predictive capabilities, although slightly behind XGBoost in overall effectiveness for rescue applications.

Multi-layer perceptrons had slightly weaker performance, with an accuracy of 75.9% and a sensitivity of 68.1%. Although its specificity (81.0%) was considerable, the lower precision (70.3%) and the F1 score (69.1%) highlight some inconsistencies in its predictions. MLP may be suitable for general scenarios, but may require optimization or integration with other models for critical rescue operations where high sensitivity is crucial. Another model with K-Nearest Neighbors achieved an accuracy of 75.0%, with lower sensitivity (63.4%) but relatively high specificity (82.7%). Its precision of 70.5% and an F1 score of 66.7% suggest a trade-off between identifying true positives and minimizing false positives. Although this algorithm may be a viable option for scenarios with low urgency, its lower sensitivity makes it less reliable. AdaBoost displayed an accuracy of 71.5% with a sensitivity of 67.6% and a specificity of 74.1%. Its precision of 64.0% and an F1 score of 65.7% were among the lowest, reflecting the limited ability to handle complex rescue scenarios. Despite its potential for simplicity and interpretability, the overall performance of the ADA was suboptimal for the critical requirements of recommending treatments. Logistic Regression exhibited the lowest accuracy (70.8%) and specificity (71.1%), along with a sensitivity of 70.5%. Its precision of 61.4% and an F1 score of 65.6% further highlighted its limitations for the recommendation of medicines.

In summary, XGBoost was the best choice among others due to its performance, while Random Forest, with its high specificity, could serve as a complementary model to reduce over treatment. MLP and KNN offer moderate performance, but require enhancement to match the criticality of rescue operations. ADA and LR demonstrated the least suitability due to their lower overall performance metrics. An comparative performance analysis including all metrics is highlighted in Figure 9.22.

Classification	Accuracy	Sensitivity	Specificity	Precision	F1 Score
XGB	82.8	78.2	85.6	77.6	77.8
RF	82.0	73.4	87.7	79.8	76.4
MLP	75.9	68.1	81.0	70.3	69.1
KNN	75.0	63.4	82.7	70.5	66.7
ADA	71.5	67.6	74.1	64.0	65.7
LR	70.8	70.5	71.1	61.4	65.6

Table 9.8: Performance of Medicine Recommendation Model - Respiratory Complications. All Values are in Percentage

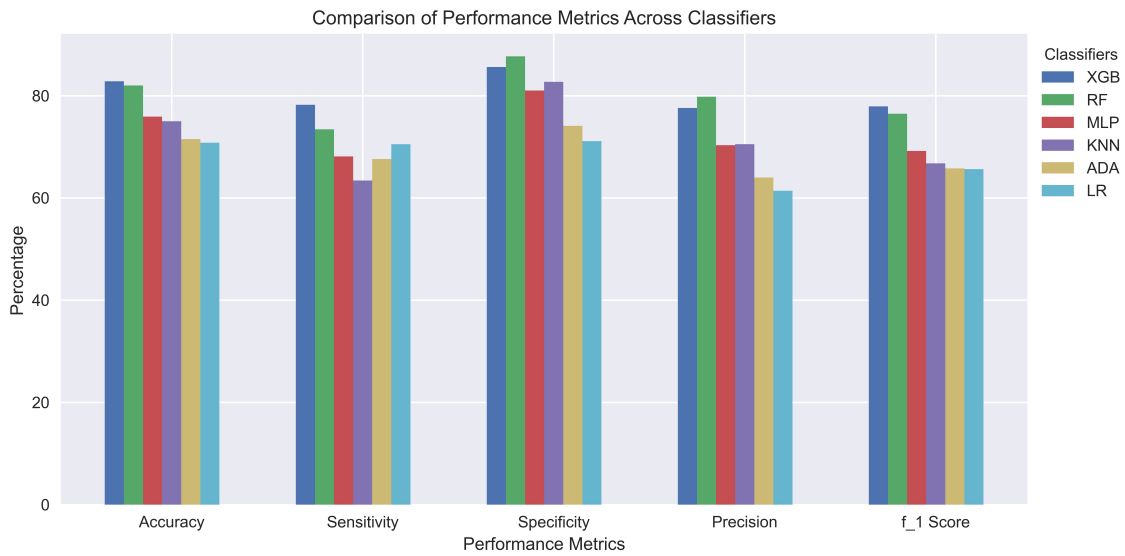


Figure 9.22: Comparative Analysis of Performance Metrics for Respiratory Complication

The performance of LR algorithm was also poor for respiratory complications medication recommendation like cardiovascular. In Figure 9.23 and Figure 9.24, the confusion metrics for the individual labels are drawn for the strongest and weakest models XGB and LR, respectively. The LR confusion matrix showed that although the model performed well in detecting true positives for crystalloid medication, its false positive rate (1746) was very high, resulting in frequent misclassification of patients who needed crystalloid treatment when they did not. For anaesthetics medication, the model did not detect any true positives ($TP = 0$), which means that it did not correctly classify a single patient as requiring this treatment. Instead, all positive cases were misclassified as negative ($FN = 659$). This reflects an alarming sensitivity issue, as no anaesthetic treatments would be recommended, even when genuinely needed. For the bronchodilation drug, performance was moderate, but still struggled with sensitivity as it correctly identified 1096 positive cases (TP) but missed 779 cases

(FN). All together, this led to poor performance for the LR algorithm. XGB, on the contrary, exhibited better performance in all drug recommendations. For crystalloid treatment, it reduced false positives (661 vs 1746) and balanced sensitivity and precision, unlike LR, which suffered from over-predictions. For analgetika treatment, LR did not completely detect true positives, while XGB identified 342 true positives with minimal false positives (72). For bronchodilation, XGB significantly lowered false negatives (422 vs 779) and false positives (371 vs 455), resulting in better sensitivity and precision. The XGB algorithm excels due to its enhanced ensemble methodology, superior management of feature interactions, and strong regularization, which makes it more dependable for precise treatment suggestions compared to LR's constraints with linear decision boundaries and its susceptibility to class imbalances.

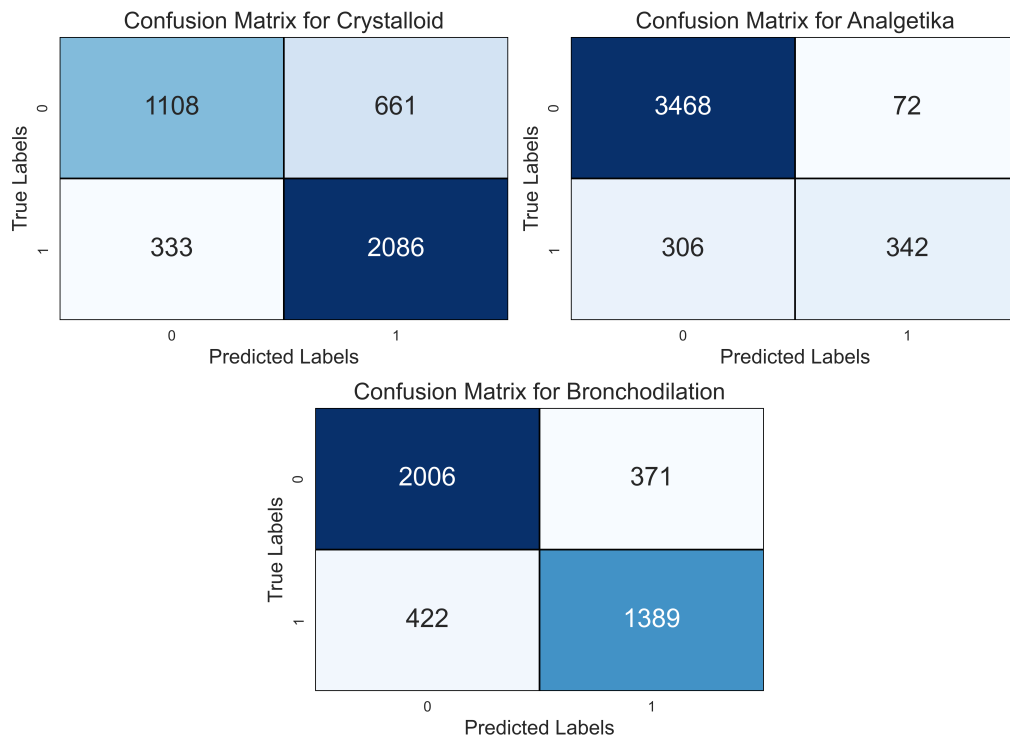


Figure 9.23: Confusion Matrix for Individual Labels with XGB Algorithm

9.3.3 Neurological Treatment Recommendation

Looking at the performance of neurological models in Table 9.9, XGB again led the performance chart in all metrics except specificity. It achieved the highest accuracy (83.7%) and balanced sensitivity (72.7%) and specificity (89.7%), outperforming Random Forest, which followed closely with the accuracy of 83.1%. Although RF was slightly lower in all metrics, its performance was comparable, making it another strong candidate for medication recommendation for neurological cases. The multilayer perceptron model exhibited the highest specificity of 90.1% but had a lower sensitivity (66.9 %), indicating its tendency to miss positive cases while being precise

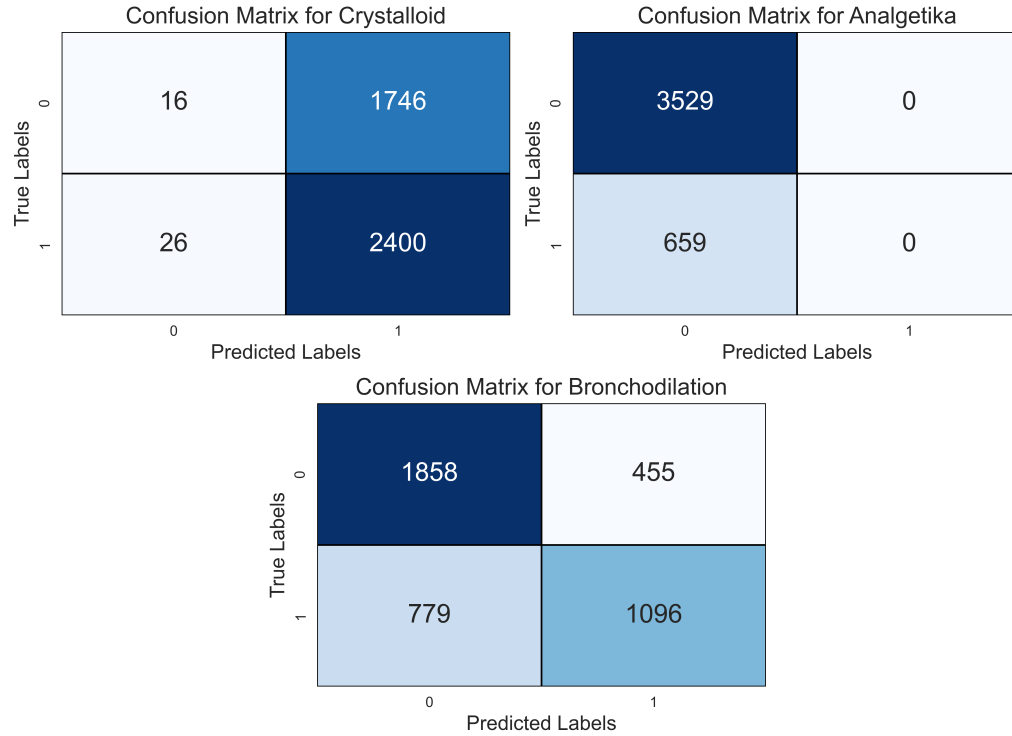


Figure 9.24: Confusion Matrix for Individual Labels with LR Algorithm

with negative cases. K-Nearest Neighbors struggled further, with a decrease in sensitivity to 58.3% and a much lower F1 score (63.2%), reflecting a poorer performance in balancing precision and recall.

Adaptive Boosting (ADA) and Logistic Regression (LR) performed weaker than other algorithms. ADA, while achieving a reasonable specificity of 88.6%, had the lowest sensitivity (48.4%), making it unreliable and unusable in rescue situations. LR, with the lowest overall metrics (accuracy: 69.0%; F1 score: 53.2%), failed due to its limitations in handling complex and non-linear patterns in the data. In conclusion, XGB and RF were the top performers, with XGB having a slight edge due to its better sensitivity and precision balance, making it the most reliable model for recommending treatment paths for neurological complications. The comparative performance of all models and the evaluation metric can be seen in Figure 9.22.

Classification	Accuracy	Sensitivity	Specificity	Precision	F1 Score
XGB	83.7	72.7	89.7	79.4	75.9
RF	83.1	72.5	89.0	78.3	75.2
MLP	81.8	66.9	90.1	78.8	72.3
KNN	75.9	58.3	85.6	69.2	63.2
ADA	74.4	48.4	88.6	70.1	57.2
LR	69.0	50.8	78.7	56.0	53.2

Table 9.9: Performance of Medicine Recommendation Model - Neurological Complications. All Values are in Percentage

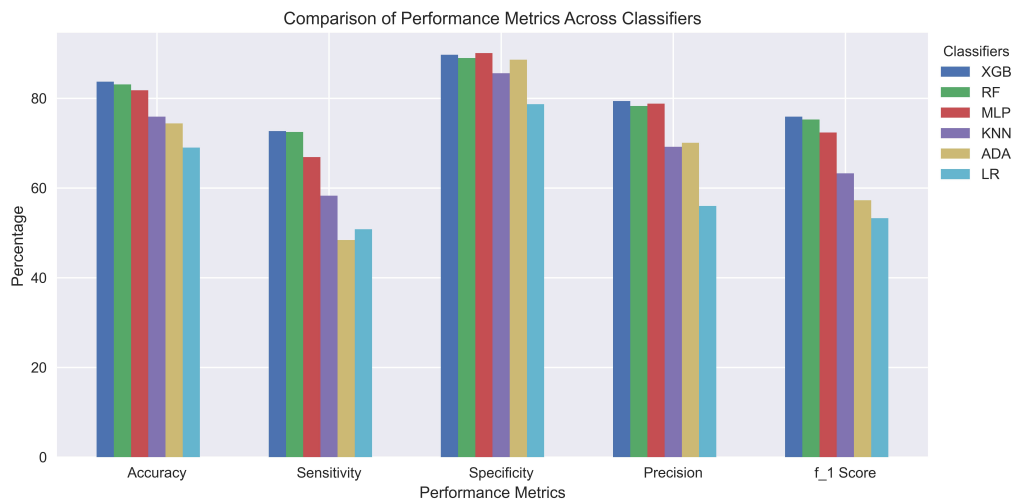


Figure 9.25: Comparative Analysis of Performance Metrics for Neurological Complication

The performance of the Logistic Regression algorithm can be further explained with the use of confusion matrix representation in Figure 9.26. For antiemetic medication prediction, it had a high false positive rate (655) and moderate false negatives (420). Although it could detect true positives (1,050), its lack of precision and balance in predictions highlights its struggle to differentiate between classes effectively. For crystalloid treatment, the model suffered from significant false positives (878) and a high number of false negatives (650), indicating its inability to establish a reliable decision boundary between the two classes. However, LR totally failed with sedative medication, detecting only a single true positive and false positive, with 823 false negatives. This suggests severe under-fitting and an inability to recognize patterns in this specific treatment category. XGB model, which provided the best performance and was significantly better for three medication recommendations. The algorithm demonstrated lower false positives and false negatives compared to LR, resulting in a more accurate classification of all predictions of medications, as can be found in

Figure 9.27. Although the model had challenges like higher true negative of 524 for Sedative, but it is caused by high data imbalance. In general, it can act as a reliable, quick and precise tool to recommend lifesaving treatments in high-stress rescue situations.

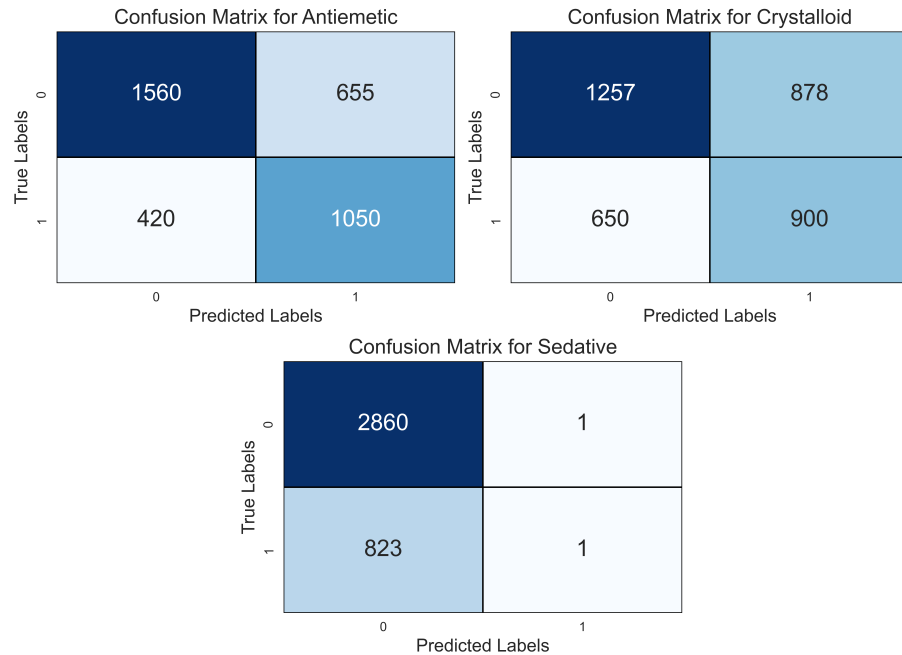


Figure 9.26: Confusion Matrix for Individual Labels with LR Algorithm

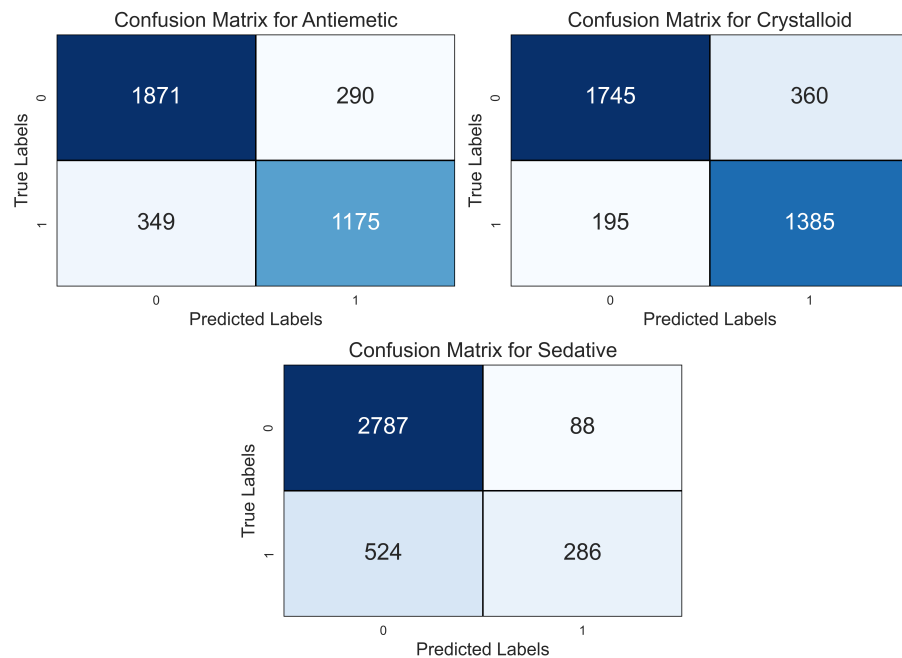


Figure 9.27: Confusion Matrix for Individual Labels with XGB Algorithm

9.3.4 Psychiatric Treatment Recommendation

The medications with the highest frequency of use for psychiatric patients were found to be the same as those for neurological patients. For the predictive analysis of these medications, Random Forest achieved the highest overall performance with an accuracy of 89.0%, a sensitivity of 88.9%, a specificity of 89.1%, a precision of 87.4%, and an F1 score of 88.1%. Its superior ability to handle complex decision boundaries and imbalanced data made it highly reliable for rescue scenarios, where accurate and robust recommendations are critical. XGBoost, with an accuracy of 87.6%, demonstrated competitive performance, particularly in sensitivity (87.2%) and specificity (88.0%). However, its precision (86.7%) and F1 score (86.9%) were slightly lower than Random Forest, limiting its effectiveness in highly complex situations.

The multilayer perceptron (MLP) algorithm also performed reasonably well, achieving an accuracy of 86.6%, a sensitivity of 86.0%, a specificity of 87.1%, a precision of 85.6%, and an F1 score of 85.8%. While MLP handled non-linear relationships effectively, it lagged slightly behind Random Forest and XGBoost in precision and F1 score, impacting its reliability in critical applications. Adaptive Boosting (ADA) displayed moderate performance, with an accuracy of 82.3%, a sensitivity of 82.4%, a specificity of 82.2%, a precision of 80.5%, and an F1 score of 81.4%. While it worked well on simpler patterns, its limited ability to generalize to complex decision spaces reduced its performance.

Logistic Regression (LR), with an accuracy of 81.4%, sensitivity of 82.0%, specificity of 80.9%, precision of 78.9%, and an F1 score of 80.4%, struggled to handle intricate relationships, which made it less suitable for scenarios requiring precise recommendations. Lastly, K-Nearest Neighbors (KNN) exhibited the lowest overall performance, with an accuracy of 80.6%, a sensitivity of 73.0%, a specificity of 87.4%, a precision of 83.8%, and an F1 score of 78.0%. The algorithm's susceptibility to noisy data and computational inefficiency in high-dimensional spaces made it less practical for rescue situations despite its high specificity.

Overall, the RF algorithm stood out as the best performing model, offering a balance of accuracy, robustness, and interpretability, making it ideal for recommending treatment paths in rescue situations. XGBoost closely followed, offering similar reliability with added computational efficiency. In Figure 9.28, a comparative analysis of performance metrics of all algorithms are shown.

Classification	Accuracy	Sensitivity	Specificity	Precision	F1 Score
RF	89.0	88.9	89.1	87.4	88.1
XGB	87.6	87.2	88.0	86.7	86.9
MLP	86.6	86.0	87.1	85.6	85.8
ADA	82.3	82.4	82.2	80.5	81.4
LR	81.4	82.0	80.9	78.9	80.4
KNN	80.6	73.0	87.4	83.8	78.0

Table 9.10: Performance of Medicine Recommendation Model - Psychiatric Complications. All Values are in Percentage

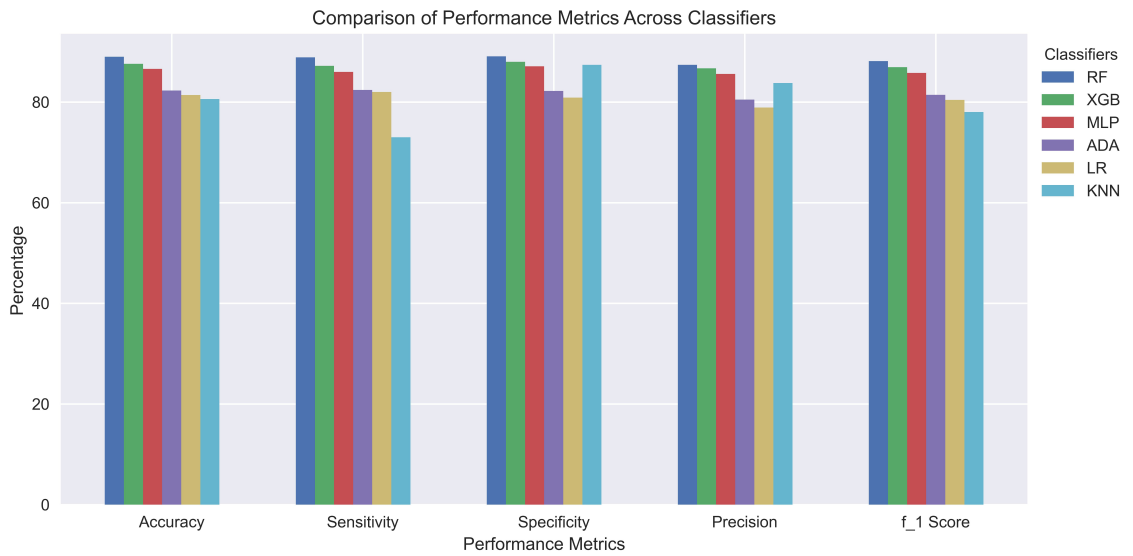


Figure 9.28: Comparative Analysis of Performance Metrics for Psychiatric Complication

The strongest and weakest algorithm found for the recommendation model were Random Forest and Logistic Regression, while the latter had a very poor specificity of 78.9%. For antiemetic medication The model correctly identified most negative cases and there were no false positives as can be found in Figure 9.29. However, a notable number of positive cases (86) were misclassified as negative and completely failed to predict any positive cases for antiemetic treatment. The algorithm showed high false positives (132) and false negatives (94) for crystalloid predictions, leading to poor specificity. For sedative prediction, the number of false positives (113) is much higher than the number of false negatives (21), suggesting a tendency to overpredict the positive class, which is not ideal for rescue medications. Compared to that, Random Forest had a better performance for the three medications in terms of false positive or false negative, as visible in Figure 9.30. It ensured that the model had fewer misclassifications and better sensitivity and specificity. Two specific reasons why the

RF algorithm performed better are its robustness to data noise and providing higher weight for incidences that have fewer occurrences.

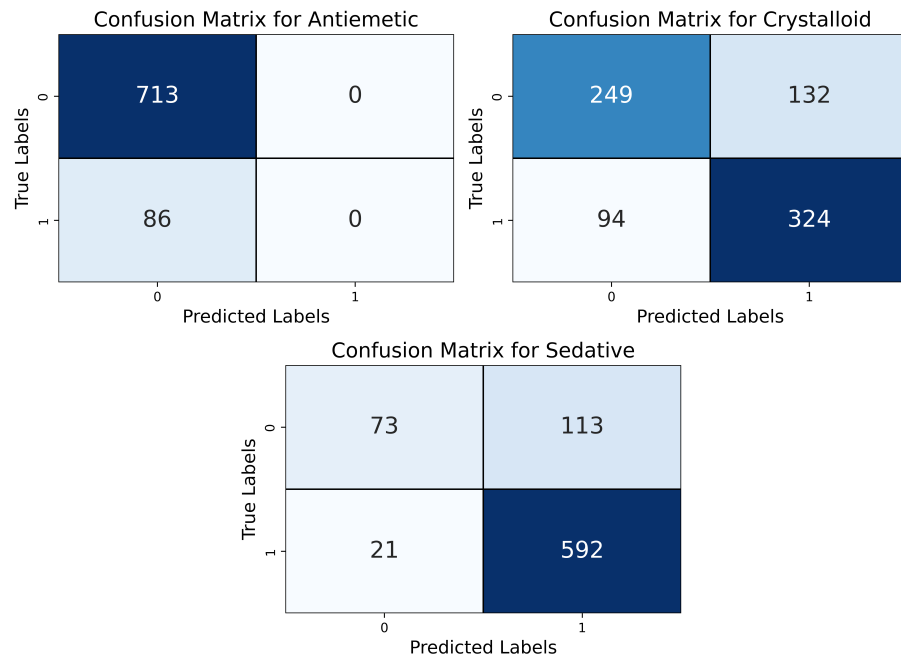


Figure 9.29: Confusion Matrix for Individual Labels with LR Algorithm

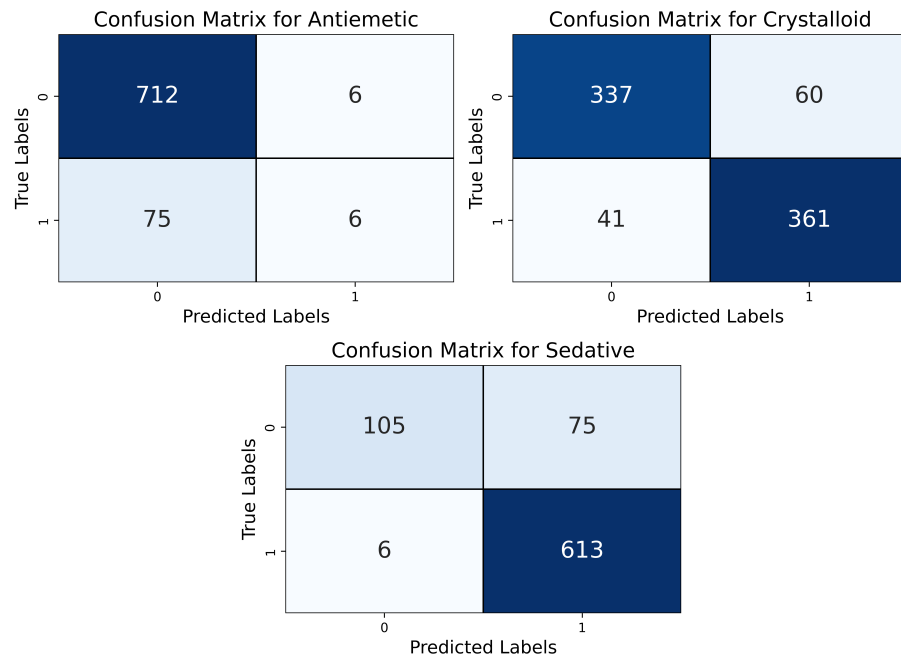


Figure 9.30: Confusion Matrix for Individual Labels with RF Algorithm

9.4 Performance of Software

Software testing is a significant approach in the software development lifecycle that ensures the successful functionality of a software program. It involves verifying and validating that a software application is free of bugs and meets the functional, non-functional, and user requirements set at the planning phase without any contrast. For the proposed rescue care assistance software, a testing environment is established to assess its performance according to the requirement analysis detailed in Section 8.1.1. In this study, three use cases served as test data to evaluate the effectiveness of the software. The data of the use cases, as can be seen in Table 9.11, were extracted from the original rescue events with the diagnostic information of the rescue personnel.

Table 9.11: PATIENT DATA FROM USE CASES

Patient Health Data	Usecase 1	Usecase 2	Usecase 3
Respiratory rate	18	12	11
Blood pressure	145/91	142/85	132/82
Mean areterial pressure	104.4	104	99
Pulse rate	105	106	83
Blood glucose	124	139	105
SpO2	97	100	86
Observation	Chest pain Respiratory ok No injury Mentally fit Conscious Not alcoholic	No Chest pain	No Chest pain
		Respiratory ok	Respiratory low
		No injury	Head injury
		Abdominal pain	Mentally unfit
		Pre-abdominal illness	Consciousness not ok
		Mentally unfit	Head discomfort
		Conscious	Pre-neurological illness
		Not alcoholic	Alcoholic

9.4.1 Use-Case 1

Original diagnosis: The patient was found with unexplained chest pain referring to cardiovascular complication.

The predictive analysis of the XGB and ANN models in Table 9.12 suggests that the patient is experiencing a cardiovascular complication. The patient had chest pain, which was the deciding factor. The second highest probability is for respiratory complications, as respiratory-related chest pain is a common symptom and can suggest a serious or potentially life-threatening condition [28].

Table 9.12: Health Complication Probability Prediction For Use Case - 1

Complication	Probability XGB (%)	Probability Keras (%)	Probability Pytorch (%)
Cardiovascular	95.3	100	98.94
Respiratory	70.76	66.18	76.81
Neurological	8.15	13.86	9.61
Psychiatric	7.28	18.8	18.07
Metabolic	7.02	2.92	12.32
Abdominal	5.9	9.93	15.35

9.4.2 Use-Case 2

Original diagnosis: The patient had appendicitis and cyst on the right side of abdomen referring to abdominal complication.

As can be seen in Table 9.13, the software outcome suggested that the patient had abdominal complications that matched the original diagnosis from rescue personnel. The deciding factor was the presence of abdominal pain which was diagnosed as appendicitis. The patient also had a history of abdominal illness. The other probabilities were not high enough except the outcome of PyTorch model for metabolic complication. Because, during the training period, the model found a close relation of abdominal diseases with metabolic complication. Although, the patient was not fit mentally, the software outcome showed a low probability of psychiatric complication as the other symptoms did not refer to that.

9.4.3 Use-Case 3

Original diagnosis: The patient had suicidal tendency which belongs to Psychiatric complication. Differential diagnosis: Laceration, Craniocerebral injury - Neurological complication.

The patient in use case 3 was not mentally fit and consciousness level was poor. The patient was also found to be alcoholic during rescue, which shows signs of mental instability. Due to mental illness, the patient had an accident that caused a serious head injury. The outcome demonstrated in Table 9.14 indicates that the patient

Table 9.13: Health Complication Probability Prediction For Use Case - 2

Complication	Probability XGB (%)	Probability Keras (%)	Probability Pytorch (%)
Abdominal	83.3	74.43	65.94
Cardiovascular	41.09	28.24	47.5
Respiratory	25.41	27.63	27.87
Psychiatric	20.34	32.23	34.56
Metabolic	10.26	38.41	63.24
Neurology	8.15	13.86	9.61

should be treated first for neurological complication, which is justified because the patient complained of head discomfort and the head injury should be prioritized first. In addition, the patient had a history of pre-neurological illness. An important insight into the outcome also suggests that although the patient had a low respiratory rate and oxygen saturation, the probability of respiratory complication is not very high. According to Goldman et al. in [60], patients with neurological injuries frequently exhibit irregular breathing patterns that can hinder oxygen from entering the bloodstream. Patient data taken in rescue events often show dynamic characteristics

Table 9.14: Health Complication Probability Prediction For Use Case - 3

Complication	Probability XGB (%)	Probability Keras (%)	Probability Pytorch (%)
Neurology	99.32	99.09	99.6
Psychiatric	84.29	96.23	89.99
Abdominal	23.77	22.09	8.76
Metabolic	21.78	52.79	74.49
Cardiovascular	18.87	0	44.38
Respiratory	11.8	15.52	27.87

because health vitals are generally not stable and change over time. So to implement this dynamic situation, the data for use case 3 was modified by adding an offset of 20% in the health vital values. The impact in the software output for the manipulation can be seen in Table 9.15.

Table 9.15: Health Complication Probability After Modifying Health Vital Data of Use Case - 3

Complication	Probability XGB (%)	Probability Keras (%)	Probability Pytorch (%)
Neurology	99.32	99.09	99.6
Psychiatric	80.98	94.82	89.78
Abdominal	5.65	22.14	9.99
Metabolic	11.57	36.5	33.45
Cardiovascular	25.46	0.01	43.33
Respiratory	49.02	14.79	27.87

The result found here is similar to the unmodified data indicating that the models were reliable and robust. Based on this research, robustness was found to be attributed to machine learning algorithms that generate output considering not only the individual feature values but also the interrelationships between features.

9.5 Performance in Hardware Platform

For each health complication, three machine learning models were chosen that were used to recognize situations. As per the previous section on the performance of the ML model, the best performing algorithms found were XGBoost and Random Forest. Initial development and testing of the models were conducted on the local computer using a CPU, employing Python frameworks. Afterwards, the models were deployed on the other accelerators: FPGA PYNQ Z2 both PS and PL sides, Microcontroller BLE Sense 33, and Raspberry Pi 5B. The CPU used in the thesis came with the following configuration:

- Processor: Intel Core-i9 14th generation that supports 32 threads through Intel's Hyper-Threading technology
- Architecture: X86 Complex Instruction Set Computing (CISC)
- Memory: 128 GB
- Operating System: Ubuntu 22.04.5 LTS
- Operating System Version: 64 bit

The most important evaluation criteria were to identify whether the hardware meets timing requirements for the recognition of critical health complication, which is done by determining the average inference time for the test data. Table 9.16 is created to highlight how the models performed for each complication in terms of inference time.

Based on the table, the summary of the performance can be explained as follows:

Table 9.16: Average Inference Time (milliseconds) Across Accelerators for Different ML Models and Health Complications

Complication	Model	CPU	Microcontroller	Rasp. Pi	FPGA-PS	FPGA-PL
Cardiovascular	RF	0.85	0.11	2.28	1.49	0.39
	XGB	0.50	0.22	0.48	0.95	0.37
Respiratory	RF	0.75	0.12	1.26	1.47	0.45
	XGB	0.55	0.27	0.90	0.96	0.45
Neurology	RF	1.51	0.11	1.09	1.45	0.66
	XGB	0.60	0.13	1.23	0.94	0.43
Psychiatry	RF	0.81	0.13	1.29	1.40	0.54
	XGB	0.65	0.15	0.59	0.90	0.53
Metabolic	RF	0.76	0.15	1.38	1.46	0.40
	XGB	0.50	0.22	0.66	1.00	0.44
Abdominal	RF	0.70	0.14	1.01	1.39	0.44
	XGB	0.50	0.21	0.41	0.94	0.40

- Cardiovascular Complications:** For cardiovascular complications, the microcontroller achieved the fastest inference for both Random Forest (0.11 ms) and XGBoost (0.22 ms) algorithm, outperforming all other platforms. The FPGA-PL accelerator also demonstrated strong performance, particularly for XGBoost (0.37 ms), reducing inference time by 26% compared to the CPU (0.50 ms). The Raspberry Pi, on the other hand, exhibited the highest inference time for Random Forest (2.28 ms), highlighting its limited capability for this complication.
- Respiratory Complications:** In the case of respiratory complications, the microcontroller again led with the lowest inference times for both Random Forest (0.12 ms) and XGBoost (0.27 ms). FPGA in PL side also performed well, achieving a time of 0.45 ms for both models, representing a significant improvement over Raspberry Pi (1.26 ms and 0.90 ms, respectively).
- Neurology Complications:** The microcontroller remained dominant, with times of 0.11 ms for Random Forest and 0.13 ms for XGBoost, while the FPGA-PL side provided competitive results, particularly for XGBoost, with an inference time of 0.43 ms, which was a 28% reduction compared to the CPU. . Accelerator like local computer CPU or Raspberry Pi had inference time close to 1 millisecond or above.
- Psychiatry Complications:** For psychiatric complications, the microcontroller demonstrated superior efficiency, with inference times of 0.13 ms and 0.15 ms for Random Forest and XGBoost, respectively. The FPGA-PL also reduced inference times compared to the CPU by 33% for Random Forest and 18% for XGBoost. The Zynq SoC and Raspberry pi had as expected, higher inference time than other accelerators.

- **Metabolic Complications:** The microcontroller using TinyML platform achieved again the fastest inference times (0.15 ms for Random Forest and 0.22 ms for XGBoost), while the FPGA-PL demonstrated significant reductions in inference time compared to the CPU, achieving 0.40 ms and 0.44 ms, respectively. Meanwhile, the Raspberry Pi showed the slowest performance for Random Forest (1.38 ms), highlighting its unsuitability for low-latency applications.
- **Abdominal Complications:** For abdominal complications, the microcontroller again provided the lowest inference times (0.14 ms for Random Forest and 0.21 ms for XGBoost). The FPGA-PL followed, with times of 0.44 ms and 0.40 ms, representing reductions of 37% and 20% compared to the CPU having inference time . The Raspberry Pi displayed moderate performance for XGBoost (0.41 ms) but was slower for Random Forest (1.01 ms).

In summary, the microcontroller consistently demonstrated the fastest inference times across all health complications and ML models, making it the most suitable platform for real-time, life-critical rescue scenarios. The FPGA-PL also exhibited strong performance, achieving substantial reductions in inference times compared to the CPU and Raspberry Pi, making it a competitive choice for accelerated ML inference. The Raspberry Pi, while affordable and versatile, showed higher inference times, indicating its limitations for time-sensitive applications. The FPGA-PS generally performed better than Raspberry Pi but lagged behind the FPGA-PL and microcontroller. These findings highlight the importance of selecting hardware platforms that balance computational efficiency and latency for deploying ML models in medical emergencies.

The microcontroller used in this research had a random access memory (RAM) of only 256 kilobytes, which can be used to run ML computation and intermediate mathematical operations. Although there are some resource limitations of microcontroller in terms of processing power, memory, and parallelism, they are purpose-built for specific, lightweight computations with minimal overhead. They lack the complex operating systems and multitasking capabilities of general-purpose processors (like CPUs and RPI), allowing them to utilize resources exclusively to inference tasks. Such a strategy is useful for small-scale, decision-tree-based models like RF and XGB, which rely on numerous lightweight operations (comparisons and conditional branching) rather than heavy arithmetical calculations like ANN models. In addition, both RF and XGB models are often quantized or pruned when deployed on microcontrollers, further reducing computation and memory demands. This hardware-specific optimization amplifies the performance benefits of microcontrollers for such models. The PL of FPGA performed also well because of its parallel processing capability, which is particularly effective for the tree-based structure of RF and XGB. Each node traversal or tree evaluation can be pipelined or executed in parallel, reducing overall inference time. Also, it allows for the creation of application-specific designs tailored to the model's structure, such as custom tree traversal units which makes the decision boosting faster. The CPU of local computer and Raspberry Pi had a higher inference time due to their general purpose architecture which is not specifically designed for

real-time ML inference. Their hierarchical memory design relies heavily on caching for performance. But RF and XGB models often require irregular memory access patterns (e.g., accessing tree nodes), which can lead to cache misses and frequent DRAM access, significantly slowing inference times [53]. Overall, the inference times of all resource-constrained devices were acceptable as they were very fast in generating predictive outcomes. The difference in inference time is not very high and is all in the millisecond range. It is a proof that the situation recognition algorithm can be highly effective in rescue scenarios if deployed in portable accelerators.

Conclusions and Future Work

The research presented in this dissertation was financially supported by the Federal Ministry of Education and Research, Germany, through the KIRETT project. This is a KMU-Innovative funding initiative that supports the collaboration of small-medium companies with research organizations. The aim of this doctoral thesis was to implement machine learning-based algorithms for detecting health complications and recommending treatments, with a focus on identifying the feasibility of such a solution in real-world scenarios through an embedded software solution and portable hardware deployment. This work bridges the gap between theoretical advancements in machine learning and their practical application in real-world healthcare settings. As rescue situations are quite different from clinical conditions, the research proposes a reliable medical support system that is scalable and sustainable for any challenging environment like emergencies. However, medical conditions require careful consideration and relying solely on AI solutions is not advisable. Instead, this research emphasizes the role of AI as an effective support system aimed at improving rescue care, ultimately contributing to improved patient survival rates.

10.1 Summary of the Thesis

The thesis outlines a comprehensive workflow that begins with the management of rescue data on individual health issues, then uses these data to develop machine learning algorithms designed as a medical decision support system. In Chapter 3, a presentation of the raw rescue data is provided highlighting the information that can be used for the analysis of health predictions. This thesis also clarifies the technical concepts employed, such as natural language processing, machine learning algorithms, and hardware compatibility, which shaped the engineering contribution to the research. A major part of the research work was dedicated to the organization and management

of the rescue data explained in Chapters 5 and 6. Data was the backbone of this research and also the major component for algorithm development. The 10 year rescue record was huge and cannot be handled manually, so different algorithms were developed to manage the data, extract information, and preprocess. Several challenges were also faced during data management, as explained in those chapters. The difficulties could not be solved only through algorithms but required multiple discussions with medical experts, rescue station employees, and data providers working on the KIRETT project. Some of the health complications required advanced data analysis as no predictive analysis of these complications can be conducted in a traditional ML-based setup. So, a questionnaire was developed that represents the symptoms of the patients that correspond to the patient’s diagnostic information recorded in rescue events.

The next part of the thesis was dedicated to the development of software to identify the best ML model for situation recognition and treatment recommendation, as can be seen in Chapter 7. In total, seven machine learning models were carefully chosen for complication prediction and six for medication recommendation. According to the software output, the ML algorithms demonstrated exceptional accuracy in detecting a range of health complications, such as cardiovascular irregularities, respiratory disorders, and metabolic abnormalities. Key performance metrics, such as precision, recall, F1 score, and accuracy, consistently demonstrated high benchmarks for each individual complication. For example: Cardiovascular anomaly detection achieved an accuracy greater than 96% and a high precision greater than 98% with the Random Forest model. The best models identified for other complications also had the metrics over 85% which is satisfactory for a medical decision support system. Models designed for treatment recommendations were challenged to predict multiple medications. Certain medications were used less often, leading to significant data imbalance and adversely affecting the performance of the model. However, each of these models achieved high accuracy and precision scores that exceeded 80%, confirming their suitability for future applications. Advanced ensemble techniques, such as Random Forests and Extreme Gradient Boosting, were particularly effective in classification tasks, while neural networks excelled in more complex scenarios, such as multi-label medicine recommendations. The medication recommendation models reduced decision-making delays for healthcare providers by providing accurate suggestions in less than 1 second, thereby improving patient care.

The thesis also aimed to demonstrate the real-time applicability of the developed models through rescue care assistance software and deployment in portable accelerators, as explained in Chapter 8. The software reliably outputs a percentage-based likelihood of health complications when patient data are provided, utilizing top-performing machine learning models tailored for each specific complication. In addition, it can be used on any computing device with enough hardware resources, like computers used in the rescue vehicle or a handheld tablet of rescuers. The inference time results from the hardware deployment of the situation recognition software was in the millisecond range, highly suitable in dynamic rescue scenarios where the likelihood fluctuates every second as new health information is recorded. Such a system can be

considered as a significant improvement in real-time health monitoring, offering early detection capabilities that are crucial to saving lives in emergency scenarios. Some certain challenges were also faced during the research, mostly due to the disorganized rescue record received from the rescue station. As the data size was huge, outlier detection and data filtering could be done without a specially designed algorithm. The algorithm also needed to understand the context of each text-extracted feature where advanced NLP techniques were implemented. A significant improvement was observed when such features were created and applied to the ML models. Despite the effectiveness achieved through AI algorithms, real-world deployment scenarios may introduce edge cases that require further validation. A distinct limitation became evident when situation recognition models encountered difficulties operating on low-power devices, such as microcontrollers and FPGAs. There were environmental dependency mismatch issues for some Python libraries that could only be solved by downgrading to the deprecated version of the library packages. The AI algorithms produced results that were not fixed, showing changes in performance metrics that varied between 1 and 3% depending on the hardware used or the timing of execution.

10.2 Future Enhancement

The research conducted in this thesis provided a solid proof of how ML algorithms can provide effective support in rescue events. It also provided an opportunity to expand the support software in other medical domains where emergency patients are involved. This will be possible if the designed ML system can be integrated with IoT ecosystems to enable seamless communication between devices, patients, and healthcare providers. Leveraging cloud computing to handle computationally intensive tasks, offloading edge devices, and ensuring scalability are other potentials that need to be explored. Rescuers attending emergency events do not have medical experience like doctors; instead, they are trained in first-aid treatments. Incorporating explainable AI methods can be beneficial here to enhance trust and transparency in ML-driven recommendations. An obvious application that can provide further support is the use of a large language model that can be customized with prediction-based ML algorithms to obtain guided context and follow-up recommendations. Patients' condition can change rapidly in every second, which forces rescuers to change their diagnostic approach and treatment path. When developers incorporate LLM into their systems, they gain an advantage during rapidly changing health situations, as the LLM is capable of swiftly adjusting to these dynamic conditions and offering up-to-date text recommendations.

There is also a distinct possibility for advancement in creating a wearable device utilizing the engineered ML algorithms, meant for patient use in home settings or within hospital environments. This kind of wearable could contribute to telemedicine applications or to the treatment of people with chronic diseases. The situation detection algorithm researched here worked with complication groups as the number of diseases was too high to use for prediction. However, the algorithm can be implemented

into rescue systems to detect frequent health complications during emergencies, such as brain strokes or cardiac arrests, and provide guided treatment.

Publications

Journal

- Ahammed, A. S., Ezekiel, A. M., and R. Obermaisser. "Time-Efficient Identification Procedure for Neurological Complications of Rescue Patients in an Emergency Scenario Using Hardware-Accelerated Artificial Intelligence Models." *Algorithm*, vol. 16, 2023.
- Oleh, U., Obermaisser, R., and A. S. Ahammed. "A Review of Recent Techniques for Human Activity Recognition: Multimodality, Reinforcement Learning, and Language Models." *Algorithms*, vol. 17, no. 10, 2024, p. 434.

Conference Paper

- Ahammed, A. S., Donthireddy, S. R., and R. Obermaisser. "Detection of Respiratory Emergency Situation of Rescue Patients with Machine Learning Algorithms." *IECON 2022–48th Annual Conference of the IEEE Industrial Electronics Society*, 2022, p. 6.
- Zenkert, J., Weber, C., Nadeem, M., Bender, L., Fathi, M., Ahammed, A. S. "Kirett—a wearable device to support rescue operations using artificial intelligence to improve first aid." *2022 IEEE International Smart Cities Conference (ISC2)*, p. 1–4.
- Ahammed, A. S., Ezekiel, M., and R. Obermaisser. "A Novel Analysis of Performance and Inference Time of Machine Learning Models to Detect Cardiovascular Emergency Situations of Rescue Patients." *2022 International Conference on Artificial Intelligence of Things (ICAIoT)*, 2022, pp. 1–6.
- Ahammed, A. S., and R. Obermaisser. "Smart Health Software to Support Rescue Personnel in Emergency Situations." *2024 International Conference on Smart Systems and Technologies (SST)*, 2024, pp. 165–170.
- Ahammed, A. S., Hossain, M. S. A., and R. Obermaisser. "A Computer Vision Approach for Autonomous Cars to Drive Safe at Construction Zones." *arXiv preprint arXiv:2409.15809*, 2024.
- Hossain, M. S. A., Ahammed, A. S., Biswas, D. P., and R. Obermaisser. "Impact Analysis of Data Drift Towards the Development of Safety-Critical Automotive System." *2024 International Symposium ELMAR*, 2024, pp. 325–330.

Acronyms

- **NLP**: Natural Language Processing
- **AI**: Artificial Intelligence
- **XGB**: Xtreme Gradient Boosting
- **BP**: Blood Pressure
- **IV**: Intra Venous
- **ML**: Machine Learning
- **RF**: Random Forest
- **SVM**: Support Vector Machine
- **LR**: Logistic Regression
- **NB**: Naive Bayes
- **MLBDD**: Machine Learning Based Disease Diagnosis
- **LLM**: Large Language Model
- **ANN**: Artificial Neural Network
- **CNN**: Convolutional Neural Network
- **ANN**: Recurrent Neural Network
- **LSTM**: Long Short Term Memory
- **GAN**: Generative Adversarial Networks
- **GUI**: Graphical User Interface
- **ORDBMS**: Object Relational Database Base Management System
- **API**: Application Programming Interface
- **RFECV**: Recursive Feature Elimination Cross Validation

- **IQR**: Inter **Q**uartile **R**ange
- **MLSMOTE**: Multi **L**evel **S**ynthetic **M**inority **O**versampling **T**Echnique
- **SFS**: Sequential **F**eature **S**elector
- **GPU**: **G**raphic **P**rocessing **U**nit
- **CPU**: Central **P**rocessing **U**nit
- **FPGA**: **F**ield **P**rogrammable **G**ate **A**rray
- **VTA**: **V**ersatile **T**ensor **A**ccelerator
- **ROC**: **R**eciever **O**perator **C**haracterstic
- **AUC**: **A**rea **U**nder the **C**urve
- **CPR**: **C**ardio **P**ulmonary **R**esuscitation
- **SBP**: **S**ystolic **B**lood the **P**ressure
- **DBP**: **D**iastolic **B**lood the **P**ressure
- **FAST**: **F**ace **A**rm **S**peech **T**ime
- **GCS**: **G**lasgow **C**oma **S**cale
- **CNS**: Central **N**ervous **S**ystem
- **RBF**: **R**adial **B**asis **F**unction
- **CISC**: **C**omplex **I**nstruction **S**et **C**omputing
- **PS**: **P**rocessing **S**ubsystem
- **PL**: **P**rogrammable **L**ogic
- **SoC**: **S**ystem on the **C**hip
- **RTL**: **R**egister **T**ransfer **L**evel
- **IP**: **I**ntellectual **P**roperty
- **AXI**: **A**dvanced **eX**tensible **I**nterface
- **GPIO**: **G**eneral **P**urpose **I**nput **O**utput



Appendix

A.1 Appendix A: List of Hyperparameters Relevant to Health Complications

A.1.1 Situation Recognition Models

Table A.1: Hyperparameter Grid for XGBoost

Parameter	Grid Values
Min_child_weight	3, 5, 8, 10
Gamma	0.8, 1, 1.2, 0.5, 2, 3
Subsample	0.6, 0.7, 0.8, 1.0, 0.9
Colsample_bytree	0.6, 0.8, 1.0, 0.9
Max_depth	1, 2, 4, 5, 7, 8

Table A.2: Tuned Hyperparameters of Health Complications for XGBoost Algorithm

Parameter	Cardiovascular	Respiratory	Neurological	Psychiatric	Metabolic	Abdominal
Min_child_weight	10	5	3	3	3	10
Gamma	2	1	1.2	2	0.5	0.8
Subsample	0.9	0.7	0.7	0.9	0.8	0.9
Colsample_bytree	0.6	0.6	1	0.6	0.6	0.9
Max_depth	2	2	4	4	2	2

Table A.3: Hyperparameter Grid for Random Forest

Parameter	Grid Values
Max_features	auto, sqrt
n_estimators	80, 150, 200
Max_samples	0.6, 0.7, 0.8
Max_depth	10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110
Criterion	gini, entropy
Min_samples_split	2, 3, 5, 8

Table A.4: Tuned Hyperparameters of Health Complications for Random Forest Algorithm

Parameter	Cardiovascular	Respiratory	Neurological	Psychiatric	Metabolic	Abdominal
criterion	entropy	gini	gini	entropy	entropy	entropy
Max_depth	10	20	30	10	100	110
Max_features	sqrt	sqrt	sqrt	sqrt	sqrt	sqrt
Max_samples	0.7	0.6	0.6	0.7	0.6	0.6
Min_samples_split	3	8	8	8	8	5
n_estimators	150	150	80	150	200	200

Table A.5: Hyperparameter Grid for Support Vector Machine

Parameter	Grid Values
Kernel	rbf, linear
Gamma	0.1, 0.01, 0.001
C	1.0, 0.1, 10

Table A.6: Tuned Hyperparameters of Health Complications for Support Vector Machine Algorithm

Parameter	Cardiovascular	Respiratory	Neurological	Psychiatric	Metabolic	Abdominal
Kernel	rbf	rbf	rbf	rbf	rbf	rbf
Gamma	0.001	0.001	0.1	0.01	0.1	0.1
C	1.0	1.0	1.0	10	1.0	1.0

Table A.7: Hyperparameter Grid for K-Nearest Neighbors

Parameter	Grid Values
n_neighbors	2, 4, 6, 8
weights	uniform, distance
metric	euclidean, manhattan, minkowski

Table A.8: Tuned Hyperparameters of Health Complications for K-Nearest Neighbors (KNN) Algorithm

Parameter	Cardiovascular	Respiratory	Neurological	Psychiatric	Metabolic	Abdominal
n_neighbors	8	4	6	8	8	8
weights	distance	uniform	uniform	uniform	uniform	distance
metric	euclidean	euclidean	euclidean	euclidean	manhattan	manhattan

Table A.9: Hyperparameter Grid for Logistic Regression

Parameter	Grid Values
C	0.001, 0.01, 0.1, 1, 10, 100, 1000
Penalty	l1, l2
Solver	newton-cg, lbfgs, liblinear

Table A.10: Tuned Hyperparameters of Health Complications for Logistic Regression Algorithm

Parameter	Cardiovascular	Respiratory	Neurological	Psychiatric	Metabolic	Abdominal
C	0.001	0.001	0.1	0.1	0.01	0.001
Penalty	l1	l1	l2	l2	l1	l2
Solver	liblinear	liblinear	newton-cg	newton-cg	liblinear	liblinear

Table A.11: Hyperparameter Grid for Naive Bayes

Parameter	Grid Values
Var_smoothing	1e-09, 1e-08, 1e-07

Table A.12: Tuned Hyperparameters of Health Complications for Naive Bayes Algorithm

Parameter	Cardiovascular	Respiratory	Neurological	Psychiatric	Metabolic	Abdominal
Var_smoothing	1e-09	1e-09	1e-09	1e-09	1e-09	1e-09

A.1.2 Treatment Recommendation Models

Table A.13: Hyperparameter Grid for XGB Algorithm

Parameter	Grid Values
Estimator__max_depth	3, 5, 7
Estimator__learning_rate	0.01, 0.1, 0.2
Estimator__n_estimators	50, 100, 150
Estimator__subsample	0.7, 0.8, 0.9
Estimator__colsample_bytree	0.7, 0.8, 1.0

Table A.14: Tuned Hyperparameters of Treatment Recommendation Models with XGB Algorithm

Parameter	Cardiovascular	Respiratory	Neurological	Psychiatric
Estimator__colsample_bytree	1	1	1	0.8
Estimator__learning_rate	0.01	0.1	0.1	0.1
Estimator__max_depth	7	7	7	7
Estimator__n_estimators	150	100	150	100
Estimator__subsample	0.7	0.8	0.8	0.7

Table A.15: Hyperparameter Grid for Random Forest Algorithm

Parameter	Grid Values
Estimator__n_estimators	50, 100, 150
Estimator__max_depth	None, 10, 20, 30
Estimator__min_samples_split	2, 5, 10
Estimator__min_samples_leaf	1, 2, 4

Table A.16: Tuned Hyperparameters of Treatment Recommendation Models with Random Forest Algorithm

Parameter	Cardiovascular	Respiratory	Neurological	Psychiatric
Estimator__max_depth	10	20	None	None
Estimator__min_samples_leaf	2	2	4	1
Estimator__min_samples_split	5	10	10	10
Estimator__n_estimators	50	150	150	100

Table A.17: Hyperparameter Grid for Logistic Regression Algorithm

Parameter	Grid Values
Estimator__C	(−4, 4, 10)

Table A.18: Tuned Hyperparameters of Treatment Recommendation Models with Logistic Regression Algorithm

Parameter	Cardiovascular	Respiratory	Neurological	Psychiatric
Estimator__C	2.783	21.544	0.046	0.359

Table A.19: Hyperparameter Grid for Multi-Layer Perceptron Algorithm

Parameter	Grid Values
Estimator__hidden_layer_sizes	(50,), (100,), (100, 50), (50, 25)
Estimator__activation	relu, tanh
Estimator__solver	adam, SGD
Estimator__learning_rate	constant, adaptive
Estimator__alpha	0.0001, 0.001, 0.01

Table A.20: Tuned Hyperparameters of Treatment Recommendation Models with Multi-Layer Perceptron Algorithm

Parameter	Cardiovascular	Respiratory	Neurological	Psychiatric
Estimator__activation	relu	relu	relu	relu
Estimator__alpha	0.0001	0.0001	0.0001	0.0001
Estimator__hidden_layer_sizes	(100, 50)	(100, 50)	(100, 50)	(100, 50)
Estimator__learning_rate	constant	constant	constant	constant
Estimator__solver	adam	adam	adam	adam

Table A.21: Hyperparameter Grid for AdaBoost Algorithm

Parameter	Grid Values
Estimator__n_estimators	50, 100, 150
Estimator__learning_rate	0.01, 0.1, 0.5, 1.0

Table A.22: Tuned Hyperparameters of Treatment Recommendation Models with AdaBoost Algorithm

Parameter	Cardiovascular	Respiratory	Neurological	Psychiatric
Estimator__n_estimators	50	50	150	50
Estimator__learning_rate	1.0	0.5	0.5	0.5

Table A.23: Hyperparameter Grid for K-Nearest Neighbors Algorithm

Parameter	Grid Values
Estimator__n_neighbors	3, 5, 7, 9
Estimator__weights	uniform, distance
Estimator__metric	euclidean, manhattan, minkowski

Table A.24: Tuned Hyperparameters of Treatment Recommendation Models with K-Nearest Neighbors Algorithm

Parameter	Cardiovascular	Respiratory	Neurological	Psychiatric
<code>Estimator__metric</code>	euclidean	euclidean	euclidean	manhattan
<code>Estimator__n_neighbors</code>	9	9	7	9
<code>Estimator__weights</code>	distance	uniform	distance	uniform

A.2 Appendix B: Cross-validation Results

A.2.1 Situation Recognition Models

Table A.25: Cross Validation Result of ML Models for Cardiovascular Complication

Model	Metrics	Fold 1	Fold 2	Fold 3	Average
XGB	Accuracy	96.420%	96.686%	96.686%	96.597%
	Precision	96.559%	96.749%	96.781%	96.696%
	Recall	96.420%	96.686%	96.686%	96.597%
	F1 Score	96.410%	96.680%	96.678%	96.590%
RF	Accuracy	96.414%	96.310%	96.726%	96.483%
	Precision	96.496%	96.510%	96.701%	96.569%
	Recall	96.426%	96.420%	96.674%	96.507%
	F1 Score	96.377%	96.348%	96.738%	96.488%
SVC	Accuracy	94.411%	94.566%	94.249%	94.409%
	Precision	94.413%	94.576%	94.250%	94.413%
	Recall	94.411%	94.566%	94.249%	94.409%
	F1 Score	94.408%	94.562%	94.247%	94.406%
KNN	Accuracy	96.241%	96.247%	96.431%	96.306%
	Precision	96.295%	96.296%	96.470%	96.354%
	Recall	96.241%	96.247%	96.431%	96.306%
	F1 Score	96.235%	96.241%	96.427%	96.301%
ANN	Accuracy	95.641%	95.796%	95.623%	95.687%
	Precision	99.143%	99.212%	99.301%	99.219%
	Recall	91.463%	91.735%	91.276%	91.491%
	F1 Score	95.148%	95.327%	95.120%	95.198%
LR	Accuracy	94.266%	94.000%	94.214%	94.160%
	Precision	94.288%	94.020%	94.225%	94.178%
	Recall	94.266%	94.000%	94.214%	94.160%
	F1 Score	94.269%	94.003%	94.216%	94.163%
NB	Accuracy	93.856%	93.804%	93.873%	93.845%
	Precision	93.887%	93.835%	93.904%	93.875%
	Recall	93.856%	93.804%	93.873%	93.845%
	F1 Score	93.860%	93.808%	93.877%	93.848%

Table A.26: Cross Validation Result of ML Models for Respiratory Complication

Model	Metrics	Fold 1	Fold 2	Fold 3	Average
XGB	Accuracy	96.749%	96.680%	96.691%	96.707%
	Precision	96.845%	96.758%	96.782%	96.795%
	Recall	96.749%	96.680%	96.691%	96.707%
	F1 Score	96.742%	96.674%	96.685%	96.700%
RF	Accuracy	96.663%	96.564%	96.535%	96.587%
	Precision	96.785%	96.541%	96.576%	96.634%
	Recall	96.726%	96.512%	96.483%	96.574%
	F1 Score	96.720%	96.547%	96.482%	96.583%
SVC	Accuracy	94.671%	94.283%	94.434%	94.463%
	Precision	94.682%	94.292%	94.437%	94.470%
	Recall	94.671%	94.283%	94.434%	94.463%
	F1 Score	94.667%	94.279%	94.431%	94.459%
KNN	Accuracy	96.513%	96.154%	96.241%	96.303%
	Precision	96.547%	96.210%	96.288%	96.349%
	Recall	96.513%	96.154%	96.241%	96.303%
	F1 Score	96.508%	96.148%	96.236%	96.297%
ANN	Accuracy	95.387%	95.796%	95.375%	95.519%
	Precision	98.551%	98.886%	98.473%	98.636%
	Recall	91.483%	92.049%	91.531%	91.688%
	F1 Score	94.885%	95.345%	94.875%	95.035%
LR	Accuracy	93.712%	94.156%	94.203%	94.024%
	Precision	93.731%	94.167%	94.221%	94.040%
	Recall	93.712%	94.156%	94.203%	94.024%
	F1 Score	93.715%	94.158%	94.205%	94.026%
NB	Accuracy	93.880%	94.145%	93.943%	93.989%
	Precision	93.920%	94.187%	93.962%	94.023%
	Recall	93.880%	94.145%	93.943%	93.989%
	F1 Score	93.884%	94.149%	93.946%	93.993%

Table A.27: Cross Validation Result of ML Models for Neurological Complication

Model	Metrics	Fold 1	Fold 2	Fold 3	Average
XGB	Accuracy	85.910%	86.228%	86.434%	86.191%
	Precision	85.984%	86.304%	86.505%	86.264%
	Recall	85.910%	86.228%	86.434%	86.191%
	F1 Score	85.908%	86.226%	86.432%	86.188%
RF	Accuracy	86.200%	86.116%	86.434%	86.250%
	Precision	86.279%	86.165%	86.497%	86.314%
	Recall	86.191%	86.116%	86.499%	86.269%
	F1 Score	86.195%	86.112%	86.420%	86.242%
SVC	Accuracy	84.322%	84.490%	83.948%	84.254%
	Precision	84.332%	84.514%	83.949%	84.265%
	Recall	84.322%	84.490%	83.948%	84.254%
	F1 Score	84.321%	84.487%	83.948%	84.252%
KNN	Accuracy	83.612%	84.004%	83.192%	83.603%
	Precision	84.222%	84.713%	83.867%	84.267%
	Recall	83.612%	84.004%	83.192%	83.603%
	F1 Score	83.541%	83.925%	83.111%	83.526%
ANN	Accuracy	85.976%	85.882%	86.350%	86.069%
	Precision	84.421%	84.518%	84.373%	84.437%
	Recall	88.024%	87.648%	89.020%	88.231%
	F1 Score	86.185%	86.054%	86.634%	86.291%
LR	Accuracy	85.499%	85.686%	85.742%	85.643%
	Precision	85.552%	85.725%	85.812%	85.696%
	Recall	85.499%	85.686%	85.742%	85.643%
	F1 Score	85.497%	85.685%	85.739%	85.641%
NB	Accuracy	80.678%	81.024%	81.211%	80.971%
	Precision	82.129%	82.320%	82.662%	82.370%
	Recall	80.678%	81.024%	81.211%	80.971%
	F1 Score	80.444%	80.819%	80.986%	80.750%

Table A.28: Cross Validation Results for Psychiatric Complication

Model	Metric	Fold 1	Fold 2	Fold 3	Average
XGB	accuracy	88.392%	88.482%	88.550%	88.475%
	precision	88.441%	88.526%	88.699%	88.556%
	recall	88.392%	88.482%	88.550%	88.475%
	f1_score	88.401%	88.491%	88.565%	88.486%
RF	accuracy	87.444%	87.466%	87.285%	87.398%
	precision	87.616%	87.373%	87.478%	87.489%
	recall	87.602%	87.556%	87.240%	87.466%
	f1_score	87.549%	87.466%	87.388%	87.468%
SVC	accuracy	86.066%	85.772%	85.840%	85.893%
	precision	86.064%	85.781%	85.847%	85.897%
	recall	86.066%	85.772%	85.840%	85.893%
	f1_score	86.065%	85.776%	85.843%	85.895%
KNN	accuracy	86.134%	85.411%	85.727%	85.757%
	precision	86.359%	85.497%	85.974%	85.943%
	recall	86.134%	85.411%	85.727%	85.757%
	f1_score	86.153%	85.426%	85.747%	85.775%
ANN	Accuracy	88.008%	87.624%	87.850%	87.827%
	Precision	90.017%	89.597%	90.700%	90.105%
	Recall	87.326%	87.032%	86.195%	86.851%
	F1 Score	88.651%	88.296%	88.390%	88.446%
LR	accuracy	87.037%	85.795%	85.953%	86.262%
	precision	87.219%	86.174%	86.235%	86.542%
	recall	87.037%	85.795%	85.953%	86.262%
	f1_score	87.050%	85.807%	85.967%	86.275%
NB	accuracy	85.908%	86.043%	85.456%	85.802%
	precision	86.254%	86.328%	85.736%	86.106%
	recall	85.908%	86.043%	85.456%	85.802%
	f1_score	85.926%	86.062%	85.476%	85.821%

Table A.29: Cross Validation Results for Metabolic Complication

Model	Metric	Fold 1	Fold 2	Fold 3	Average
XGB	accuracy	86.795%	87.589%	86.850%	87.078%
	precision	86.822%	87.708%	86.914%	87.148%
	recall	86.795%	87.589%	86.850%	87.078%
	f1_score	86.708%	87.477%	86.746%	86.977%
RF	accuracy	85.790%	85.757%	87.027%	86.191%
	precision	85.946%	85.905%	86.787%	86.213%
	recall	85.643%	85.816%	86.436%	85.965%
	f1_score	85.394%	86.085%	85.953%	85.811%
SVC	accuracy	67.622%	68.174%	68.056%	67.950%
	precision	69.733%	70.873%	70.878%	70.495%
	recall	67.622%	68.174%	68.056%	67.950%
	f1_score	64.438%	64.857%	64.633%	64.643%
KNN	accuracy	83.959%	82.861%	84.161%	83.660%
	precision	84.122%	82.877%	84.341%	83.780%
	recall	83.959%	82.861%	84.161%	83.660%
	f1_score	83.758%	82.708%	83.958%	83.475%
ANN	Accuracy	80.177%	80.910%	82.299%	81.129%
	Precision	79.086%	80.843%	81.630%	80.519%
	Recall	71.800%	71.519%	74.684%	72.668%
	F1 Score	75.267%	75.896%	78.002%	76.388%
LR	accuracy	72.674%	72.134%	72.104%	72.304%
	precision	73.134%	72.745%	72.459%	72.779%
	recall	72.674%	72.134%	72.104%	72.304%
	f1_score	71.458%	70.730%	70.885%	71.025%
NB	accuracy	72.467%	72.370%	73.582%	72.806%
	precision	72.277%	72.121%	73.368%	72.588%
	recall	72.467%	72.370%	73.582%	72.806%
	f1_score	71.936%	72.064%	73.301%	72.433%

Table A.30: Cross Validation Results for Abdominal Complication

Model	Metric	Fold 1	Fold 2	Fold 3	Average
XGB	accuracy	93.673%	93.947%	93.508%	93.709%
	precision	93.687%	93.960%	93.510%	93.719%
	recall	93.673%	93.947%	93.508%	93.709%
	f1_score	93.669%	93.943%	93.505%	93.706%
RF	accuracy	92.905%	93.233%	92.959%	93.033%
	precision	92.881%	93.320%	92.912%	93.038%
	recall	92.832%	93.252%	92.886%	92.990%
	f1_score	92.935%	93.206%	92.843%	92.995%
SVC	accuracy	82.958%	82.370%	81.675%	82.335%
	precision	83.100%	82.648%	81.809%	82.519%
	recall	82.958%	82.370%	81.675%	82.335%
	f1_score	82.898%	82.273%	81.608%	82.260%
KNN	accuracy	91.571%	91.880%	91.533%	91.661%
	precision	91.596%	91.905%	91.572%	91.691%
	recall	91.571%	91.880%	91.533%	91.661%
	f1_score	91.574%	91.883%	91.537%	91.665%
ANN	Accuracy	86.597%	87.582%	87.271%	87.150%
	Precision	84.270%	85.204%	84.781%	84.751%
	Recall	91.527%	92.364%	92.292%	92.061%
	F1 Score	87.749%	88.640%	88.377%	88.255%
LR	accuracy	82.319%	81.730%	81.730%	81.926%
	precision	82.335%	81.724%	81.726%	81.928%
	recall	82.319%	81.730%	81.730%	81.926%
	f1_score	82.289%	81.713%	81.711%	81.904%
NB	accuracy	81.825%	81.401%	81.364%	81.530%
	precision	81.824%	81.424%	81.360%	81.536%
	recall	81.825%	81.401%	81.364%	81.530%
	f1_score	81.809%	81.370%	81.351%	81.510%

A.2.2 Treament Recommendation Model

Table A.31: Cardio Medication Recommendation Model Cross Validation Results

Model	Metric	Fold 1	Fold 2	Fold 3	Mean
XGB	accuracy	61.976%	62.842%	62.642%	62.487%
	precision	67.919%	68.841%	39.660%	58.806%
	recall	23.900%	23.389%	49.362%	32.217%
	f1	33.397%	32.783%	34.646%	33.609%
RF	accuracy	63.479%	64.526%	63.189%	63.732%
	precision	65.795%	64.900%	41.792%	57.496%
	recall	28.197%	29.347%	48.256%	35.267%
	f1	38.808%	38.949%	34.883%	37.547%
LR	accuracy	58.242%	56.193%	59.408%	57.948%
	precision	59.710%	64.768%	39.911%	54.796%
	recall	5.639%	5.389%	26.782%	12.603%
	f1	10.055%	9.711%	27.076%	15.614%
MLP	accuracy	59.882%	62.614%	61.731%	61.409%
	precision	63.496%	66.135%	46.310%	58.647%
	recall	26.957%	26.232%	45.143%	32.777%
	f1	36.338%	36.461%	34.005%	35.601%
ADA	accuracy	57.286%	60.337%	57.904%	58.509%
	precision	65.617%	64.845%	48.337%	59.600%
	recall	22.578%	23.347%	50.919%	32.281%
	f1	31.012%	32.528%	35.489%	33.010%
K-NN	accuracy	53.188%	55.009%	54.077%	54.091%
	precision	56.985%	56.255%	40.804%	51.348%
	recall	32.121%	34.168%	43.830%	36.707%
	f1	40.978%	42.411%	35.622%	39.671%

Table A.32: Respiratory Medication Recommendation Model Cross Validation Results

Model	Metric	Fold 1	Fold 2	Fold 3	Mean
XGB	accuracy	54.744%	55.907%	55.302%	55.318%
	precision	57.311%	49.082%	44.175%	50.189%
	recall	48.640%	31.870%	28.097%	36.202%
	f1	52.197%	36.081%	31.200%	39.826%
RF	accuracy	56.279%	56.791%	55.581%	56.217%
	precision	59.466%	54.663%	50.258%	54.796%
	recall	47.719%	21.106%	30.724%	33.183%
	f1	51.431%	28.591%	32.270%	37.431%
LR	accuracy	44.233%	43.349%	43.767%	43.783%
	precision	67.013%	44.911%	35.416%	49.113%
	recall	2.456%	19.916%	15.469%	12.614%
	f1	4.693%	26.990%	19.888%	17.190%
MLP	accuracy	50.744%	53.209%	50.279%	51.411%
	precision	53.962%	56.944%	46.585%	52.497%
	recall	47.105%	25.124%	28.981%	33.737%
	f1	50.085%	30.746%	31.949%	37.594%
ADA	accuracy	45.628%	45.953%	46.000%	45.860%
	precision	58.049%	51.858%	42.767%	50.891%
	recall	40.263%	23.041%	28.928%	30.744%
	f1	47.540%	23.462%	29.350%	33.450%
K-NN	accuracy	45.535%	43.535%	44.233%	44.434%
	precision	51.391%	49.480%	41.234%	47.368%
	recall	45.702%	28.869%	34.397%	36.323%
	f1	47.844%	35.240%	35.112%	39.398%

Table A.33: Neurological Medication Recommendation Model Cross Validation Results

Model	Metric	Fold 1	Fold 2	Fold 3	Mean
XGB	accuracy	65.164%	65.632%	66.468%	65.754%
	precision	52.926%	52.971%	43.856%	49.918%
	recall	50.611%	50.639%	48.709%	49.986%
	f1	46.891%	46.629%	43.084%	45.535%
RF	accuracy	66.355%	67.078%	67.830%	67.087%
	precision	53.944%	57.381%	41.002%	50.776%
	recall	51.018%	48.374%	49.106%	49.499%
	f1	46.327%	45.376%	43.352%	45.019%
LR	accuracy	31.178%	37.516%	34.894%	34.529%
	precision	49.177%	50.107%	41.814%	47.033%
	recall	44.810%	42.126%	51.172%	46.036%
	f1	42.636%	41.462%	43.827%	42.642%
MLP	accuracy	59.379%	62.697%	62.085%	61.387%
	precision	54.909%	55.064%	48.743%	52.905%
	recall	53.867%	51.041%	26.579%	43.829%
	f1	47.676%	44.813%	32.071%	41.520%
ADA	accuracy	50.915%	51.042%	49.745%	50.567%
	precision	51.333%	50.058%	42.186%	47.859%
	recall	48.372%	46.840%	49.861%	48.357%
	f1	44.483%	43.411%	43.666%	43.853%
K-NN	accuracy	53.041%	51.467%	51.872%	52.127%
	precision	53.096%	47.339%	42.428%	47.621%
	recall	50.102%	54.037%	45.252%	49.797%
	f1	49.627%	46.389%	42.697%	46.238%

Table A.34: Psychiatric Medication Recommendation Model Cross Validation Results

Model	Metric	Fold 1	Fold 2	Fold 3	Mean
XGB	accuracy	93.662%	94.308%	93.545%	93.838%
	precision	54.840%	62.807%	54.364%	57.337%
	recall	44.855%	60.811%	51.873%	52.513%
	f1	49.235%	60.907%	51.308%	53.817%
RF	accuracy	94.131%	94.894%	93.897%	94.308%
	precision	61.855%	64.729%	53.212%	59.932%
	recall	42.744%	56.351%	55.331%	51.476%
	f1	49.332%	59.108%	53.328%	53.923%
LR	accuracy	79.812%	80.282%	81.338%	80.477%
	precision	57.998%	62.394%	52.157%	57.516%
	recall	48.021%	54.865%	46.542%	49.809%
	f1	51.407%	57.000%	47.284%	51.897%
MLP	accuracy	91.021%	90.962%	91.256%	91.080%
	precision	58.785%	62.479%	50.003%	57.089%
	recall	48.021%	53.378%	51.729%	51.043%
	f1	51.541%	56.049%	50.678%	52.756%
ADA	accuracy	83.627%	83.744%	85.563%	84.311%
	precision	58.855%	63.872%	51.259%	57.995%
	recall	42.348%	60.270%	50.288%	50.969%
	f1	49.183%	61.119%	50.031%	53.444%
K-NN	accuracy	87.969%	86.444%	87.793%	87.402%
	precision	60.033%	64.306%	54.364%	59.568%
	recall	52.111%	54.730%	52.161%	53.001%
	f1	52.148%	58.109%	50.694%	53.650%

A.3 Appendix C: Text Features for Individual Complication

Feature Name	Questionnaire	Possible Feature Value
Preillness	Any abdominal preillness?	peritonitis — magen — gi blutung — gallenkolik — nierenkolik — hämoriéd — harn — darmverschluss — appendizitis — pankreatitis — reflux — gastr — hoden — nieren
Abdominal Pain	Pain in any of the abdominal area?	<i>Körperteil:</i> oberbauch — abdominelle — unterbauch — bauch — abdominal — unterleib — flanke — niere — abdom — unterem quardrant — leber — darmausgang — darm — rektal — rektum — after — peg — unterbekleidung — intestinal
Abdomen Abnormality	Any abnormalities relevant to abdomen area?	<i>Signs:</i> erbrechen — erbrochen — erbricht — blutung — vomit <i>Type of Abnormality:</i> kaffeesatz — kaffesatz — blut — schwarz — schwarzer — konzentriert

Table A.35: Text Features For Abdominal Complication

Feature Name	Questionnaire	Possible Feature Value
Preillness	Any psychiatric preillness?	depression, manie, suizidalität, agitation
Intoxication	Any medication overdose?	intox, lsd, drogen, canabis, methadon, gras, tabletten
Alcoholism	Is the patient alcoholic?	betrunken, alkoh, vodka, heroin, wodka, starkalkoholisiert, c2, canabis, besoffen
Mental Abnormalities	Does the patient complain of any abnormality he/she is feeling?	hochachtungsvoll, phantasie, panik, panisch, angst, überdosierung, psych, halluzin
Psychiatric Symptoms	Does the patient show any strong psychiatric symptoms?	keinen-lebenswill, wein, aufgelöst, suizid, depress, stress, verwirrt, angst, wahnhaft, erregt, euphorisch, aggressiv, unruh, entzug, borderlinesyndrom, borderline, erregung

Table A.36: Text Features For Psychiatry Complication

Feature Name	Questionnaire	Possible Feature Value
Preillness	Any neurological preillness?	Amnesi — Apoplex — bek. Krampf — Epilepsie — Epilepticus — Fieberkrampf — ICB — Meningismus — Migräne — Schlaganfall
Neuro Abnormalities	Does the patient have any neurological abnormalities?	Anfall — Anfälle — eingenässt — Epileptikus — Halbseitensymptomatik — hemisymptomatik — Klonisch — krampf — lähmung — mundwinkel — Parese — Tonisch — verwaschen — zungenbiss
Communication Disorder	Does the patient show a communication disorder in terms of speech, vision, movement or consciousness?	Bewußtseinstrübung — Bewusstseinsveränderung — desorient — gefühlstörung — kontrollverlust — orientierungslosigkeit — Psychovegetative Störung — sehstörung — sprachstörung — Unkoordiniert — verlangsamt — verwaschene sprache — verwirtheit — Wesensveränderung — Wortfindung — wortfindungsstörung — wortsalat

Table A.37: Text Features For Neurology Complication

Feature Name	Questionnaire	Possible Feature Value
Preillness	Any metabolic preillness?	hypoglykämie — hyperglykämie — exsikkose — exsikose — exikose — exzikose — urämie — magen — diabetis — diabetes
Metabolic Abnormalities	Does the patient have any metabolic health abnormalities?	exikiert — exikkiert — exsiktiert — gastritis — gastroent — magendarminfekt — wenig urin — stark konzentriertem urin — durchfall — wenig harn — erbrechen — wenig getrunken
Pain/Dryness	Does the patient exhibit typical pain or dryness?	<i>Symptoms:</i> schmerz — trocken — getrock <i>Bodypart:</i> bauch — extremität — krampfartiger

Table A.38: Text Features For Metabolic Complication

Feature Name	Questionnaire	Possible Feature Value
Preillness	Any respiratory preillness?	Tracheostoma — Asthma — Asthmaanfall — asthmaticus — COPD — Aspiration — Bronchitis — Hyperventilation — Laryngeotracheobronchitis — Epiglottitis — Pneumothorax — Hämoptyse — Dyspnoe — Pseudokrupp — Covid — Corona — Pneumonie — Lungenentzündung — tachypnoe — tracheotomi — dispnoe
Respiratory Abnormality	Does the patient have any respiratory health abnormalities?	Kehlkopftumor — Paradoxe Atmung — Ödem — spastik — Fremdkörper — Bronchial — geräusch — husten — atemgeräusch — Kurzatmigkeit — Atemstillstand — Rauchgas — atemnot — hustet — ATEMSTÖRUNG — Atemhilfe — sputum — apnoe — stridor — Reizung — Verschleim — sekret — Inhalation — bradypnoe — giemen — Zyanose — krupp — Dysphagie

Table A.39: Text Features For Respiratory Complication

Bibliography

- [1] Youssef Abadade, Anas Temouden, Hatim Bamoumen, Nabil Benamar, Yousra Chtouki, and Abdelhakim Senhaji Hafid. A comprehensive survey on tinyml. *IEEE Access*, 2023.
- [2] Hossein Abbasi, Maysam Orouskhani, Samaneh Asgari, and Sara Shomal Zadeh. Automatic brain ischemic stroke segmentation with deep learning: A review. *Neurosci. Inform.*, page 100145, 2023.
- [3] Maher GM Abdolrasol, SM Suhail Hussain, Taha Selim Ustun, Mahidur R Sarker, Mahammad A Hannan, Ramizi Mohamed, Jamal Abd Ali, Saad Mekhilef, and Abdalrhman Milad. Artificial neural networks based optimization techniques: A review. *Electronics*, 10(21):2689, 2021.
- [4] Abdullah A Abdullah, Masoud M Hassan, and Yaseen T Mustafa. A review on bayesian deep learning in healthcare: Applications and challenges. *IEEE Access*, 10:36538–36562, 2022.
- [5] Håkon B Abrahamsen. A remotely piloted aircraft system in major incident management: concept and pilot, feasibility study. *BMC emergency medicine*, 15:1–12, 2015.
- [6] Abu Shad Ahammed, Sampada Reddy Donthireddy, and Roman Obermaisser. Detection of respiratory emergency situation of rescue patients with machine learning algorithms. In *IECON 2022–48th Annual Conference of the IEEE Industrial Electronics Society*, pages 1–6. IEEE, 2022.
- [7] Abu Shad Ahammed, Aniebiet Micheal Ezekiel, and Roman Obermaisser. Time-efficient identification procedure for neurological complications of rescue patients in an emergency scenario using hardware-accelerated artificial intelligence models. *Algorithms*, 16(5):258, 2023.
- [8] Abu Shad Ahammed, Micheal Ezekiel, and Roman Obermaisser. A novel analysis of performance and inference time of machine learning models to detect cardiovascular emergency situations of rescue patients. In *2022 International Conference on Artificial Intelligence of Things (ICAIoT)*, pages 1–6. IEEE, 2022.

- [9] Md Manjurul Ahsan, Shahana Akter Luna, and Zahed Siddique. Machine-learning-based disease diagnosis: A comprehensive review. In *Healthcare*, volume 10, page 541. MDPI, 2022.
- [10] KM Al-Aubidy, AM Derbas, and AW Al-Mutairi. Real-time healthcare monitoring system using wireless sensor network. *International Journal of Digital Signals and Smart Systems*, 1(1):26–42, 2017.
- [11] N. E. Andrew, D. A. Cadilhac, V. Sundararajan, A. G. Thrift, P. H. Anderson, N. A. Lannin, and M. F. Kilkenny. Linking australian stroke clinical registry data with australian government medicare and medication dispensing claims data and the potential for bias. *Australian and New Zealand Journal of Public Health*, 45:364–369, 2021.
- [12] Gufran Ahmad Ansari, Salliah Shafi Bhat, Mohd Dilshad Ansari, Sultan Ahmad, Jabeen Nazeer, and AEM Eljialy. Performance evaluation of machine learning techniques (mlt) for heart disease prediction. *Computational and Mathematical Methods in Medicine*, 2023(1):8191261, 2023.
- [13] Nerea Arandia, Jose Ignacio Garate, and Jon Mabe. Embedded sensor systems in medical devices: Requisites and challenges ahead. *Sensors*, 22(24):9917, 2022.
- [14] R. Aringhieri, M. E. Bruni, S. Khodaparasti, and J. T. v. Essen. Emergency medical services and beyond: addressing new challenges through a wide literature review. *Computers & Operations Research*, 78:349–368, 2017.
- [15] Fatemeh Arjmandnia and Ehsan Alimohammadi. The value of machine learning technology and artificial intelligence to enhance patient safety in spine surgery: a review. *Patient Safety in Surgery*, 18(1):11, 2024.
- [16] N. K. M. Astuti, N. W. Utami, and I. G. P. K. Juliharta. Classification of blood donor data using c4.5 and k-nearest neighbor methods (case study: utd pmi bali province). *Jurnal Pilar Nusa Mandiri*, 18:9–16, 2022.
- [17] Audrey Roy Greenfeld. *Cookiecutter Documentation*, 2024. Accessed: 2024-06-06.
- [18] Mariette Awad, Rahul Khanna, Mariette Awad, and Rahul Khanna. Support vector machines for classification. *Efficient learning machines: Theories, concepts, and applications for engineers and system designers*, pages 39–66, 2015.
- [19] D. S. F. Azzahrah and A. Alamsyah. Comparison of probabilistic neural network (pnn) and k-nearest neighbor (k-nn) algorithms for diabetes classification. *Recursive Journal of Informatics*, 1:73–82, 2023.

- [20] Luca Bacco, Fabrizio Russo, Luca Ambrosio, Federico D’Antoni, Luca Vollero, Gianluca Vadalà, Felice Dell’Orletta, Mario Merone, Rocco Papalia, and Vincenzo Denaro. Natural language processing in low back pain and spine diseases: a systematic review. *Frontiers in Surgery*, 9:957085, 2022.
- [21] M. Bakator and D. Radosav. Deep learning and medical diagnosis: a review of literature. *Multimodal Technologies and Interaction*, 2:47, 2018.
- [22] S. Benjamens, P. Dhunoo, and B. Meskó. The state of artificial intelligence-based fda-approved medical devices and algorithms: an online database. *NPJ Digital Medicine*, 3, 2020.
- [23] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(2), 2012.
- [24] Richard A Berk and Richard A Berk. Random forests. *Statistical learning from a regression perspective*, pages 233–295, 2020.
- [25] S. Bird. Nltk. *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computatio*, 1:63–70, 2002.
- [26] Barry Boehm and Victor R Basili. Defect reduction top 10 list. *Computer*, 34(1):135–137, 2001.
- [27] A. Boonmoh. Use of dictionaries and online tools for reading by thai efl learners in a naturalistic setting. *Lexikos*, 31, 2021.
- [28] Fraser JH Brims, Helen E Davies, and YC Gary Lee. Respiratory chest pain: diagnosis and treatment. *Medical Clinics*, 94(2):217–232, 2010.
- [29] L. Capogrosso, F. Cunico, D. S. Cheng, F. Fummi, and M. Cristani. A machine learning-oriented survey on tiny machine learning. *IEEE Access*, 12:23406–23426, 2024.
- [30] I. Chahid, A. K. Elmiad, and M. Badaoui. Data preprocessing for machine learning applications in healthcare: a review. *2023 14th International Conference on Intelligent Systems: Theories and Applications (SITA)*, 2023.
- [31] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.
- [32] Tianqi Chen, Thierry Moreau, Ziheng Jiang, Haichen Shen, Eddie Q Yan, Leyuan Wang, Yuwei Hu, Luis Ceze, Carlos Guestrin, and Arvind Krishnamurthy. Tvm: end-to-end optimization stack for deep learning. *arXiv preprint arXiv:1802.04799*, 11(2018):20, 2018.

- [33] KR1442 Chowdhary and KR Chowdhary. Natural language processing. *Fundamentals of artificial intelligence*, pages 603–649, 2020.
- [34] European Commission. Report from the commission to the european parliament and the council on the effectiveness of the implementation of the single european emergency number '112', 2022. Accessed: YYYY-MM-DD.
- [35] Wikimedia Commons Contributors. Rettungswagen (opel movano) Örk lv nÖ (blitzer-ausführung), 2024. Accessed: 2024-11-18.
- [36] S. Dayal. Ai on edge: a mass accessible tool for decision support systems. *Decision Support Systems (DSS) and Tools [Working Title]*, 2024.
- [37] L. A. Dias, J. C. Ferreira, and M. A. C. Fernandes. Parallel implementation of k-means algorithm on fpga. *IEEE Access*, 8:41071–41084, 2020.
- [38] B. Ehsani-Moghaddam, J. Queenan, J. MacKenzie, and R. Birtwhistle. Mucopolysaccharidosis type ii detection by naïve bayes classifier: an example of patient classification for a rare disease using electronic medical records from the canadian primary care sentinel surveillance network. *Plos One*, 13:e0209018, 2018.
- [39] Khadiga T Elhussiny, Ahmed M Hassan, Ahmed Abu Habssa, and Ali Mokhtar. Prediction of water distribution uniformity of sprinkler irrigation system based on machine learning algorithms. *Scientific Reports*, 13(1):20885, 2023.
- [40] Malcolm Elliott and Alysia Coventry. Critical care: the eight vital signs of patient monitoring. *British Journal of Nursing*, 21(10):621–625, 2012.
- [41] J. Emberson, K. R. Lees, P. D. Lyden, L. Blackwell, G. W. Albers, E. Bluhmki, T. G. Brott, G. Cohen, S. M. Davis, G. A. Donnan, J. C. Grotta, G. Howard, M. Kaste, M. Koga, R. v. Kummer, M. G. Lansberg, R. I. Lindley, G. Murray, J. M. Olivot, M. Parsons, B. C. Tilley, D. Toni, K. Toyoda, N. Wahlgren, J. M. Wardlaw, W. Whiteley, G. J. d. Zoppo, C. Baigent, P. Sandercock, and W. Hacke. Effect of treatment delay, age, and stroke severity on the effects of intravenous thrombolysis with alteplase for acute ischaemic stroke: a meta-analysis of individual patient data from randomised trials. *The Lancet*, 384:1929–1935, 2014.
- [42] Bukuroshe Elira Epoka. Literature review of qualitative data with natural language processing. *Journal of Robotics Spectrum*, 1:056–065, 2023.
- [43] T Sai Eswar and V Karthick. Realtime visual object recognition using support vector machine comparing with k-nearest neighbor algorithm for improving accuracy. *Journal of Pharmaceutical Negative Results*, pages 831–837, 2022.

- [44] Yongxian Fan, Meng Liu, and Guicong Sun. An interpretable machine learning framework for diagnosis and prognosis of covid-19. *Plos one*, 18(9):e0291961, 2023.
- [45] A. Fanariotis, T. Orphanoudakis, and V. Fotopoulos. Reducing the power consumption of edge devices supporting ambient intelligence applications. *Information*, 15:161, 2024.
- [46] Alabhya Farkiya, Prashant Saini, Shubham Sinha, and Sharmishta Desai. Natural language processing using nltk and wordnet. *Int J Comput Sci Inf Technol*, 6(6):5465–5469, 2015.
- [47] Wubalem Fekadu, Awoke Mihiretu, Tom KJ Craig, and Abebaw Fekadu. Multidimensional impact of severe mental illness on family members: systematic review. *BMJ open*, 9(12):e032391, 2019.
- [48] Fernando Ferrando, Yolanda Mira, et al. Effective and safe management of oral anticoagulation therapy in patients who use the internet-accessed telecontrol tool sintromacweb. *Interactive journal of medical research*, 4(2):e3610, 2015.
- [49] Guisseppi A Forgionne and Rajiv Kohli. Hmss: a management support system for concurrent hospital decision making. *Decision Support Systems*, 16(3):209–229, 1996.
- [50] Python Software Foundation. Graphical user interfaces with tk.
- [51] Peter I Frazier. Bayesian optimization. In *Recent advances in optimization and modeling of contemporary problems*, pages 255–278. Informs, 2018.
- [52] Xingyu Fu, Yingyue Chen, Jingru Yan, Yumin Chen, and Feng Xu. Bgrf: A broad granular random forest algorithm. *Journal of Intelligent & Fuzzy Systems*, 44(5):8103–8117, 2023.
- [53] Archit Gajjar, Priyank Kashyap, Aydin Aysu, Paul Franzon, Yongjin Choi, Chris Cheng, Giacomo Pedretti, and Jim Ignowski. Rd-faxid: Ransomware detection with fpga-accelerated xgboost. *ACM Transactions on Reconfigurable Technology and Systems*, 17(4):1–33, 2024.
- [54] Benjamin J Galatzan, Jane M Carrington, and Sheila Gephart. Testing the use of natural language processing software and content analysis to analyze nursing hand-off text data. *CIN: Computers, Informatics, Nursing*, 39(8):411–417, 2021.
- [55] Xing Gao, Qi Lv, and Shike Hou. Progress in the application of portable ultrasound combined with artificial intelligence in pre-hospital emergency and disaster sites. *Diagnostics*, 13(21):3388, 2023.

- [56] F. García-García, D. Lee, F. J. Mendoza-Garcés, and S. García-Gutiérrez. Reliable prediction of difficult airway for tracheal intubation from patient pre-operative photographs by machine learning methods. *Computer Methods and Programs in Biomedicine*, 248:108118, 2024.
- [57] Kritika Gaur and Miheer M Jagtap. Role of artificial intelligence and machine learning in prediction, diagnosis, and prognosis of cancer. *Cureus*, 14(11), 2022.
- [58] D. Giardino, M. Matta, and S. Spanò. An automatic aw-som vhdh ip-core generator. *International Journal on Advanced Science, Engineering and Information Technology*, 9:1136–1141, 2019.
- [59] Ahmad Yousaf Gill, Ayesha Saeed, Saad Rasool, Ali Husnain, and Hafiz Khawar Hussain. Revolutionizing healthcare: How machine learning is transforming patient diagnoses-a comprehensive review of ai’s impact on medical diagnosis. *Journal of World Science*, 2(10):1638–1652, 2023.
- [60] Matthew Goldman, Brandon Lucke-Wold, Jason Katz, Bavly Dawoud, and Abeer Dagra. Respiratory patterns in neurological injury, pathophysiology, ventilation management, and future innovations: A systematic review. *Exploratory research and hypothesis in medicine*, 8(4):338, 2023.
- [61] Lawrence O Gostin, Laura A Levit, Sharyl J Nass, et al. Beyond the hipaa privacy rule: enhancing privacy, improving health through research. 2009.
- [62] Xin Guan, Bo Zhang, Ming Fu, Mengying Li, Xu Yuan, Yaowu Zhu, Jing Peng, Huan Guo, and Yanjun Lu. Clinical and inflammatory features based machine learning model for fatal risk prediction of hospitalized covid-19 patients: results from a retrospective cohort study. *Annals of medicine*, 53(1):257–266, 2021.
- [63] A. Gunawan and I. Fenriana. Design of diabetes prediction application using k-nearest neighbor algorithm. *Bit-Tech*, 6:110–117, 2023.
- [64] Atul Krishna Gupta. *Deep Learning on Microcontrollers Learn How to Develop Embedded AI Applications Using TinyML*. BPB Publications, Birmingham, UK, 2023.
- [65] James A Hanley and Barbara J McNeil. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143(1):29–36, 1982.
- [66] J. Hatwell, M. M. Gaber, and R. M. A. Azad. Ada-whips: explaining adaboost classification with applications in the health sciences. *BMC Medical Informatics and Decision Making*, 20, 2020.
- [67] Sheng He, Leon G Leanse, and Yanfang Feng. Artificial intelligence and machine learning assisted drug delivery for effective treatment of infectious diseases. *Advanced drug delivery reviews*, 178:113922, 2021.

- [68] K. Hsu and H. Tseng. Gptpu: accelerating applications using edge tensor processing units. 2021.
- [69] Safeen Huda, Muntasir Mallick, and Jason H. Anderson. Clock gating architectures for fpga power reduction. In *2009 International Conference on Field Programmable Logic and Applications*, pages 112–118, 2009.
- [70] IEEE Computer Society. Software requirements specifications. <https://www.computer.org/resources/software-requirements-specifications>, 2024. Accessed: 2024-06-05.
- [71] A. N. Iffah'da and A. Desiani. Implementasi algoritma k-nearest neighbor (k-nn) dan single layer perceptron (slp) dalam prediksi penyakit sirosis biliari primer. *Jurnal Ilmiah Informatika*, 7:65–74, 2022.
- [72] WUM Intersites. Advanced data preprocessing for intersites web usage mining.
- [73] Kevin Jamieson and Ameet Talwalkar. Non-stochastic best arm identification and hyperparameter optimization. In *Artificial intelligence and statistics*, pages 240–248. PMLR, 2016.
- [74] Manuela Nayantara Jeyaraj, Senuri Sucharitharathna, Chathurika Senarath, Yasanthy Kanagaraj, and Indraka Udayakumara. Cognitive visual-learning environment for postgresql. *arXiv preprint arXiv:2205.04834*, 2022.
- [75] Fei Jiang, Yong Jiang, Hui Zhi, Yi Dong, Hao Li, Sufeng Ma, Yilong Wang, Qiang Dong, Haipeng Shen, and Yongjun Wang. Artificial intelligence in health-care: past, present and future. *Stroke and vascular neurology*, 2(4), 2017.
- [76] Miao Jin, Qinzhuo Liao, Shirish Patil, Abdulazeez Abdulraheem, Dhafer Al-Shehri, and Guenther Glatz. Hyperparameter tuning of artificial neural networks for well production estimation considering the uncertainty in initialized parameters. *ACS omega*, 7(28):24145–24156, 2022.
- [77] T. Jo, K. Nho, and A. J. Saykin. Deep learning in alzheimer’s disease: diagnostic classification and prognostic prediction using neuroimaging data. *Frontiers in Aging Neuroscience*, 11, 2019.
- [78] Ram D Joshi and Chandra K Dhakal. Predicting type 2 diabetes using logistic regression and machine learning approaches. *International journal of environmental research and public health*, 18(14):7346, 2021.
- [79] Min-Gyue Jung, Seon-A Youn, Jayon Bae, and Yong-Lak Choi. A study on data input and output performance comparison of mongodb and postgresql in the big data environment. In *2015 8th international conference on database theory and application (DTA)*, pages 14–17. IEEE, 2015.

- [80] A. Jurek, Y. Bi, S. Wu, and C. Nugent. A survey of commonly used ensemble-based classification techniques. *The Knowledge Engineering Review*, 29:551–581, 2013.
- [81] Haris Kamal, Victor Lopez, and Sunil A Sheth. Machine learning in acute ischemic stroke neuroimaging. *Frontiers in neurology*, 9:945, 2018.
- [82] Aristeidis Karras, Anastasios Giannaros, Christos Karras, Leonidas Theodorakopoulos, Constantinos S Mammassis, George A Krimpas, and Spyros Sioutas. Tinyml algorithms for big data management in large-scale iot systems. *Future Internet*, 16(2):42, 2024.
- [83] G. Khoerunnisa and W. Astuti. Prediction of retweets based on user, content, and time features using eusboost. *Jurnal Resti (Rekayasa Sistem Dan Teknologi Informasi)*, 6:442–447, 2022.
- [84] Diksha Khurana, Aditya Koli, Kiran Khatter, and Sukhdev Singh. Natural language processing: state of the art, current trends and challenges. *Multimedia tools and applications*, 82(3):3713–3744, 2023.
- [85] Diane M Korngiebel and Sean D Mooney. Considering the possibilities and pitfalls of generative pre-trained transformer 3 (gpt-3) in healthcare delivery. *NPJ Digital Medicine*, 4(1):93, 2021.
- [86] Karl-Patrik Kresoja, Matthias Unterhuber, Rolf Wachter, Holger Thiele, and Philipp Lurz. A cardiologist’s guide to machine learning in cardiovascular disease prognosis prediction. *Basic research in cardiology*, 118(1):10, 2023.
- [87] Ming Kuang, Hang-Tong Hu, Wei Li, Shu-Ling Chen, and Xiao-Zhou Lu. Articles that use artificial intelligence for ultrasound: A reader’s guide. *Frontiers in Oncology*, 11:631813, 2021.
- [88] Deepak Kumar, Priyanka Pramod Pawar, Hari Gonaygunta, Geeta Sandeep Nadella, Karthik Meduri, and Shoumya Singh. Machine learning’s role in personalized medicine & treatment optimization. *World Journal of Advanced Research and Reviews*, 21(2):1675–1686, 2024.
- [89] S. A. Kumar. Enhancing the scope for automated code generation and parallelism by optimizing loops through loop unrolling. *2020 Fourth International Conference on Inventive Systems and Control (ICISC)*, 2020.
- [90] Vankamaddi Sunil Kumar and K Vidhya. Heart plaque detection with improved accuracy using naive bayes and comparing with least squares support vector machine. *Cardiometry*, (25):1595–1599, 2022.

- [91] Yogesh Kumar, Apeksha Koul, Ruchi Singla, and Muhammad Fazal Ijaz. Artificial intelligence in disease diagnosis: a systematic literature review, synthesizing framework and future research agenda. *Journal of ambient intelligence and humanized computing*, 14(7):8459–8486, 2023.
- [92] Agus Kurniawan. Iot projects with arduino nano 33 ble sense. *Berkeley: Apress*, 129, 2021.
- [93] Semi-Supervised Learning. Semi-supervised learning. *CSZ2006. html*, 5:2, 2006.
- [94] Hye-Jin Lee, Dong-Oh Kim, Bub-Joo Kang, and Sang-Woo Ban. Mobile embedded health-care system working on wireless sensor network. In *2011 Third International Conference on Communications and Mobile Computing*, pages 161–164. IEEE, 2011.
- [95] W. Lei. Revolutionizing machine learning: a comprehensive analysis of asic hardware accelerators and their applications. *Applied and Computational Engineering*, 51:152–157, 2024.
- [96] A. Li, J. P. Winebrake, and K. D. Kovacs. Facilitating deep learning through preprocessing of optical coherence tomography images. *BMC Ophthalmology*, 23, 2023.
- [97] Bohan Li. Random search plus: A more effective random search for machine learning hyperparameters optimization. 2020.
- [98] Lisha Li, Kevin Jamieson, Afshin Rostamizadeh, Katya Gonina, Moritz Hardt, Benjamin Recht, and Ameet Talwalkar. Massively parallel hyperparameter tuning. 2018.
- [99] Lisha Li, Kevin G Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Efficient hyperparameter optimization and infinitely many armed bandits. *CoRR*, abs/1603.06560, 16, 2016.
- [100] Lisha Li, Kevin G Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: Bandit-based configuration evaluation for hyperparameter optimization. In *ICLR (Poster)*, page 53, 2017.
- [101] Meng-Pan Li, Wen-Cai Liu, Bo-Lin Sun, Nan-Shan Zhong, Zhi-Li Liu, Shan-Hu Huang, Zhi-Hong Zhang, and Jia-Ming Liu. Prediction of bone metastasis in non-small cell lung cancer based on machine learning. *Frontiers in Oncology*, 12:1054300, 2023.
- [102] Qian Li, Hao Peng, Jianxin Li, Congying Xia, Renyu Yang, Lichao Sun, Philip S Yu, and Lifang He. A survey on text classification: From traditional to deep learning. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 13(2):1–41, 2022.

- [103] Hong Liang, Xiao Sun, Yunlei Sun, and Yuan Gao. Text feature extraction based on deep learning: a review. *EURASIP journal on wireless communications and networking*, 2017:1–12, 2017.
- [104] Yizeng Liang, Qing-Song Xu, Hong-Dong Li, and Dong-Sheng Cao. Support vector machines for classification and regression. *Support Vector Machines and Their Application in Chemistry and Biotechnology*, pages 15–48, 2011.
- [105] H. Lin, H. Chen, L. Weng, J. Shao, and J. Lin. Automatic detection of oral cancer in smartphone-based images using deep learning for early diagnosis. *Journal of Biomedical Optics*, 26, 2021.
- [106] Q. Ling. Machine learning algorithms review. *Applied and Computational Engineering*, 4:91–98, 2023.
- [107] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. v. d. Laak, B. v. Ginneken, and C. I. Sánchez. A survey on deep learning in medical image analysis. *Medical Image Analysis*, 42:60–88, 2017.
- [108] Ioannis E Livieris, Theodore Kotsilieris, Ioannis Dimopoulos, and Panagiotis Pintelas. Decision support software for forecasting patient’s length of stay. *Algorithms*, 11(12):199, 2018.
- [109] Jiafa Lu, Xin Wang, Linghao Chen, Xuedong Sun, Rui Li, Wanjing Zhong, Yajing Fu, Le Yang, Weixiang Liu, and Wei Han. Unmanned aerial vehicle based intelligent triage system in mass-casualty incidents using 5g and artificial intelligence. *World journal of emergency medicine*, 14(4):273, 2023.
- [110] S. Luo. How the red cross society empowers humanitarian assistance: take the example of post-earthquake assistance in turkey. *Lecture Notes in Education Psychology and Public Media*, 15:274–279, 2023.
- [111] D. Lyell, E. Coiera, J. Chen, P. Shah, and F. Magrabi. How machine learning is embedded to support clinician decision making: an analysis of fda-approved medical devices. *BMJ Health & Care Informatics*, 28:e100301, 2021.
- [112] Li Ma and Suohai Fan. Cure-smote algorithm and hybrid algorithm for feature selection and parameter optimization based on random forests. *BMC bioinformatics*, 18:1–18, 2017.
- [113] PM Mah. Analysis of artificial intelligence and natural language processing significance as expert systems support for e-health using pre-train deep learning models. *Acadlore Transactions on AI and Machine Learning*, 1(2):68–80, 2022.
- [114] I. Manoharan, J. L. J, E. C. Sowmiya, S. Harishma, and J. Amose. Medical data analytics and wearable devices. *EAI Endorsed Transactions on Smart Cities*, 6:e2, 2022.

- [115] E. Manor and S. Greenberg. Custom hardware inference accelerator for tensor-flow lite for microcontrollers. *IEEE Access*, 10:73484–73493, 2022.
- [116] Eneida A Mendonça, Janet Haas, Lyudmila Shagina, Elaine Larson, and Carol Friedman. Extracting information on pneumonia in infants using natural language processing of radiology reports. *Journal of biomedical informatics*, 38(4):314–321, 2005.
- [117] Hao Meng, Wasswa Shafik, S Mojtaba Matinkhah, and Zubair Ahmad. A 5g beam selection machine learning algorithm for unmanned aerial vehicle applications. *Wireless Communications and Mobile Computing*, 2020(1):1428968, 2020.
- [118] E. Mezghani, M. D. Silveira, C. Pruski, E. Expósito, and K. Drira. An ontology-driven adaptive system for the patient treatment management. *International Conferences on Software Engineering and Knowledge Engineering*, 2016.
- [119] Riccardo Miotto, Fei Wang, Shuang Wang, Xiaoqian Jiang, and Joel T Dudley. Deep learning for healthcare: review, opportunities and challenges. *Briefings in bioinformatics*, 19(6):1236–1246, 2018.
- [120] Tom M Mitchell. *Machine learning*, volume 1. McGraw-Hill New York, 1997.
- [121] Senthilkumar Mohan, Chandrasegar Thirumalai, and Gautam Srivastava. Effective heart disease prediction using hybrid machine learning techniques. *IEEE access*, 7:81542–81554, 2019.
- [122] Wen Qi Mok, Wenru Wang, and Sok Ying Liaw. Vital signs monitoring to detect patient deterioration: An integrative literature review. *International journal of nursing practice*, 21:91–98, 2015.
- [123] Maryam Mooghali, Austin M Stroud, Dong Whi Yoo, Barbara A Barry, Alyssa A Grimshaw, Joseph S Ross, Xuan Zhu, and Jennifer E Miller. Barriers and facilitators to trustworthy and ethical ai-enabled medical care from patient and healthcare provider perspectives: A literature review. *medRxiv*, pages 2023–10, 2023.
- [124] S. Morag, M. Kieninger, C. Eissnert, S. Auer, S. Dendorfer, D. Popp, J. Hoffmann, and B. Kieninger. Comparison of different techniques for prehospital cervical spine immobilization: biomechanical measurements with a wireless motion capture system. *Plos One*, 18:e0292300, 2023.
- [125] Thierry Moreau, Tianqi Chen, Luis Vega, Jared Roesch, Eddie Yan, Lianmin Zheng, Josh Fromm, Ziheng Jiang, Luis Ceze, Carlos Guestrin, et al. A hardware–software blueprint for flexible deep learning specialization. *IEEE Micro*, 39(5):8–16, 2019.

- [126] Muhasshanah Muhasshanah, Mohammad Tohir, Dewi Andariya Ningsih, Neny Yuli Susanti, Astik Umiyah, and Lia Fitria. Comparison of the performance results of c4. 5 and random forest algorithm in data mining to predict childbirth process. *CommIT (Communication and Information Technology) Journal*, 17(1):51–59, 2023.
- [127] A. Nayak, B. Božić, and L. Longo. Data quality assessment and recommendation of feature selection algorithms: an ontological approach. *Journal of Web Engineering*, 2023.
- [128] Aurélie Névél, Hercules Dalianis, Sumithra Velupillai, Guergana Savova, and Pierre Zweigenbaum. Clinical natural language processing in languages other than english: opportunities and challenges. *Journal of biomedical semantics*, 9:1–13, 2018.
- [129] OECD. *AI in Health: Huge Potential, Huge Risks*. Organisation for Economic Co-operation and Development (OECD), Paris, France, 2024.
- [130] Joseph Chukwudi Okeibunor, Anelisa Jaca, Chinwe Juliana Iwu-Jaja, Ngozi Idemili-Aronu, Housseynou Ba, Zukiswa Pamela Zantsi, Asiphe Mavis Nd-lambe, Edison Mavundza, Derrick Muneene, Charles Shey Wiysonge, et al. The use of artificial intelligence for delivery of essential health services across who regions: a scoping review. *Frontiers in Public Health*, 11:1102185, 2023.
- [131] V. Ong and D. Suhartono. Using k-nearest neighbor in optical character recognition. *ComTech: Computer, Mathematics and Engineering Applications*, 7:53, 2016.
- [132] OpenAI. Ai-generated image: Minimalist icon-style illustration representing health sensor data, 2024. Generated using OpenAI’s DALL-E tool.
- [133] A. Orfanoudaki, A. Giannoutsou, S. W. Hashim, D. Bertsimas, and R. Hagberg. Machine learning models for mitral valve replacement: a comparative analysis with the society of thoracic surgeons risk score. *Journal of Cardiac Surgery*, 37:18–28, 2021.
- [134] Rajkumar Palaniappan, Kenneth Sundaraj, and Sebastian Sundaraj. A comparative study of the svm and k-nn machine learning algorithms for the diagnosis of respiratory pathologies using pulmonary acoustic signals. *BMC bioinformatics*, 15:1–8, 2014.
- [135] Chinmaya K Panda and Habib Md R Karim. Deep machine learning might aid in combating intensive care unit-acquired weakness. *Cureus*, 16(4), 2024.
- [136] Sivaprasad G Pavithraa. Analysis and comparison of prediction of heart disease using novel support vector machine and logistic regression algorithm. *Cardiom-etry*, (25):783–787, 2022.

- [137] Sergio Peñafiel, Analia Hurtado, Marcela Aguirre, Inti Paredes, and Vladimir Pizarro. Implementation and evaluation of an oncological case management system among public and private healthcare providers in chile. *Data and Metadata*, 2:122–122, 2023.
- [138] M. Pramularsih and M. Riasetiawan. Solving the imbalanced and limited data labeled for automated essay scoring using cost sensitive xgboost and pseudo-labeling. *International Journal of Advanced Computer Science and Applications*, 13, 2022.
- [139] Philipp Probst, Marvin N Wright, and Anne-Laure Boulesteix. Hyperparameters and tuning strategies for random forest. *Wiley Interdisciplinary Reviews: data mining and knowledge discovery*, 9(3):e1301, 2019.
- [140] Winnie WY Pun. *A design method for object-oriented programming*. PhD thesis, UCL (University College London), 1990.
- [141] M. Rowland and A. O. Adefuye. Factors that influence human error and patient safety in the pre-hospital emergency care setting: perspectives of emergency care practitioners in a middle-income country. 2021.
- [142] Umer Saeed, Syed Yaseen Shah, Jawad Ahmad, Muhammad Ali Imran, Qammer H Abbasi, and Syed Aziz Shah. Machine learning empowered covid-19 patient monitoring using non-contact sensing: An extensive review. *Journal of pharmaceutical analysis*, 12(2):193–204, 2022.
- [143] Swapnil Sayan Saha, Sandeep Singh Sandha, and Mani Srivastava. Machine learning for microcontroller-class hardware: A review. *IEEE Sensors Journal*, 22(22):21362–21390, 2022.
- [144] Amin Sahebi, Marco Barbone, Marco Procaccini, Wayne Luk, Georgi Gaydadjiev, and Roberto Giorgi. Distributed large-scale graph processing on fpgas. *Journal of big Data*, 10(1):95, 2023.
- [145] Anwar Sanusi, Chrystia Aji Putra, and Fawwaz Ali Akbar. Implementation of adaboost algorithm on c50 for improving the performance of liver disease classification. *JEECS (Journal of Electrical Engineering and Computer Sciences)*, 8:93–102, 2023.
- [146] P. H. Saputro, W. F. Zalmi, and R. Syahputra. Performance testing of knn and logistic regression algorithms in classifying heart disease susceptibility. *International Journal of Computer and Information System (IJCIS)*, 4:140–144, 2023.
- [147] Patrick Schober and Thomas R Vetter. Logistic regression in medical research. *Anesthesia & Analgesia*, 132(2):365–366, 2021.

- [148] Patrick Schratz, Jannes Muenchow, Eugenia Iturritxa, Jakob Richter, and Alexander Brenning. Hyperparameter tuning and performance assessment of statistical and machine-learning algorithms using spatial data. *Ecological Modelling*, 406:109–120, 2019.
- [149] John Semmlow. *Signals and Systems for Bioengineers: A MATLAB-Based Introduction*, chapter Biomedical Engineering, pages 261–316. Academic Press, second edition edition, 2012.
- [150] Kennedy Senagi and Nicolas Jouandeau. Parallel construction of random forest on gpu. *The Journal of Supercomputing*, 78(8):10480–10500, 2022.
- [151] Guang Shi, Zheng Chen, and Renyuan Zhang. Efficient support vector machine toward medical data processing. In *Proceedings of Seventh International Congress on Information and Communication Technology: ICICT 2022, London, Volume 2*, pages 751–760. Springer, 2022.
- [152] Ali H Shoeb and John V Guttag. Application of machine learning to epileptic seizure detection. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 975–982, 2010.
- [153] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25, 2012.
- [154] Sedat Sonko, Ayodeji Matthew Monebi, Emmanuel Augustine Etukudoh, Femi Osasona, Akoh Atadoga, and Cosmas Dominic Daudu. Reviewing the impact of embedded systems in medical devices in the usa. *International Medical Science Research Journal*, 4(2):158–169, 2024.
- [155] Tom Springer, Erik Linstead, Peiyi Zhao, and Chelsea Parlett-Pelleriti. Towards qos-based embedded machine learning. *Electronics*, 11(19):3204, 2022.
- [156] P. H. Stefanovski, R. Vladimir Radkov, T. Lyubomir Ilkov, T. Pencho Tonchev, T. Yoana Mladenova, K. Vihar Manchev, and R. Radko Nikolov. Analysis of mortality in the emergency department at a university hospital in pleven. *The Journal of international medical research*, 45(5):1553–1561, 2017.
- [157] L. Stornaiuolo, F. Carloni, R. Pressiani, G. Natale, M. D. Santambrogio, and D. Sciuto. Enabling transparent hardware acceleration on zynq soc for scientific computing. *ACM SIGBED Review*, 17:30–35, 2020.
- [158] Muralidhar Subramanian, Vishu Krishnamurthy, and Redwood Shores. Performance challenges in object-relational dbmss. *IEEE Data Eng. Bull.*, 22(2):27–31, 1999.

- [159] Sona Taheri and Musa Mammadov. Learning the naive bayes classifier with optimization models. *International Journal of Applied Mathematics and Computer Science*, 23(4):787–795, 2013.
- [160] PyTorch Team. Real-time machine learning on raspberry pi, 2024. Accessed: 2024-12-16.
- [161] Afandi Nur Aziz Thohari and Rima Dias Ramadhani. Performance comparison supervised machine learning models to predict customer transaction through social media ads. *Journal of Computer Networks, Architecture and High Performance Computing*, 4(2):116–126, 2022.
- [162] M. M. Thu, M. Méndez Real, M. Pelcat, and P. Besnier. You only get one-shot: eavesdropping input images to neural network by spying soc-fpga internal bus. *Proceedings of the 18th International Conference on Availability, Reliability and Security*, 2023.
- [163] R. Todeschini, D. Ballabio, M. Cassotti, and V. Consonni. N3 and bnn: two new similarity based classification methods in comparison with other classifiers. *Journal of Chemical Information and Modeling*, 55:2365–2374, 2015.
- [164] Jack V Tu. Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes. *Journal of clinical epidemiology*, 49(11):1225–1231, 1996.
- [165] Siti Zubaidah Mohd Tumari and Rubita Sudirman. Logistic regression of working memory impairments in children based on single-trial erp features. In *2014 2nd International Conference on Technology, Informatics, Management, Engineering & Environment*, pages 264–269. IEEE, 2014.
- [166] S. Uddin, I. Haque, H. Lu, M. A. Moni, and E. Gide. Comparative performance analysis of k-nearest neighbour (knn) algorithm and its different variants for disease prediction. *Scientific Reports*, 12, 2022.
- [167] Muhammad Usama, Junaid Qadir, Aunn Raza, Hunain Arif, Kok-Lim Alvin Yau, Yehia Elkhatib, Amir Hussain, and Ala Al-Fuqaha. Unsupervised machine learning for networking: Techniques, applications and research challenges. *IEEE access*, 7:65579–65615, 2019.
- [168] Aly Al-Amyn Valliani, Daniel Ranti, and Eric Karl Oermann. Deep learning and neurology: a systematic review. *Neurology and therapy*, 8(2):351–365, 2019.
- [169] B Venkatesh and J Anuradha. A review of feature selection and its methods. *Cybernetics and information technologies*, 19(1):3–26, 2019.
- [170] Vilros. Raspberry pi 5: Elevating single board computing to new heights, 2024. Accessed: 2024-06-16.

- [171] A. Wahid, F. Azim, and F. Firdausi. Application of data mining to classify receiving social assistance using the naïve bayes method. *INSIDE - Jurnal Sistem Informatika Cerdas*, 1:62–66, 2023.
- [172] Abdul Wahid, Fauzan Azim, and Faiz Firdausi. Application of data mining to classify receiving social assistance using the naïve bayes method. *INSIDE-Jurnal Sistem Informatika Cerdas*, 1(2):62–66, 2023.
- [173] J. Wang and X. Li. Pynq framework based object recognition implementation using convolution neural network (cnn). *Review of Computer Engineering Studies*, 9:136–140, 2022.
- [174] Liyang Wang, Xiaoya Wang, Angxuan Chen, Xian Jin, and Huilian Che. Prediction of type 2 diabetes risk and its effect evaluation based on the xgboost model. In *Healthcare*, volume 8, page 247. MDPI, 2020.
- [175] X. Wang and L. Hsu. Treatment-seeking delays in patients with acute myocardial infarction and use of the emergency medical service. *Journal of International Medical Research*, 41:231–238, 2013.
- [176] Xinzhi Wang, Jiahao Li, Ze Zheng, Yudong Chang, and Min Zhu. Entity and relation extraction with rule-guided dictionary as domain knowledge. *Frontiers of Engineering Management*, 9(4):610–622, 2022.
- [177] Yanbu Wang, Linqing Liu, and Chao Wang. Trends in using deep learning algorithms in biomedical prediction systems. *Frontiers in Neuroscience*, 17:1256351, 2023.
- [178] J Christopher Westland. The cost of errors in software development: evidence from industry. *Journal of Systems and Software*, 62(1):1–9, 2002.
- [179] Felix Wex, Guido Schryen, Stefan Feuerriegel, and Dirk Neumann. Emergency response in natural disaster management: Allocation and scheduling of rescue units. *European Journal of Operational Research*, 235(3):697–708, 2014.
- [180] Z. Yan, Z. Sun, Y. Zheng, J. Bu, and W. Wang. A high-performance key-value query solution based on hash dictionary and trie tree. *Proceedings of the 2015 International Conference on Computer Science and Intelligent Communication*, 2015.
- [181] De-Ming Yang, Tai-Jay Chang, Kai-Feng Hung, Mong-Lien Wang, Yen-Fu Cheng, Su-Hua Chiang, Mei-Fang Chen, Yi-Ting Liao, Wei-Qun Lai, and Kung-Hao Liang. Smart healthcare: A prospective future medical approach for covid-19. *Journal of the Chinese Medical Association*, 86(2):138–146, 2023.
- [182] N. Ye. Naïve bayes classifier. *Data Mining*, pages 31–36, 2013.

- [183] Yalan Zhang, Jingwen Xie, Enlong Fu, Wan Cai, and Wentan Xu. Artificial intelligence in cardiology: a bibliometric study. *American Journal of Translational Research*, 16(3):1029, 2024.
- [184] J. Zhao, Y. Chang, D. Li, C. Xia, H. Cui, K. Zhang, and X. Feng. On retargeting the ai programming framework to new hardware. *Lecture Notes in Computer Science*, pages 39–51, 2018.
- [185] T. Zhao. Fpga-based machine learning: platforms, applications, design considerations, challenges, and future directions. *Highlights in Science, Engineering and Technology*, 62:96–101, 2023.
- [186] Xinyu Zhao, Bo Liu, Qunwei Lin, Jiabin Huang, and Liqiang Yu. Effective combination of 3d-densenet’s artificial intelligence technology and gallbladder cancer diagnosis model. *Frontiers in Computing and Intelligent Systems*, 6(3):81–84, 2023.