

A Major Project Report on

**A REVOLUTIONIZING EMERGENCY HEALTHCARE IN
DEVELOPING INDIA: AN AI-INTEGRATED AMBULANCE
SYSTEM FOR TIMELY INTERVENTION IN CRITICAL
CONDITIONS**

Submitted to

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, HYDERABAD

In Partial fulfilment of the requirement for the award of degree of the

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

By

P. Sai Ram - 20RA1A0572

V. Sai Bharath - 20RA1A0566

P. Sai Rohith - 20RA1A0568

Under the guidance of

Dr. S. Kavitha

Professor & Head of the Department

Department of Computer Science and Engineering



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

KOMMURI PRATAP REDDY INSTITUTE OF TECHNOLOGY

(Affiliated to JNTUH, Ghanpur(V), Ghatkesar(M), Medchal(D)-500088)

2020- 2024

KOMMURI PRATAP REDDY INSTITUTE OF TECHNOLOGY
(Affiliated to JNTUH, Ghanpur(V), Ghatkesar(M), Medchal(D)-500088)



CERTIFICATE

This is to certify that the project work entitled **“A REVOLUTIONIZING EMERGENCY HEALTHCARE IN DEVELOPING INDIA: AN AI-INTEGRATED AMBULANCE SYSTEM FOR TIMELY INTERVENTION IN CRITICAL CONDITIONS”** is submitted by Mr. P. Sai Ram, Mr. V. Sai Bharath, Mr. P. Sai Rohith bonafied students of **Kommuri Pratap Reddy Institution of Technology** in partial fulfillment of the requirement for the reward of the Degree of Bachelor of Technology in **Computer Science and Engineering** of the **Jawaharlal Nehru Technological University Hyderabad**, during the year 2023-2024.

Internal Guide

Dr. S .Kavitha

HOD

Dr. S .Kavitha

Project Coordinator

Mrs. G. Sujatha

External Examiner

DECLARATION

We hereby declare that this project work entitled “**A REVOLUTIONIZING EMERGENCY HEALTHCARE IN DEVELOPING INDIA: AN AI-INTEGRATED AMBULANCE SYSTEM FOR TIMELY INTERVENTION IN CRITICAL CONDITIONS**” in partial fulfillment of requirements for the award of degree of **Computer Science and Engineering** is a bonafide work carried out by us during the academic year 2023- 2024.

We further declare that this project is a result of our effort and has not been submitted for the award of any degree by us to any institute.

By

P. SAI RAM (20RA1A0572)

V. SAI BHARATH (20RA1A0566)

P.SAI ROHITH (20RA1A0568)

ACKNOWLEDGEMENT

It gives us immense pleasure to acknowledge with gratitude, the help and support extended throughout the project report from the following:

We will be very much grateful to almighty our **Parents** who have made us capable of carrying out our job.

We express our profound gratitude to **Dr. P. Srinivas Rao, Principal of Kommuri Pratap Reddy Institute of Technology**, who has encouraged in completing our project report successfully.

We are grateful to **Dr. S. Kavitha** who is our **Head of the Department, CSE** for her amiable ingenious and adept suggestions and pioneering guidance during the project report.

We express our gratitude and thanks to the Project Coordinator **Mrs. G. Sujatha, Assistant Professor**, of our department for her contribution for making it success with in the given time duration.

We express our deep sense of gratitude and thanks to **Internal Guide, Dr. S. Kavitha, Professor and Head of the Department, CSE** for her guidance during the project report.

We are also very thankful to our **Management, Staff Members** and all **Our Friends** for their valuable suggestions and timely guidance without which we would not have been completed it.

By

P. SAI RAM (20RA1A0572)

V. SAI BHARATH (20RA1A0566)

P. SAI ROHITH (20RA1A0568)

Vision of the Institute

To emerge as a premier institute for high quality professional graduates who can contribute to economic and social developments of the Nation.

Mission of the Institute

Mission	Statement
IM₁	To have holistic approach in curriculum and pedagogy through industry interface to meet the needs of Global Competency.
IM₂	To develop students with knowledge, attitude, employability skills, entrepreneurship, research potential and professionally Ethical citizens.
IM₃	To contribute to advancement of Engineering & Technology that would help to satisfy the societal needs.
IM₄	To preserve, promote cultural heritage, humanistic values and Spiritual values thus helping in peace and harmony in the society.

Vision of the Department

To Provide Quality Education in Computer Science for the innovative professionals to work for the development of the nation.

Mission of the Department

Mission	Statement
DM1	Laying the path for rich skills in Computer Science through the basic knowledge of mathematics and fundamentals of engineering
DM2	Provide latest tools and technology to the students as a part of learning Infrastructure
DM3	Training the students towards employability and entrepreneurship to meet the societal needs.
DM4	Grooming the students with professional and social ethics

Program Educational Objectives (PEOs)

PEO'S	Statement
PEO1	The graduates of Computer Science and Engineering will have successful career in technology.
PEO2	The graduates of the program will have solid technical and professional foundation to continue higher studies.
PEO3	The graduate of the program will have skills to develop products, offer services and innovation.
PEO4	The graduates of the program will have fundamental awareness of industry process, tools and technologies.

Program Outcomes

PO1	Engineering Knowledge: Apply the knowledge of mathematics, science, Engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
PO2	Problem Analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO3	Design/development of Solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
PO4	Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
PO5	Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
PO6	The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
PO7	Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental context, and demonstrate the knowledge of, and need for sustainable development.
PO8	Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9	Individual and team network: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
PO10	Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, being able to comprehend and write effective reports and design documentation, make Effective presentations, and give and receive clear instructions.
PO11	Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environment.
PO12	Life-Long learning: Recognize the need for, and have the preparation and able to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES

PSO1	Foundation of mathematical concepts: To use mathematical methodologies to crack problem using suitable mathematical analysis, data structure and suitable algorithm.
PSO2	Foundation of Computer Science: The ability to interpret the fundamental concepts and methodology of computer systems. Students can understand the functionality of hardware and software aspects of computer systems.
PSO3	Foundation of Software development: The ability to grasp the software development lifecycle and methodologies of software systems. Possess competent skills and knowledge of software design process.

ABSTRACT

The field of emergency medical services (EMS) has undergone significant advancements over the years, with a focus on improving response times, patient care, and overall outcomes. The integration of artificial intelligence (AI) and human interaction technologies into ambulances represents a transformative approach to enhance emergency medical care. The background encompasses the evolution of EMS, the rise of AI in healthcare, and the potential for synergies between technology and human interactions in emergency situations. Historically, ambulances have primarily been vehicles equipped with basic life support equipment and staffed by paramedics and emergency medical technicians to provide initial care during transportation to a medical facility. Communication with hospitals and the processing of patient information be manual and time-consuming. The traditional system is lack of real-time data analysis capabilities and decision support tools that AI can offer in emergency situations. The problem at hand is optimizing emergency medical responses and care through the integration of AI and human interaction technologies within ambulances. This involves addressing challenges such as quick and accurate diagnosis, communication between emergency responders and healthcare facilities, and the provision of real-time medical information to enhance decision-making. The goal is to create a seamless, intelligent, and responsive system that improves patient outcomes during critical moments. The need for the intelligence ambulance arises from the recognition that leveraging AI and human interaction technologies can significantly enhance the efficiency and effectiveness of emergency medical services. In critical situations, quick and accurate decision-making is crucial, and the integration of intelligent systems can provide valuable support to healthcare professionals. This approach can also improve communication, data sharing, and coordination between ambulances, hospitals, and other healthcare entities.

CONTENTS

ABSTRACT.....	x
LIST OF FIGURES	xiii
CHAPTER 1.....	01
INTRODUCTION.....	01
1.1 History.....	01
1.2 Problem Statement.....	02
1.3 Research Motivation.....	03
1.4 Applications.....	04
CHAPTER 2.....	05
LITERATURE SURVEY	05
CHAPTER 3.....	07
EXISTING SYSTEM.....	07
3.1 Decision Tree	07
3.1.1 DTC algorithm	07
3.1.2 Important Features of DTC	07
3.1.3 Assumptions for DTC.....	08
3.2 Drawbacks of Decision Tree Classifier.....	08
CHAPTER 4	10
PROPOSED SYSTEM.....	10
4.1 Overview.....	10
4.2 Random Forest Algorithm.....	11
4.2.1 Important Features of Random Forest.....	12
4.2.2 Assumptions for Random Forest.....	13
4.2.3 Types of Ensembles.....	13

4.3 Advantages.....	14
CHAPTER 5.....	16
UML DIAGRAM.....	16
CHAPTER 6.....	22
SOFTWARE ENVIRONMENT.....	22
CHAPTER 7.....	34
SYSTEM REQUIREMENTS SPECIFICATIONS.....	34
CHAPTER 8.....	35
FUNCTIONAL REQUIREMENTS.....	35
CHAPTER 9.....	40
SOURCE CODE.....	40
CHAPTER 10.....	54
10.1 Implementation Description.....	54
10.2 Dataset Description.....	55
10.3 Results Description.....	56
CHAPTER 11.....	63
CONCLUSIONS & FUTURE SCOPE.....	63
REFERENCES.....	65

LIST OF FIGURES

1. Architectural Block Diagram of Proposed System.....	11
2. Random Forest Algorithm.....	12
3. Main GUI application of proposed intelligence ambulance server side.....	56
4. Selecting the dataset in the GUI application.....	57
5. Presents the pre-processed data from the dataset.....	57
6. Displays the dataset and count plot of target columns.....	58
7. Displays the performance evaluation and confusion matrix of the decision tree model.....	58
8. Displays the performance evaluation and confusion matrix of the random forest model.....	59
9. Displays the performance evaluation and confusion matrix of the KNN model.....	59
10. Displays the comparison of performance metrics in Decision Tree, RFC and KNN models	60
11. Displays the start of cloud server in the GUI console.	60
12. Main GUI application of ambulance side.....	61
13. Selecting the test dataset and reporting to hospital server.....	62
14. Represents the predication of test data by the server model.....	62

CHAPTER 1

INTRODUCTION

1.1 History

The evolution of Emergency Medical Services (EMS) has marked significant progress, driven by a continuous effort to enhance response times, patient care, and overall outcomes. In this context, the integration of Artificial Intelligence (AI) and human interaction technologies into ambulances stands out as a transformative approach to elevate the standards of emergency medical care.

Traditionally, ambulances have functioned as vehicles equipped with basic life support equipment, staffed by paramedics and emergency medical technicians. Their primary role has been to provide initial care during transportation to a medical facility. Communication with hospitals and the processing of patient information have been manual and time-consuming, lacking real-time data analysis capabilities and decision support tools that AI can offer in emergency situations.

The challenge at hand involves optimizing emergency medical responses and care through the seamless integration of AI and human interaction technologies within ambulances. This optimization targets critical areas such as quick and accurate diagnosis, efficient communication between emergency responders and healthcare facilities, and the provision of real-time medical information to enhance decision-making during emergencies.

The ultimate goal is to create a seamless, intelligent, and responsive system that significantly improves patient outcomes during critical moments. The need for an intelligent ambulance arises from the understanding that leveraging AI and human interaction technologies can markedly enhance the efficiency and effectiveness of emergency medical services. In critical situations, where quick and accurate decision-making is crucial, the integration of intelligent systems can provide invaluable support to healthcare professionals.

This approach also addresses the improvement of communication, data sharing, and coordination between ambulances, hospitals, and other healthcare entities. The historical context underscores the limitations of traditional EMS systems and highlights the potential for innovation through the integration of AI, ushering in a new era of intelligent emergency medical services.

1.2 Problem statement

The progression of emergency medical services (EMS) has witnessed substantial advancements, with a dedicated emphasis on refining response times, elevating patient care, and enhancing overall outcomes. A pivotal transformative approach in this trajectory involves the integration of artificial intelligence (AI) and human interaction technologies into ambulances, signifying a paradigm shift in the landscape of emergency medical care.

Delving into the historical context, ambulances traditionally functioned as vehicles equipped with fundamental life support apparatus, manned by paramedics and emergency medical technicians. Their primary role centered around providing initial care during transportation to a medical facility. However, communication processes with hospitals and the manual processing of patient information were notably time-consuming. The conventional system also lacked real-time data analysis capabilities and decision support tools, aspects where AI demonstrates significant potential in emergency scenarios.

The prevailing challenge revolves around optimizing emergency medical responses and care by seamlessly integrating AI and human interaction technologies within ambulances. This optimization necessitates addressing complex issues, including the imperative for swift and accurate diagnosis, fostering effective communication between emergency responders and healthcare facilities, and ensuring the real-time provision of medical information to enhance decision-making processes.

The overarching goal is to craft a seamless, intelligent, and responsive system that markedly improves patient outcomes during critical moments. The impetus for an intelligent ambulance arises from the acknowledgment that leveraging AI and human interaction technologies has the potential to substantially enhance the efficiency and effectiveness of emergency medical services. In critical situations where quick and accurate decision-making is paramount, the integration of intelligent systems becomes a valuable asset to healthcare professionals.

This approach extends its benefits to enhancing communication, facilitating data sharing, and streamlining coordination between ambulances, hospitals, and other entities within the healthcare ecosystem. In essence, the problem statement underscores the imperative to address existing challenges in emergency medical services through the innovative integration of AI and human interaction technologies in ambulances, with the ultimate aim of redefining and elevating the standards of emergency medical care.

1.3 Research Motivation

The motivation behind researching and implementing AI and human interaction technologies in the realm of emergency medical services (EMS) stems from the continual evolution of this field, aiming to enhance response times, elevate patient care standards, and optimize overall outcomes. The backdrop of this research encompasses the dynamic progress of EMS, the pervasive influence of artificial intelligence (AI) in healthcare, and the promising potential for synergies between technological innovations and human interactions, especially in critical emergency situations.

In the historical context, ambulances have traditionally functioned as vehicles equipped with essential life support apparatus and manned by paramedics and emergency medical technicians. Their primary role involves providing initial care during the transportation of patients to medical facilities. However, the historical EMS system faced challenges, including manual and time-consuming communication processes with hospitals and the processing of patient information. Notably, the traditional system lacked the real-time data analysis capabilities and decision support tools that AI could potentially offer, particularly in urgent emergency situations.

The central problem addressed by this research is the optimization of emergency medical responses and care through the strategic integration of AI and human interaction technologies within ambulances. Key challenges include the need for swift and accurate diagnosis, effective communication channels between emergency responders and healthcare facilities, and the provision of real-time medical information to enhance the decision-making processes of healthcare professionals.

The overarching goal is to establish a seamless, intelligent, and responsive system that significantly improves patient outcomes during critical moments. The recognition of the imperative to leverage AI and human interaction technologies in ambulances arises from the understanding that such integration has the potential to substantially enhance the efficiency and effectiveness of emergency medical services. In critical situations where rapid and accurate decision-making is paramount, the integration of intelligent systems becomes a valuable asset to healthcare professionals.

This innovative approach extends its benefits beyond improved decision-making. It also encompasses enhanced communication, streamlined data sharing, and improved coordination between ambulances, hospitals, and other entities within the broader healthcare ecosystem. The research motivation, therefore, lies in the aspiration to advance the capabilities of EMS by

harnessing the synergies between AI and human interaction technologies, ultimately contributing to the ongoing transformation of emergency medical care.

1.4 Applications

The integration of artificial intelligence (AI) and human interaction technologies into ambulances has the potential to revolutionize emergency medical services (EMS) and significantly enhance patient care. In the historical context, ambulances have traditionally been equipped with basic life support tools and staffed by paramedics and emergency medical technicians. However, the communication process with hospitals and the manual handling of patient information have been time-consuming and lacked real-time data analysis capabilities and decision support tools.

The problem at hand involves optimizing emergency medical responses and care by seamlessly integrating AI and human interaction technologies within ambulances. This integration aims to address challenges such as achieving quick and accurate diagnoses, improving communication between emergency responders and healthcare facilities, and providing real-time medical information to enhance decision-making.

The ultimate goal is to create an intelligent ambulance system that responds to emergencies with efficiency and precision, ultimately improving patient outcomes during critical moments. Recognizing the potential of leveraging AI and human interaction technologies is crucial, as these advancements can significantly enhance the efficiency and effectiveness of emergency medical services. In critical situations, where quick and accurate decision-making is paramount, the integration of intelligent systems can provide valuable support to healthcare professionals.

The applications of this approach extend beyond immediate patient care. The intelligent ambulance system can also improve communication, streamline data sharing, and enhance coordination between ambulances, hospitals, and other healthcare entities. This transformative approach aligns with the broader advancements in EMS, the growing influence of AI in healthcare, and the increasing recognition of the importance of synergies between technology and human interactions in emergency situations. The integration of AI into ambulances represents a forward-looking strategy to usher in a new era of responsive and intelligent emergency medical care.

CHAPTER 2

LITERATURE SURVEY

In 2020 Akca et al. [1] put forward a paper which mainly emphasizes on “Intelligent Ambulance Management System in Smart Cities.” technique to manage ambulance and emergency services. This research is efficient to cover all the things needed to Design & Development of Intelligent Ambulance Concept – AI and Human Interface Technology Section A-Research paper 179 Eur. Chem. Bull. 2023,12(Special Issue 9), 177-188 develop a smart ambulance management framework but lacks to explain how the system can work in real time with a combination of mobile computing, cloud computing and standalone application together. In 2021 Ganesh et al. [2] presented a study on “health machine to handle covid-19 related health emergencies” technique to manage ambulance and emergency services. This research effectively covers all the requirements for developing a smart ambulance management framework, but it falls short on describing how the system may function in real time by combining mobile, cloud, and standalone applications altogether. Gargi Beri, Ashwin Channawar, Pankaj Ganjare, Amruta Gate, Prof. Vijay Gaikwad published, “Intelligent Ambulance with Traffic Control” [3]. This study includes a traffic control system as well as a health monitoring system. In health monitoring system, the patient's vital health parameters such as ECG, Heart Rate and Body Temperature are monitored. These parameters are sent to a PC in ambulance via serial communication and this data will be sent to the hospital server. Ms. Aisha Meethian, Althaf B K, Athinan Saeed, Ligin Abraham, Mohammed Samran Hashim Proposed a study on “IoT Based Traffic Control System with Patient Health Monitoring For Ambulance” in August 2022 [4]. The proposed system optimizes the route by minimizing the transport duration to the hospital by using GPS sensor networks. The health parameter of the patient is monitored using different sensors like Heart Rate Sensor, Breath Sensor and Temperature Sensor . These parameters collected from the patient are transmitted to the hospital’s database using IOT. In [5], To provide an intelligent smart health system which sense the body condition and send the data to the collaborated hospital’s website. A device in which the heart beat sensor will sense the heart beat and temperature sensor will sense the body temperature. After sensing, sensors will send respective data to the microcontroller. After that microcontroller will sent it to raspberry-pi which will connect with the internet or IOT cloud. To reach the destination on time the driver will use google map along with accidentavoidance features to save lives. In 2022 Timothy Malche et.al. [6] proposed a system m consists of a sensor node to track patients’ vitals during different activities which patients perform. The proposed sensor node collects patients’ data

using the sensors attached to the nRF5340 Development Kit (DK). The connected sensors are accelerometer, microphone, pulse oximeter, heart rate sensor, and temperature sensor. The accelerometer enables monitoring different patient physical activities, including walking, sleeping, exercising, and running. By analyzing the vitals during different activities, the doctor can prescribe treatment or give suggestions to patients. In [7], A study presents a new method for pulse detection during Out of Hospital Cardiac Arrest using the electrocardiogram (ECG) and Thoracic impedance (TI) signals. The approach uses an adaptive filter to extract the circulatory-related component from the TI referred to as impedance circulation component (ICC) and a support vector machine (SVM) classifier based on features extracted from the ECG and ICC to discriminate pulseless electrical activity (PEA) and PR interval [1]. In [8], an Enhanced deep convolutional neural network (EDCNN) has been proposed for the early detection of heart disease and diagnosis. This research has been developed on EDCNN approach to detect heart disorders in patients and to improve diagnostic precision using deep learning-based prediction models. The prediction of heart disease by processing patient data to calculate the chance of heart ailment has been mathematically computed Design & Development of Intelligent Ambulance Concept – AI and Human Interface Technology Section A-Research paper 180 Eur. Chem. Bull. 2023,12(Special Issue 9), 177-188 with distributive functions. Heart activity has been analyzed during exercise, resting, and working [2]. In [9], The proposed method use Decision Tree algorithm for feature selection method, PCA for dimension reduction and ANN for the classification. The principal component analysis (PCA) is a statistical technique that uses mathematical principles to convert a set of observations (or samples) of possibly correlated variables into a new set of observations of linearly uncorrelated variables [3]. The method used in this work involves Data Collection, Pre-processing, Feature extraction, Dimension Reduction, Classification and Result Analysis

CHAPTER 3

EXISTING SYSTEM

3.1 Decision Tree

DTC is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model. As the name suggests, "DTC is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the DTC takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output. The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

3.1.1 DTC algorithm

Step 1: In DTC n number of random records are taken from the data set having k number of records.

Step 2: Individual decision trees are constructed for each sample.

Step 3: Each decision tree will generate an output.

Step 4: Final output is considered based on Majority Voting or Averaging for Classification and regression respectively.

3.1.2 Important Features of DTC

- **Diversity**- Not all attributes/variables/features are considered while making an individual tree, each tree is different.
- **Immune to the curse of dimensionality**- Since each tree does not consider all the features, the feature space is reduced.
- **Parallelization**-Each tree is created independently out of different data and attributes. This means that we can make full use of the CPU to build DTCs.

- **Train-Test split**- In a DTC we don't have to segregate the data for train and test as there will always be 30% of the data which is not seen by the decision tree.
- **Stability**- Stability arises because the result is based on majority voting/ averaging.

3.1.3 Assumptions for DTC

Since the DTC combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all the trees predict the correct output. Therefore, below are two assumptions for a better DTC classifier:

- There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.
- The predictions from each tree must have very low correlations.

Below are some points that explain why we should use the DTC algorithm

- It takes less training time as compared to other algorithms.
- It predicts output with high accuracy, even for the large dataset it runs efficiently.
- It can also maintain accuracy when a large proportion of data is missing.

3.2 Drawbacks of Decision Tree Classifier

Decision Tree Classifiers are a popular machine learning algorithm known for their simplicity, interpretability, and versatility. However, they are not without their drawbacks. Here are some common drawbacks associated with Decision Tree Classifiers:

- **Overfitting**: Decision Trees are prone to overfitting, especially when the tree becomes too deep and complex. Overfitting occurs when the tree captures noise in the training data rather than the underlying patterns. This can lead to poor generalization to new, unseen data.
- **Instability**: Decision Trees are sensitive to small variations in the training data. A slight change in the data can result in a significantly different tree structure. This instability can make Decision Trees less reliable compared to other algorithms like XGBoosts or Gradient Boosting.

- High Variance: Decision Trees have high variance, meaning that small changes in the training data can result in different tree structures. High variance can lead to inconsistent predictions and reduced model reliability.
- Bias Towards Dominant Classes: Decision Trees tend to favor classes that are more frequent in the training data. In imbalanced datasets, where one class significantly outnumbers the others, Decision Trees can have a bias towards the majority class and perform poorly on minority classes.
- Lack of Predictive Power: Decision Trees may struggle with capturing complex relationships in the data, especially when the relationships are nonlinear. They might not perform well on datasets with intricate decision boundaries.
- Not Suitable for Numerical Predictors: Decision Trees are primarily designed for categorical or binary predictors. While they can handle numerical predictors, they may not perform as effectively as algorithms designed specifically for numerical data, such as linear regression.
- Difficulty Handling Missing Data: Traditional Decision Trees can struggle with missing data. If a predictor has missing values, the algorithm may discard the entire data point or introduce bias in the model.
- Limited Expressiveness: Decision Trees represent decision boundaries using axis-aligned splits. This means they might not perform well on problems where decision boundaries are diagonal or nonlinear, requiring more complex models.
- Not Optimized for Unstructured Data: Decision Trees are not well-suited for unstructured data types like text, audio, or images. They are primarily used for structured data.
- Greedy Nature: Decision Trees use a greedy approach to split the data at each node. They select the most informative feature at each step without considering future splits. This can lead to suboptimal trees.
- Prone to Outliers: Outliers in the data can disproportionately influence the decisions made by Decision Trees, potentially leading to less robust models.

CHAPTER 4

PROPOSED SYSTEM

4.1 Overview

In response to these challenges. The essence of the AI-driven approach involves training these models on meticulously labeled datasets containing examples of different surfaces. Through this training process, the models can autonomously learn to extract relevant features from sensor data, enabling the robot to discern and classify surfaces with heightened accuracy.

The provided Python script implements a graphical user interface (GUI) application using Tkinter for a surface identification project based on robot-sensed data. Here's a detailed explanation of the steps carried out by the application:

Dataset Upload: The application starts with a button labeled "Upload Dataset." When clicked, this button opens a file dialog, allowing the user to select the dataset file (assumed to be in CSV format). The chosen file is then loaded into the application, and its name is displayed in the text widget. The dataset is stored in the 'dataset' variable.

Dataset Preprocessing: The "Preprocess Dataset" button triggers the preprocessing phase. Missing values in the dataset are filled with zeros, and an overview of the dataset, including the first few records, is displayed in the text widget. Additionally, a count plot is generated to visualize the distribution of classes in the 'surface' column. Label encoding is applied to convert categorical class labels into numerical values.

Train-Test Splitting: The dataset is split into training and testing sets using the scikit-learn `train_test_split` function. Information about the total number of records in the dataset, as well as the training and testing sets, is displayed in the text widget.

Decision Tree Classifier: The "Decision Tree Classifier" button initiates the training of a Decision Tree classifier. The model is fitted on the training set, and predictions are made on the testing set. The evaluation metrics, including accuracy, confusion matrix, and classification report, are displayed. Additionally, a Receiver Operating Characteristic (ROC) graph is generated to visualize the model's performance.

Random Forest Classifier: The "Random Forest Classifier" button triggers the training of a Random Forest classifier. Similar to the Decision Tree model, evaluation metrics and a ROC graph are displayed in the text widget.

Prediction on Test Data: The "Prediction" button allows the user to select a file for making predictions using the trained Decision Tree classifier. Predictions are displayed in the text widget, indicating the predicted classes for each test data entry.

Performance Estimation and Comparison: The "Comparison Graph" button generates a bar graph comparing performance metrics (precision, recall, F1-score, and accuracy) between the Decision Tree classifier and the Random Forest classifier. This visual representation provides an easy comparison of the two models.

Exit: The "Exit" button closes the Tkinter GUI application.

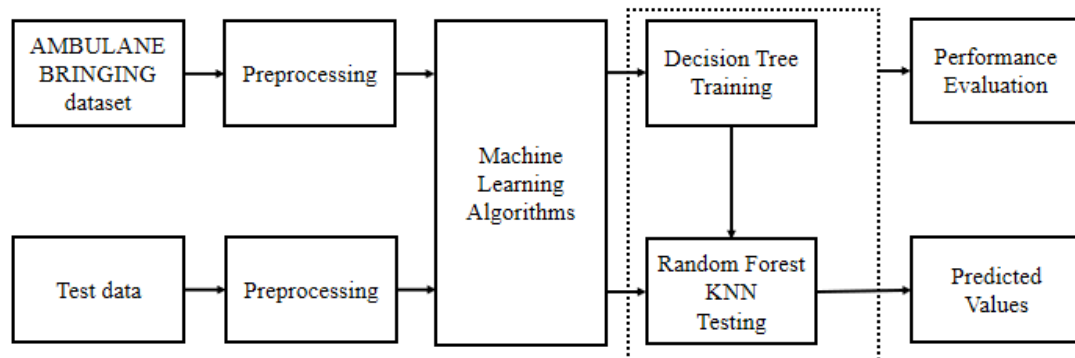


Fig. 1: Architectural Block diagram of proposed system.

4.2 Random Forest Algorithm

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model. As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on

one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output. The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

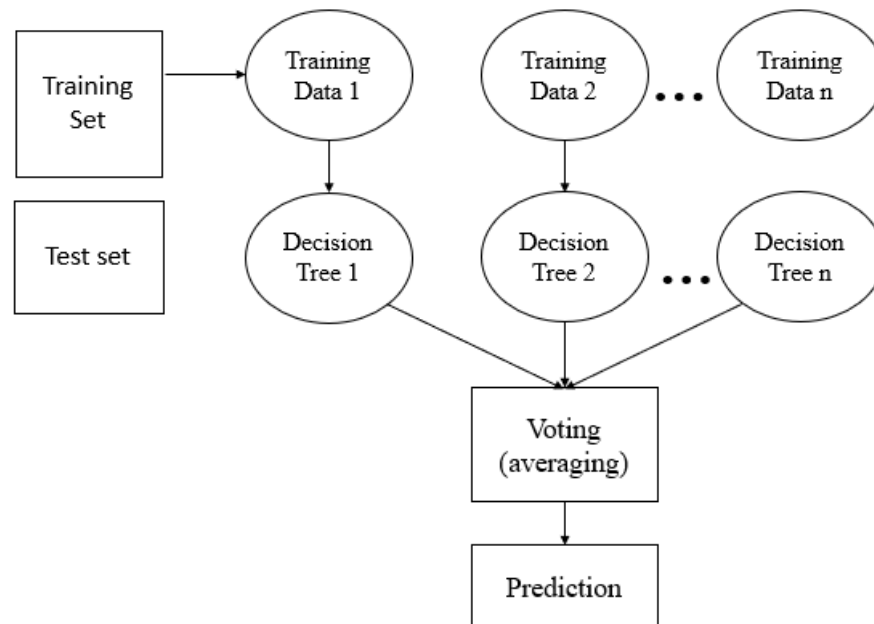


Fig. 4.2: Random Forest algorithm.

Random Forest algorithm

Step 1: In Random Forest n number of random records are taken from the data set having k number of records.

Step 2: Individual decision trees are constructed for each sample.

Step 3: Each decision tree will generate an output.

Step 4: Final output is considered based on Majority Voting or Averaging for Classification and regression respectively.

4.2.1 Important Features of Random Forest

- **Diversity**- Not all attributes/variables/features are considered while making an individual tree, each tree is different.
- **Immune to the curse of dimensionality**- Since each tree does not consider all the features, the feature space is reduced.
- **Parallelization**-Each tree is created independently out of different data and attributes. This means that we can make full use of the CPU to build random forests.

- **Train-Test split-** In a random forest we don't have to segregate the data for train and test as there will always be 30% of the data which is not seen by the decision tree.
- **Stability-** Stability arises because the result is based on majority voting/ averaging.

4.2.2 Assumptions for Random Forest

Since the random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all the trees predict the correct output. Therefore, below are two assumptions for a better Random Forest classifier:

- There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.
- The predictions from each tree must have very low correlations.

Below are some points that explain why we should use the Random Forest algorithm

- It takes less training time as compared to other algorithms.
- It predicts output with high accuracy, even for the large dataset it runs efficiently.
- It can also maintain accuracy when a large proportion of data is missing.

4.2.3 Types of Ensembles

Before understanding the working of the random forest, we must look into the ensemble technique. Ensemble simply means combining multiple models. Thus, a collection of models is used to make predictions rather than an individual model. Ensemble uses two types of methods:

Bagging– It creates a different training subset from sample training data with replacement & the final output is based on majority voting. For example, Random Forest. Bagging, also known as Bootstrap Aggregation is the ensemble technique used by random forest. Bagging chooses a random sample from the data set. Hence each model is generated from the samples (Bootstrap Samples) provided by the Original Data with replacement known as row sampling. This step of row sampling with replacement is called bootstrap. Now each model is trained independently which generates results. The final output is based on majority voting after combining the results of all models. This step which involves combining all the results and generating output based on majority voting is known as aggregation.

Boosting– It combines weak learners into strong learners by creating sequential models such that the final model has the highest accuracy. For example, ADA BOOST, XG BOOST.

4.3 Advantages

The presented Tkinter-based surface identification project utilizing Decision Tree and Random Forest classifiers offers several advantages:

- **User-Friendly Interface:** The graphical user interface (GUI) created with Tkinter enhances user interaction by providing buttons for various functionalities. This makes the application accessible and easy to use for individuals without programming expertise.
- **Dynamic Dataset Upload:** The ability to upload datasets through the "Upload Dataset" button allows users to work with diverse datasets effortlessly. This dynamic approach supports the application's adaptability to different use cases and datasets.
- **Comprehensive Preprocessing:** The "Preprocess Dataset" button automates preprocessing steps, such as handling missing values and label encoding. The generated count plot aids in visualizing the distribution of classes, offering insights into the dataset's characteristics.
- **Transparent Train-Test Splitting:** The application transparently communicates the process of splitting the dataset into training and testing sets. Information about the total records and the sizes of the training and testing sets is provided, enhancing transparency in the data preparation phase.
- **Multiple Classifier Options:** The inclusion of both Decision Tree and Random Forest classifiers offers flexibility to users. They can choose between different algorithms based on the nature of their data and the problem at hand, allowing for experimentation and model comparison.
- **Performance Metrics and Visualization:** The application computes and displays essential performance metrics, including accuracy, confusion matrix, and classification report. The incorporation of ROC curves visually represents the models' performance, aiding users in assessing the classifiers' ability to discriminate between classes.
- **Prediction on Test Data:** The "Prediction" button allows users to make predictions on new test data using the trained Decision Tree classifier. This functionality is valuable for real-world applications where the model is deployed on unseen data.
- **Comparison Graph:** The "Comparison Graph" button generates a bar graph comparing performance metrics between the Decision Tree and Random Forest classifiers. This visual

representation facilitates a quick and clear understanding of how different algorithms perform on the given dataset.

- Scalability and Adaptability: The modular structure of the application makes it scalable and adaptable. Users can extend the functionality by adding more classifiers or incorporating additional preprocessing steps to suit specific project requirements.
- Educational Value: The project serves as an educational tool for individuals learning about machine learning and classification problems. The GUI-based approach and step-by-step functionalities make it suitable for educational purposes and practical experimentation.

CHAPTER 5

UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group. The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language Is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

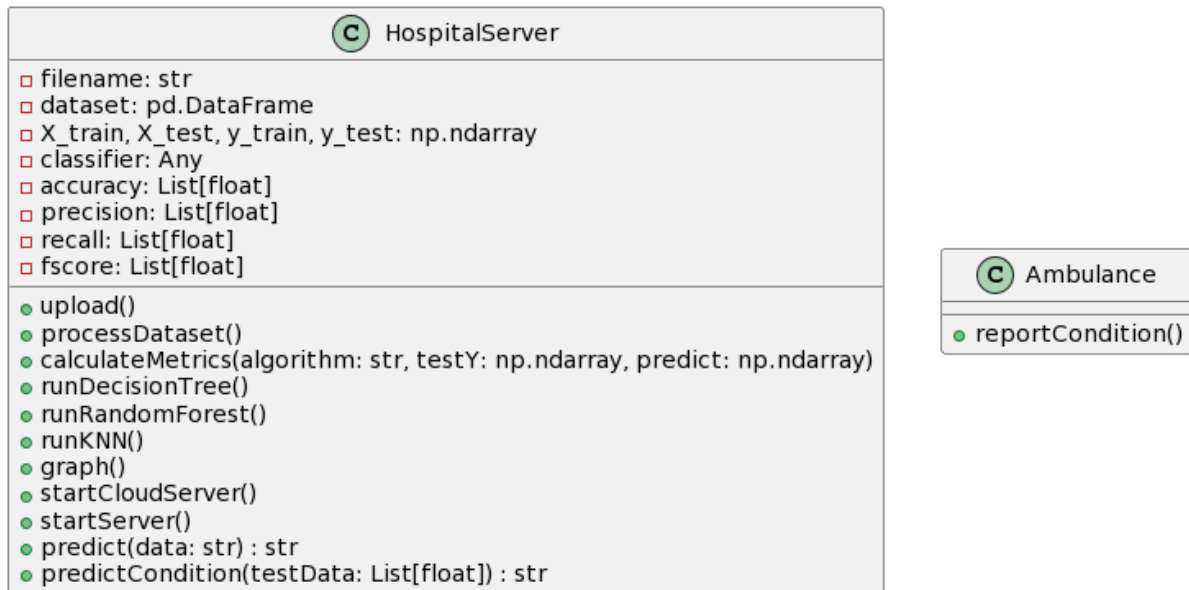
GOALS: The Primary goals in the design of the UML are as follows:

- Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
- Provide extendibility and specialization mechanisms to extend the core concepts.
- Be independent of particular programming languages and development process.
- Provide a formal basis for understanding the modeling language.
- Encourage the growth of OO tools market.
- Support higher level development concepts such as collaborations, frameworks, patterns and components.
- Integrate best practices.

Class Diagram

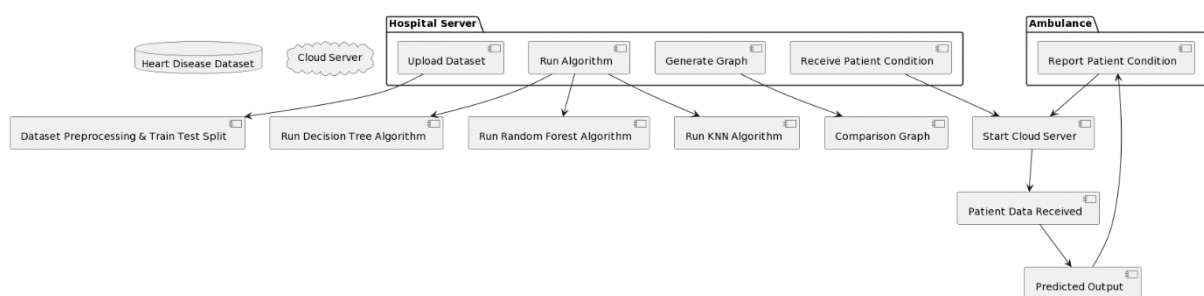
The class diagram is used to refine the use case diagram and define a detailed design of the system. The class diagram classifies the actors defined in the use case diagram into a set of interrelated classes. The relationship or association between the classes can be either an “is-a” or “has-a” relationship. Each class in the class diagram may be capable of providing certain functionalities.

These functionalities provided by the class are termed “methods” of the class. Apart from this, each class may have certain “attributes” that uniquely identify the class.



Data flow diagram

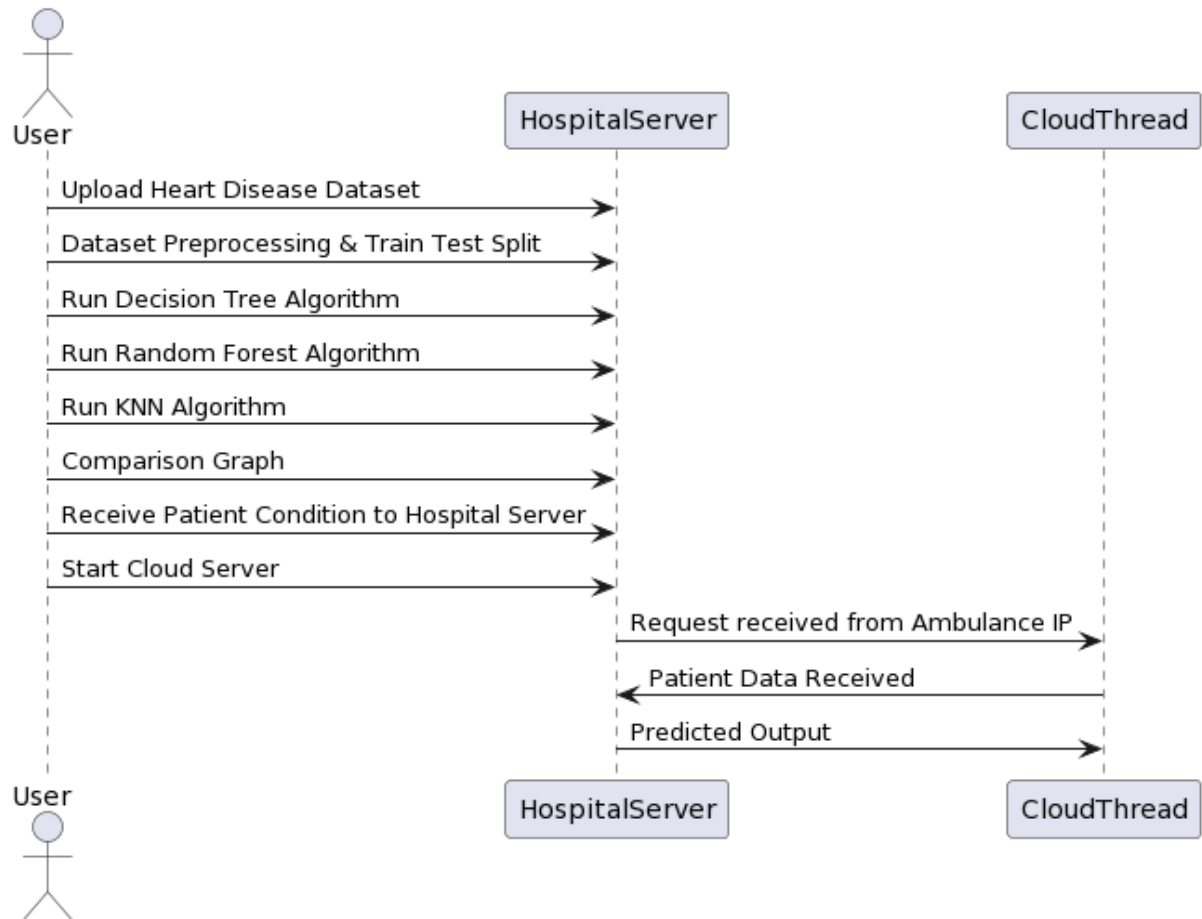
A Data Flow Diagram (DFD) is a visual representation of the flow of data within a system or process. It is a structured technique that focuses on how data moves through different processes and data stores within an organization or a system. DFDs are commonly used in system analysis and design to understand, document, and communicate data flow and processing.



Sequence Diagram

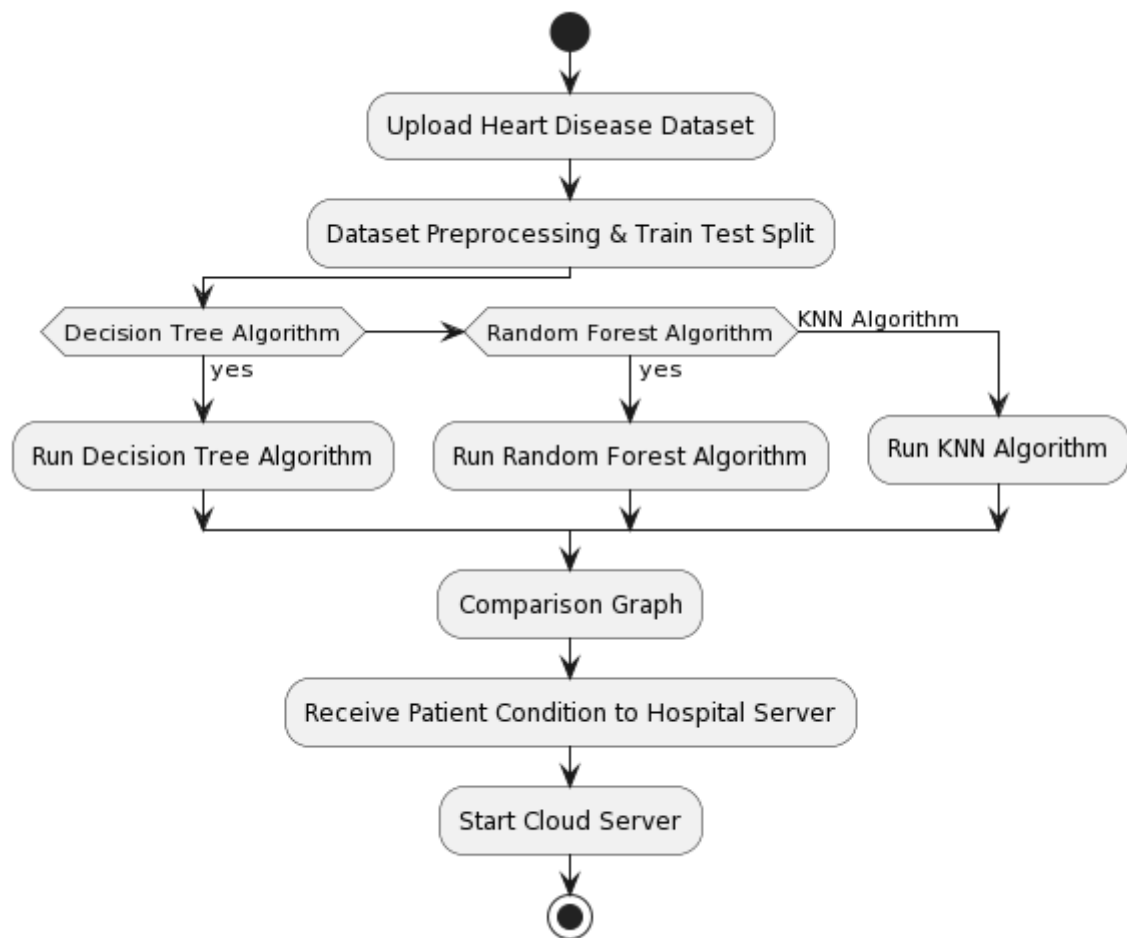
A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows, as parallel vertical lines (“lifelines”), different

processes or objects that live simultaneously, and as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

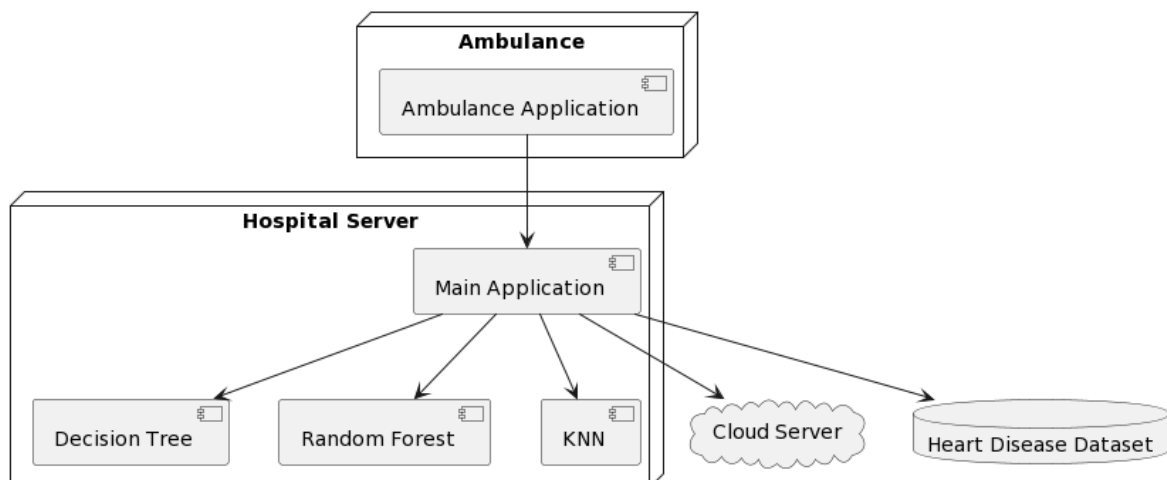


Activity diagram

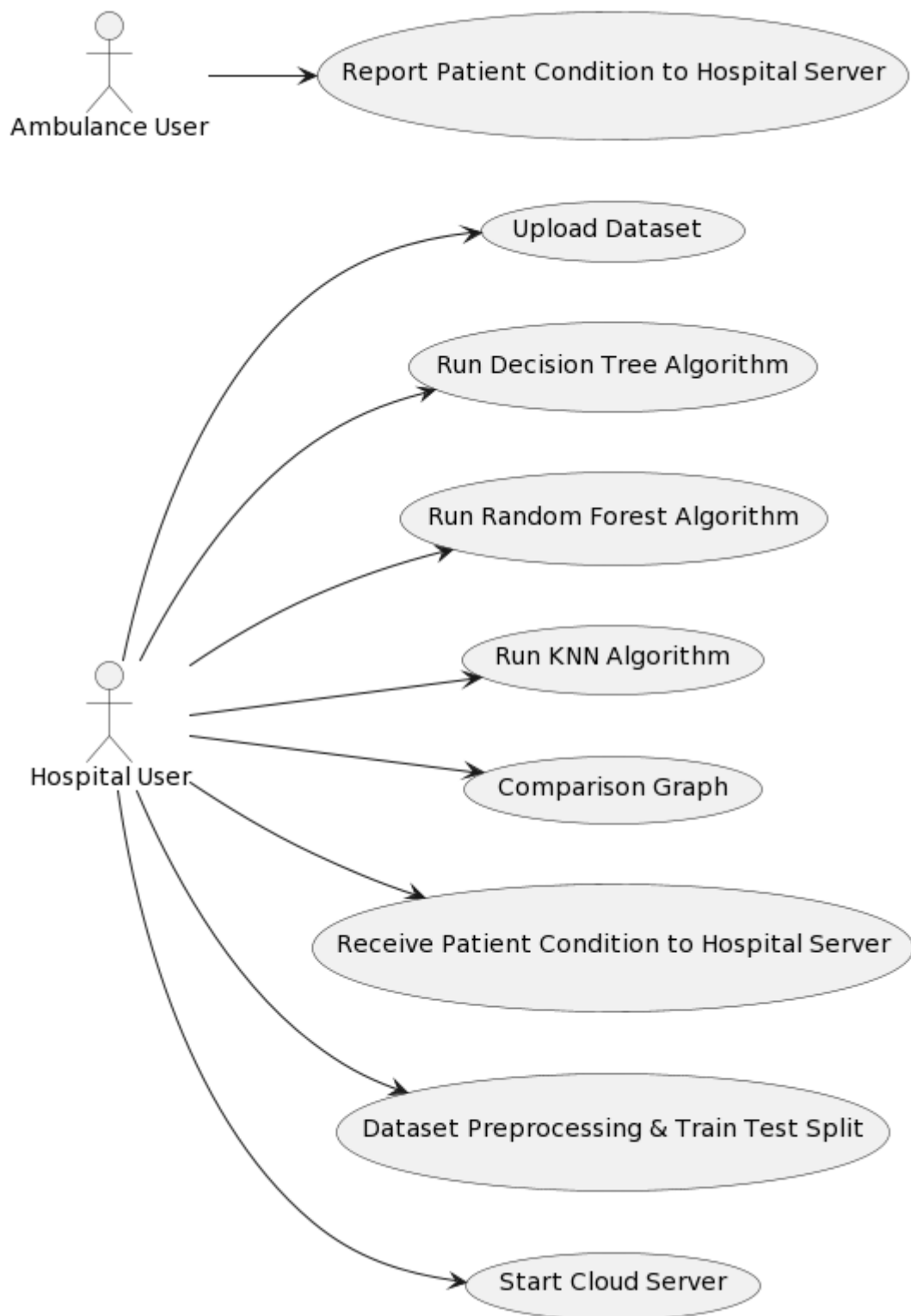
Activity diagram is another important diagram in UML to describe the dynamic aspects of the system.



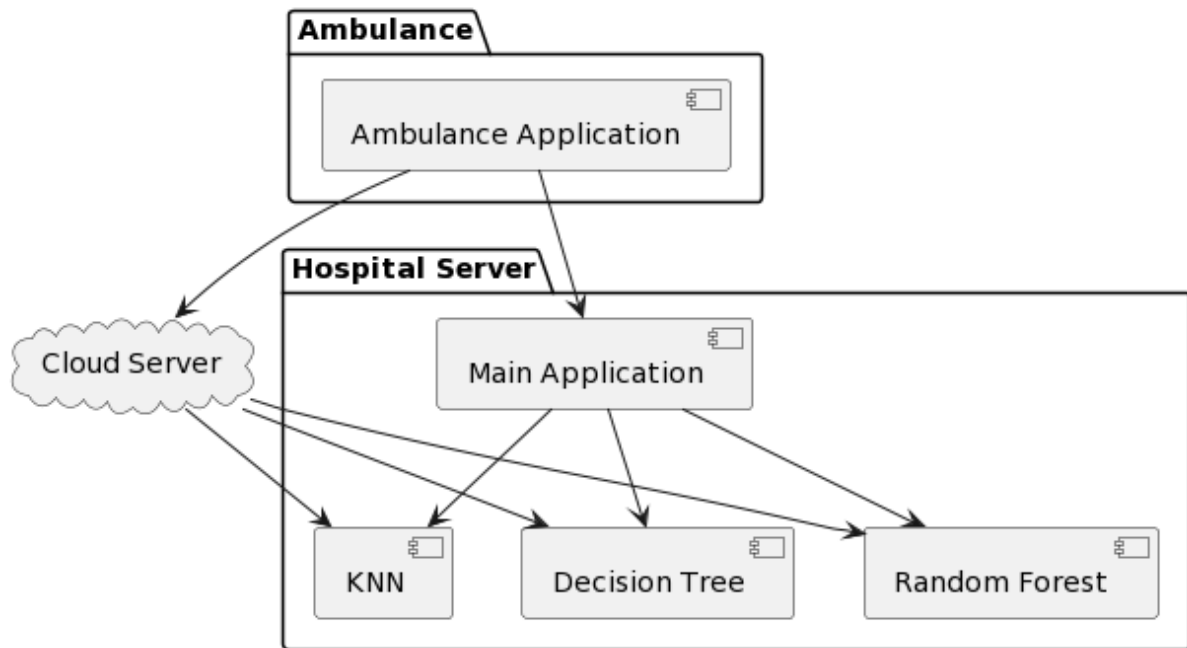
Deployment diagram: The deployment diagram visualizes the physical hardware on which the software will be deployed.



Use case diagram: The purpose of use case diagram is to capture the dynamic aspect of a system.



Component diagram: Component diagram describes the organization and wiring of the physical components in a system.



CHAPTER 6

SOFTWARE ENVIRONMENT

What is Python?

Below are some facts about Python.

- Python is currently the most widely used multi-purpose, high-level programming language.
- Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java.
- Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.
- Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber... etc.

The biggest strength of Python is huge collection of standard library which can be used for the following –

- Machine Learning
- GUI Applications (like Kivy, Tkinter, PyQt etc.)
- Web frameworks like Django (used by YouTube, Instagram, Dropbox)
- Image processing (like Opencv, Pillow)
- Web scraping (like Scrapy, BeautifulSoup, Selenium)
- Test frameworks
- Multimedia

Advantages of Python

Let's see how Python dominates over other languages.

1. Extensive Libraries

Python downloads with an extensive library and it contain code for various purposes like regular expressions, documentation-generation, unit-testing, web browsers, threading, databases, CGI, email, image manipulation, and more. So, we don't have to write the complete code for that manually.

2. Extensible

As we have seen earlier, Python can be extended to other languages. You can write some of your code in languages like C++ or C. This comes in handy, especially in projects.

3. Embeddable

Complimentary to extensibility, Python is embeddable as well. You can put your Python code in your source code of a different language, like C++. This lets us add scripting capabilities to our code in the other language.

4. Improved Productivity

The language's simplicity and extensive libraries render programmers more productive than languages like Java and C++ do. Also, the fact that you need to write less and get more things done.

5. IOT Opportunities

Since Python forms the basis of new platforms like Raspberry Pi, it finds the future bright for the Internet Of Things. This is a way to connect the language with the real world.

6. Simple and Easy

When working with Java, you may have to create a class to print 'Hello World'. But in Python, just a print statement will do. It is also quite easy to learn, understand, and code. This is why when people pick up Python, they have a hard time adjusting to other more verbose languages like Java.

7. Readable

Because it is not such a verbose language, reading Python is much like reading English. This is the reason why it is so easy to learn, understand, and code. It also does not need curly braces to define blocks, and indentation is mandatory. This further aids the readability of the code.

8. Object-Oriented

This language supports both the procedural and object-oriented programming paradigms. While functions help us with code reusability, classes and objects let us model the real world. A class allows the encapsulation of data and functions into one.

9. Free and Open-Source

Like we said earlier, Python is freely available. But not only can you download Python for free, but you can also download its source code, make changes to it, and even distribute it. It downloads with an extensive collection of libraries to help you with your tasks.

10. Portable

When you code your project in a language like C++, you may need to make some changes to it if you want to run it on another platform. But it isn't the same with Python. Here, you need to code only once, and you can run it anywhere. This is called Write Once Run Anywhere (WORA). However, you need to be careful enough not to include any system-dependent features.

11. Interpreted

Lastly, we will say that it is an interpreted language. Since statements are executed one by one, debugging is easier than in compiled languages.

Any doubts till now in the advantages of Python? Mention in the comment section.

Advantages of Python Over Other Languages

12. Less Coding

Almost all of the tasks done in Python requires less coding when the same task is done in other languages. Python also has an awesome standard library support, so you don't have to search for any third-party libraries to get your job done. This is the reason that many people suggest learning Python to beginners.

13. Affordable

Python is free therefore individuals, small companies or big organizations can leverage the free available resources to build applications. Python is popular and widely used so it gives you better community support.

The 2019 Github annual survey showed us that Python has overtaken Java in the most popular programming language category.

14. Python is for Everyone

Python code can run on any machine whether it is Linux, Mac or Windows. Programmers need to learn different languages for different jobs but with Python, you can professionally build web apps, perform data analysis and machine learning, automate things, do web scraping and also build games and powerful visualizations. It is an all-rounder programming language.

Disadvantages of Python

So far, we've seen why Python is a great choice for your project. But if you choose it, you should be aware of its consequences as well. Let's now see the downsides of choosing Python over another language.

1. Speed Limitations

We have seen that Python code is executed line by line. But since Python is interpreted, it often results in slow execution. This, however, isn't a problem unless speed is a focal point for the project. In other words, unless high speed is a requirement, the benefits offered by Python are enough to distract us from its speed limitations.

2. Weak in Mobile Computing and Browsers

While it serves as an excellent server-side language, Python is much rarely seen on the client-side. Besides that, it is rarely ever used to implement smartphone-based applications. One such application is called Carbonnelle.

The reason it is not so famous despite the existence of Brython is that it isn't that secure.

3. Design Restrictions

As you know, Python is dynamically typed. This means that you don't need to declare the type of variable while writing the code. It uses duck-typing. But wait, what's that? Well, it just means that if it looks like a duck, it must be a duck. While this is easy on the programmers during coding, it can raise run-time errors.

4. Underdeveloped Database Access Layers

Compared to more widely used technologies like JDBC (Java DataBase Connectivity) and ODBC (Open DataBase Connectivity), Python's database access layers are a bit underdeveloped. Consequently, it is less often applied in huge enterprises.

5. Simple

No, we're not kidding. Python's simplicity can indeed be a problem. Take my example. I don't do Java, I'm more of a Python person. To me, its syntax is so simple that the verbosity of Java code seems unnecessary.

This was all about the Advantages and Disadvantages of Python Programming Language.

Modules Used in Project

NumPy

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary datatypes can be defined using NumPy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

Scikit – learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use. Python

Install Python Step-by-Step in Windows and Mac

Python a versatile programming language doesn't come pre-installed on your computer devices. Python was first released in the year 1991 and until today it is a very popular high-level programming language. Its style philosophy emphasizes code readability with its notable use of great whitespace.

The object-oriented approach and language construct provided by Python enables programmers to write both clear and logical code for projects. This software does not come pre-packaged with Windows.

How to Install Python on Windows and Mac

There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python but this tutorial will solve your query. The latest or the newest version of Python is version 3.7.4 or in other words, it is Python 3.

Note: The python version 3.7.4 cannot be used on Windows XP or earlier devices.

Before you start with the installation process of Python. First, you need to know about your System Requirements. Based on your system type i.e. operating system and based processor, you must download the python version. My system type is a Windows 64-bit operating system. So the steps below are to install python version 3.7.4 on Windows 7 device or to install Python 3. Download the Python Cheatsheet here. The steps on how to install Python on Windows 10, 8 and 7 are divided into 4 parts to help understand better.

Download the Correct version into the system

Step 1: Go to the official site to download and install python using Google Chrome or any other web browser. OR Click on the following link: <https://www.python.org>



Now, check for the latest and the correct version for your operating system.

Step 2: Click on the Download Tab.



Step 3: You can either select the Download Python for windows 3.7.4 button in Yellow Color or you can scroll further down and click on download with respective to their version. Here, we are downloading the most recent python version for windows 3.7.4

Looking for a specific release?

Python releases by version number:

Release version	Release date		Click for more
Python 3.7.4	July 8, 2019	Download	Release Notes
Python 3.6.9	July 2, 2019	Download	Release Notes
Python 3.7.3	March 25, 2019	Download	Release Notes
Python 3.4.10	March 18, 2019	Download	Release Notes
Python 3.5.7	March 18, 2019	Download	Release Notes
Python 2.7.16	March 4, 2019	Download	Release Notes
Python 3.7.2	Dec. 24, 2018	Download	Release Notes

Step 4: Scroll down the page until you find the Files option.

Step 5: Here you see a different version of python along with the operating system.

Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
Clipped source tarball	Source release		6811671e5b2db4aef7b9ab01f09be	23017663	SIG
XZ compressed source tarball	Source release		d33e4aaf6097051c2eca45ee3604803	17131432	SIG
macOS 64-bit/32-bit installer	Mac OS X	for Mac OS X 10.6 and later	6428b4fa7583daf1a442c8a4ee08e6	34898416	SIG
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later	5dd605c38217a45773bf5e4a936b24f	20882845	SIG
Windows help file	Windows		d83999573a2c56b2ac56cade6b47cd2	8131761	SIG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	980b3c7fd9ec0b9abef3184a40729a2	7504291	SIG
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	a702b4b0ad76db9db3c3a3a83e563400	2688368	SIG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	28cb1c608b6d73aeb53a3bd351b4bd2	1362904	SIG
Windows x86 embeddable zip file	Windows		9fab3b818841879fda94113574139d8	6741626	SIG
Windows x86 executable installer	Windows		33cc802942a5446a3d845147e294789	25663848	SIG
Windows x86 web-based installer	Windows		1b670cfa5d317df82c309e3ea371d87c	1324608	SIG

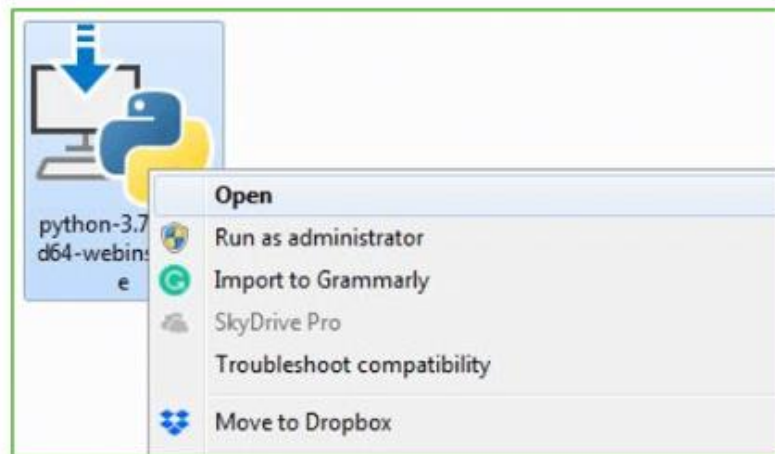
- To download Windows 32-bit python, you can select any one from the three options: Windows x86 embeddable zip file, Windows x86 executable installer or Windows x86 web-based installer.
- To download Windows 64-bit python, you can select any one from the three options: Windows x86-64 embeddable zip file, Windows x86-64 executable installer or Windows x86-64 web-based installer.

Here we will install Windows x86-64 web-based installer. Here your first part regarding which version of python is to be downloaded is completed. Now we move ahead with the second part in installing python i.e. Installation

Note: To know the changes or updates that are made in the version you can click on the Release Note Option.

Installation of Python

Step 1: Go to Download and Open the downloaded python version to carry out the installation process.



Step 2: Before you click on Install Now, Make sure to put a tick on Add Python 3.7 to PATH.



Step 3: Click on Install NOW After the installation is successful. Click on Close.



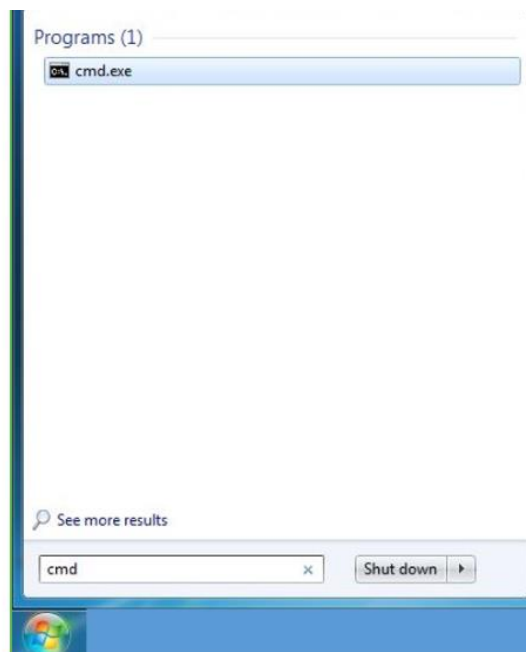
With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.

Note: The installation process might take a couple of minutes.

Verify the Python Installation

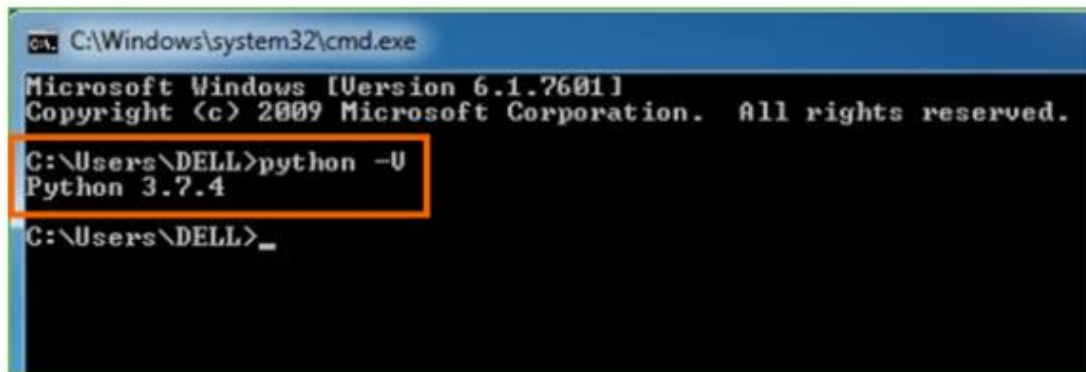
Step 1: Click on Start

Step 2: In the Windows Run Command, type “cmd”.



Step 3: Open the Command prompt option.

Step 4: Let us test whether the python is correctly installed. Type python -V and press Enter.



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\DELL>python -U
Python 3.7.4

C:\Users\DELL>_
```

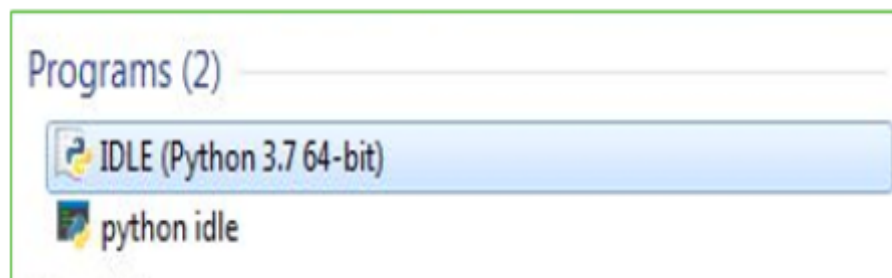
Step 5: You will get the answer as 3.7.4

Note: If you have any of the earlier versions of Python already installed. You must first uninstall the earlier version and then install the new one.

Check how the Python IDLE works

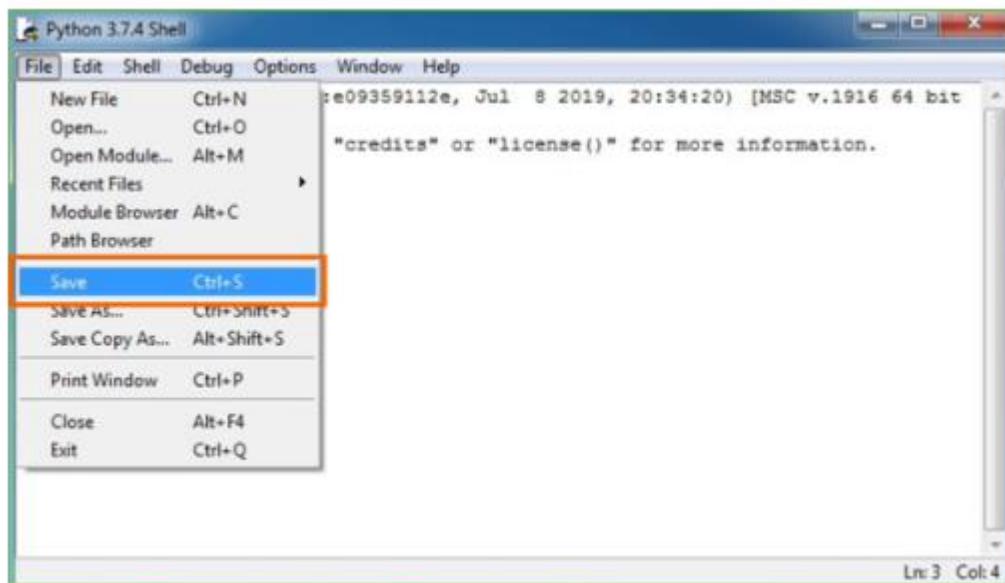
Step 1: Click on Start

Step 2: In the Windows Run command, type “python idle”.



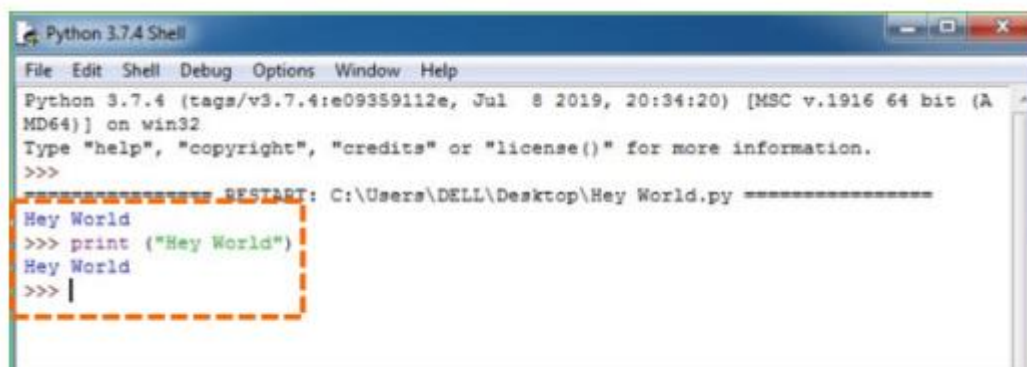
Step 3: Click on IDLE (Python 3.7 64-bit) and launch the program

Step 4: To go ahead with working in IDLE you must first save the file. Click on File > Click on Save



Step 5: Name the file and save as type should be Python files. Click on SAVE. Here I have named the files as Hey World.

Step 6: Now for e.g. enter print ("Hey World") and Press Enter.



You will see that the command given is launched. With this, we end our tutorial on how to install Python. You have learned how to download python for windows into your respective operating system.

Note: Unlike Java, Python does not need semicolons at the end of the statements otherwise it won't work.

CHAPTER 7

SYSTEM REQUIREMENTS

Software Requirements

The functional requirements or the overall description documents include the product perspective and features, operating system and operating environment, graphics requirements, design constraints and user documentation.

The appropriation of requirements and implementation constraints gives the general overview of the project in regard to what the areas of strength and deficit are and how to tackle them.

- Python IDLE 3.7 version (or)
- Anaconda 3.7 (or)
- Jupiter (or)
- Google colab

Hardware Requirements

Minimum hardware requirements are very dependent on the particular software being developed by a given Enthought Python / Canopy / VS Code user. Applications that need to store large arrays/objects in memory will require more RAM, whereas applications that need to perform numerous calculations or tasks more quickly will require a faster processor.

- Operating system : Windows, Linux
- Processor : minimum intel i3
- Ram : minimum 4 GB
- Hard disk : minimum 250GB

CHAPTER 8

FUNCTIONAL REQUIREMENTS

Output Design

Outputs from computer systems are required primarily to communicate the results of processing to users. They are also used to provide a permanent copy of the results for later consultation. The various types of outputs in general are:

- External Outputs, whose destination is outside the organization
- Internal Outputs whose destination is within organization and they are the
- User's main interface with the computer.
- Operational outputs whose use is purely within the computer department.
- Interface outputs, which involve the user in communicating directly.

Output Definition

The outputs should be defined in terms of the following points:

- Type of the output
- Content of the output
- Format of the output
- Location of the output
- Frequency of the output
- Volume of the output
- Sequence of the output

It is not always desirable to print or display data as it is held on a computer. It should be decided as which form of the output is the most suitable.

Input Design

Input design is a part of overall system design. The main objective during the input design is as given below:

- To produce a cost-effective method of input.

- To achieve the highest possible level of accuracy.
- To ensure that the input is acceptable and understood by the user.

Input Stages

The main input stages can be listed as below:

- Data recording
- Data transcription
- Data conversion
- Data verification
- Data control
- Data transmission
- Data validation
- Data correction

Input Types

It is necessary to determine the various types of inputs. Inputs can be categorized as follows:

- External inputs, which are prime inputs for the system.
- Internal inputs, which are user communications with the system.
- Operational, which are computer department's communications to the system?
- Interactive, which are inputs entered during a dialogue.

Input Media

At this stage choice has to be made about the input media. To conclude about the input media consideration has to be given to;

- Type of input
- Flexibility of format
- Speed
- Accuracy
- Verification methods
- Rejection rates
- Ease of correction

- Storage and handling requirements
- Security
- Easy to use
- Portability

Keeping in view the above description of the input types and input media, it can be said that most of the inputs are of the form of internal and interactive. As

Input data is to be the directly keyed in by the user, the keyboard can be considered to be the most suitable input device.

Error Avoidance

At this stage care is to be taken to ensure that input data remains accurate form the stage at which it is recorded up to the stage in which the data is accepted by the system. This can be achieved only by means of careful control each time the data is handled.

Error Detection

Even though every effort is make to avoid the occurrence of errors, still a small proportion of errors is always likely to occur, these types of errors can be discovered by using validations to check the input data.

Data Validation

Procedures are designed to detect errors in data at a lower level of detail. Data validations have been included in the system in almost every area where there is a possibility for the user to commit errors. The system will not accept invalid data. Whenever an invalid data is keyed in, the system immediately prompts the user and the user has to again key in the data and the system will accept the data only if the data is correct. Validations have been included where necessary.

The system is designed to be a user friendly one. In other words the system has been designed to communicate effectively with the user. The system has been designed with popup menus.

User Interface Design

It is essential to consult the system users and discuss their needs while designing the user interface:

User Interface Systems Can Be Broadly Clasified As:

- User initiated interface the user is in charge, controlling the progress of the user/computer dialogue. In the computer-initiated interface, the computer selects the next stage in the interaction.
- Computer initiated interfaces

In the computer-initiated interfaces the computer guides the progress of the user/computer dialogue. Information is displayed and the user response of the computer takes action or displays further information.

User Initiated Interfaces

User initiated interfaces fall into two approximate classes:

- Command driven interfaces: In this type of interface the user inputs commands or queries which are interpreted by the computer.
- Forms oriented interface: The user calls up an image of the form to his/her screen and fills in the form. The forms-oriented interface is chosen because it is the best choice.

Computer-Initiated Interfaces

The following computer – initiated interfaces were used:

- The menu system for the user is presented with a list of alternatives and the user chooses one; of alternatives.
- Questions – answer type dialog system where the computer asks question and takes action based on the basis of the users reply.

Right from the start the system is going to be menu driven, the opening menu displays the available options. Choosing one option gives another popup menu with more options. In this way every option leads the users to data entry form where the user can key in the data.

Error Message Design

The design of error messages is an important part of the user interface design. As user is bound to commit some errors or other while designing a system the system should be designed to be helpful by providing the user with information regarding the error he/she has committed.

This application must be able to produce output at different modules for different inputs.

Performance Requirements

Performance is measured in terms of the output provided by the application. Requirement specification plays an important part in the analysis of a system. Only when the requirement specifications are properly given, it is possible to design a system, which will fit into required environment. It rests largely in the part of the users of the existing system to give the requirement specifications because they are the people who finally use the system. This is because the requirements have to be known during the initial stages so that the system can be designed according to those requirements. It is very difficult to change the system once it has been designed and on the other hand designing a system, which does not cater to the requirements of the user, is of no use.

The requirement specification for any system can be broadly stated as given below:

- The system should be able to interface with the existing system
- The system should be accurate
- The system should be better than the existing system
- The existing system is completely dependent on the user to perform all the duties.

CHAPTER 9

SOURCE CODE

Hospital server

```
from tkinter import messagebox

from tkinter import *

from tkinter.filedialog import askopenfilename

from tkinter import simpledialog

import tkinter

import numpy as np

from tkinter import filedialog

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.metrics import confusion_matrix

import seaborn as sns

from sklearn.ensemble import RandomForestClassifier

from sklearn.tree import DecisionTreeClassifier

from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import accuracy_score

from sklearn.model_selection import train_test_split

from sklearn.metrics import precision_score

from sklearn.metrics import recall_score

from sklearn.metrics import f1_score

from sklearn.preprocessing import StandardScaler
```

```

import socket

from threading import Thread

from socketserver import ThreadingMixIn

import pickle


main = tkinter.Tk()

main.title("Intelligent Ambulance: Bringing AI and Human Interface Technology")

main.geometry("1300x1200")


global filename

global dataset

accuracy = []

precision = []

recall = []

fscore = []

global X_train, X_test, y_train, y_test

global classifier

global X, Y, sc

labels = ['Normal Heart Rate', 'Abnormal Heart Rate']


def upload():

    global filename

    global dataset

```

```

text.delete('1.0', END)

filename = filedialog.askopenfilename(initialdir = "Dataset")

text.delete('1.0', END)

text.insert(END,filename+' dataset loaded\n')

dataset = pd.read_csv(filename)

dataset.fillna(0, inplace = True)

text.insert(END,str(dataset.head())+"\n")

text.insert(END,"Dataset contains total records   : "+str(dataset.shape[0])+"\n")

text.insert(END,"Dataset contains total attributes : "+str(dataset.shape[1])+"\n")

label = dataset.groupby('target').size()

label.plot(kind="bar")

plt.show()

```

```

def processDataset():

    global X_train, X_test, y_train, y_test, sc

    global X, Y

    global dataset

    text.delete('1.0', END)

    data = dataset.values

    X = data[:,0:data.shape[1]-1]

    Y = data[:,data.shape[1]-1]

    Y = Y.astype(int)

    indices = np.arange(X.shape[0])

    np.random.shuffle(indices) #shuffling the dataset

```

```

X = X[indices]

Y = Y[indices]

sc = StandardScaler()

X = sc.fit_transform(X)

text.insert(END,"Dataset Preprocessing, Normalizing & Shuffling Task Completed\n")

text.insert(END,str(X)+"\n\n")

X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2, random_state=0)

text.insert(END,"Dataset train and test split details\n\n")

text.insert(END,"80% records used for Ensemble training Algorithm :
"+str(X_train.shape[0])+"\n")

text.insert(END,"20% records used for Ensemble testing Algorithm :
"+str(X_test.shape[0])+"\n")

def calculateMetrics(algorithm, testY, predict):

    global labels

    p = precision_score(testY, predict,average='macro') * 100

    r = recall_score(testY, predict,average='macro') * 100

    f = f1_score(testY, predict,average='macro') * 100

    a = accuracy_score(testY,predict)*100

    accuracy.append(a)

    precision.append(p)

    recall.append(r)

    fscore.append(f)

    text.insert(END,algorithm+" Accuracy : "+str(a)+"\n")

    text.insert(END,algorithm+" Precision : "+str(p)+"\n")

```



```

text.insert(END, algorithm+" Recall   : "+str(r)+"\n")

text.insert(END, algorithm+" FSCORE   : "+str(f)+"\n\n")

conf_matrix = confusion_matrix(testY, predict)

ax = sns.heatmap(conf_matrix, xticklabels = labels, yticklabels = labels, annot = True,
cmap="viridis" ,fmt ="g");

ax.set_ylim([0,len(labels)])

plt.title(algorithm+" Confusion matrix")

plt.ylabel('True class')

plt.xlabel('Predicted class')

plt.show()

```

```

def runDecisionTree():

    text.delete('1.0', END)

    global accuracy, precision, recall, fscore

    accuracy.clear()

    precision.clear()

    recall.clear()

    fscore.clear()

    global X_train, X_test, y_train, y_test

    dt_cls = DecisionTreeClassifier()

    dt_cls.fit(X_train, y_train)

    predict = dt_cls.predict(X_test)

    calculateMetrics("Decision Tree", y_test, predict)

```

```
def runRandomForest():

    global X_train, X_test, y_train, y_test, classifier

    rf_cls = RandomForestClassifier()

    rf_cls.fit(X_train, y_train)

    classifier = rf_cls

    predict = rf_cls.predict(X_test)

    calculateMetrics("Random Forest", y_test, predict)
```

```
def runKNN():

    global X_train, X_test, y_train, y_test, classifier

    knn_cls = KNeighborsClassifier(n_neighbors = 10)

    knn_cls.fit(X_train, y_train)

    predict = knn_cls.predict(X_test)

    calculateMetrics("KNN", y_test, predict)
```

```
def graph():

df=pd.DataFrame([[['DecisionTree','Precision',precision[0]],['DecisionTree','Recall',recall[0]],['DecisionTree','F1Score',fscore[0]],['DecisionTree','Accuracy',accuracy[0]],['RandomForest','Precision',precision[1]],['RandomForest','Recall',recall[1]],['RandomForest','F1Score',fscore[1]],['RandomForest','Accuracy',accuracy[1]],['KNN','Precision',precision[2]],['KNN','Recall',recall[2]],['KNN','F1Score',fscore[2]],['KNN','Accuracy',accuracy[2]]],columns=['Parameters','Algorithms','Value'])

df.pivot("Parameters", "Algorithms", "Value").plot(kind='bar')

plt.show()
```

```
running = True
```

```
def startCloudServer():
```

```
    global text
```

```
    class CloudThread(Thread):
```

```
        def __init__(self,ip,port):
```

```
            Thread.__init__(self)
```

```
            global text
```

```
            self.ip = ip
```

```
            self.port = port
```

```
            print('Request received from Ambulance IP : '+ip+' with port no : '+str(port)+'\n')
```

```
    def run(self):
```

```
        data = conn.recv(1000)
```

```
        dataset = pickle.loads(data)
```

```
        request = dataset[0]
```

```
        if request == "patientdata":
```

```
            data = dataset[1]
```

```
            text.delete('1.0', END)
```

```
            text.insert(END,"Patient Data Received : "+str(data)+"\n")
```

```
            text.update_idletasks()
```

```
            output = predict(data)
```

```
            conn.send(output.encode())
```

```
            text.insert(END,output+"\n\n")
```

```

server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

server.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)

server.bind(('localhost', 2222))

threads = []

text.insert(END, "Cloud Server Started\n\n")

while running:

    server.listen(4)

    (conn, (ip, port)) = server.accept()

    newthread = CloudThread(ip, port)

    newthread.start()

    threads.append(newthread)

for t in threads:

    t.join()

def startServer():

    Thread(target=startCloudServer).start()

def predict(data):

    data = data.split(",")

    temp = []

    for i in range(len(data)):

        temp.append(float(data[i]))

    return predictCondition(temp)

```

```

def predictCondition(testData):

    global sc, classifier

    data = []

    data.append(testData)

    data = np.asarray(data)

    data = sc.transform(data)

    predict = classifier.predict(data)

    predict = predict[0]

    msg = "Predicted Output: Patient Condition Normal"

    if predict == 1:

        msg = "Predicted Output: Patient Condition Abnormal"

    return msg

```

```

font = ('times', 16, 'bold')

title = Label(main, text="Intelligent Ambulance: Bringing AI and Human Interface Technology")

title.config(bg='dark goldenrod', fg='white')

title.config(font=font)

title.config(height=3, width=120)

title.place(x=0,y=5)

```

```

font1 = ('times', 13, 'bold')

upload = Button(main, text="Upload Heart Disease Dataset", command=upload)

upload.place(x=890,y=100)

```

```
upload.config(font=font1)
```

```
processButton = Button(main, text="Dataset Preprocessing & Train Test Split",  
command=processDataset)
```

```
processButton.place(x=890,y=150)
```

```
processButton.config(font=font1)
```

```
dtButton = Button(main, text="Run Decision Tree Algorithm", command=runDecisionTree)
```

```
dtButton.place(x=890,y=200)
```

```
dtButton.config(font=font1)
```

```
rfButton = Button(main, text="Run Random Forest Algorithm", command=runRandomForest)
```

```
rfButton.place(x=890,y=250)
```

```
rfButton.config(font=font1)
```

```
knnButton = Button(main, text="Run KNN Algorithm", command=runKNN)
```

```
knnButton.place(x=890,y=300)
```

```
knnButton.config(font=font1)
```

```
graphButton = Button(main, text="Comparison Graph", command=graph)
```

```
graphButton.place(x=890,y=350)
```

```
graphButton.config(font=font1)
```

```
predictButton = Button(main, text="Receive Patient Condition to Hospital Server",  
command=startServer)
```

```
predictButton.place(x=890,y=400)
```

```
predictButton.config(font=font1)
```

```
font1 = ('times', 12, 'bold')
```

```
text=Text(main,height=30,width=100)
```

```
scroll=Scrollbar(text)
```

```
text.configure(yscrollcommand=scroll.set)
```

```
text.place(x=10,y=100)
```

```
text.config(font=font1)
```

```
main.config(bg='lightgreen')
```

```
main.mainloop()
```

Ambulance

```
from tkinter import messagebox
```

```
from tkinter import *
```

```
from tkinter import simpledialog
```

```
import tkinter
```

```
import numpy as np
```

```
import pandas as pd
```

```
from tkinter.filedialog import askopenfilename
```

```
import tkinter
```

```
import numpy as np
```

```

from tkinter import filedialog

import socket

import pickle

import time


main = tkinter.Tk()

main.title("Ambulance Application")

main.geometry("900x500")


def reportCondition():

    text.delete('1.0', END)

    filename = filedialog.askopenfilename(initialdir = "Dataset")

    dataset = pd.read_csv(filename)

    dataset.fillna(0, inplace = True)

    dataset = dataset.values

    for i in range(len(dataset)):

        data = dataset[i]

        arr = []

        for j in range(len(data)):

            arr.append(str(data[j]))

        arr = ','.join(arr)

        client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

        client.connect(('localhost', 2222))

        features = []

```



```

features.append("patientdata")

features.append(arr)

features = pickle.dumps(features)

client.send(features)

data = client.recv(1000)

data = data.decode()

text.insert(END, "Patient Test Data = "+arr+" ==> "+data+"\n\n")

text.update()

text.update_idletasks()

time.sleep(1)

```

```

font = ('times', 16, 'bold')

title = Label(main, text='Ambulance Application')

title.config(bg='dark goldenrod', fg='white')

title.config(font=font)

title.config(height=3, width=120)

title.place(x=0,y=5)

```

```

font1 = ('times', 13, 'bold')

upload = Button(main, text="Report Patient Condition to Hospital Server",
command=reportCondition)

upload.place(x=200,y=450)

upload.config(font=font1)

```

```
font1 = ('times', 12, 'bold')

text=Text(main,height=18,width=110)

scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=10,y=100)

text.config(font=font1)


main.config(bg='lightgreen')

main.mainloop()
```

CHAPTER 10

RESULTS AND DISCUSSION

10.1 Implementation description:

The code is of two parts: the "Hospital server" and the "Ambulance" application.

Let's break down each part:

Hospital Server:

- Import Statements: Various libraries are imported, including Tkinter for GUI, NumPy, Pandas for data manipulation, Matplotlib and Seaborn for data visualization, and scikit-learn for machine learning functionalities.
- Global Variables: Global variables are declared to store information about the dataset, training data, classifiers, and various metrics like accuracy, precision, recall, and F1 score.
- GUI Setup: The Tkinter GUI is set up with buttons for uploading a dataset, preprocessing the dataset, running machine learning algorithms (Decision Tree, Random Forest, KNN), displaying comparison graphs, and starting a server to receive patient conditions.
- Function Definitions: Functions are defined for uploading a dataset, preprocessing the dataset, calculating machine learning metrics, running Decision Tree, Random Forest, and KNN algorithms, displaying comparison graphs, and starting a server to receive patient conditions.
- Server Implementation: The code implements a simple socket server to receive patient data from an ambulance. The server listens on localhost and port 2222, and it spawns a new thread for each incoming connection. The server receives patient data, predicts the patient's condition using a trained classifier, and sends the prediction back to the ambulance.
- Ambulance Prediction: The code defines a predictCondition function to predict the patient's condition using a trained classifier. The ambulance can upload a dataset of patient information, and for each patient, it connects to the hospital server to get the predicted condition.
- Graphical User Interface (GUI): The GUI components include buttons for various actions and a text widget to display information and results.

—

Ambulance Application:

- Import Statements: Similar libraries are imported as in the hospital server, including Tkinter for GUI, NumPy, Pandas, and socket for network communication.
- GUI Setup: The Tkinter GUI is set up with a button for the ambulance to report patient conditions to the hospital server.
- Function Definitions: The code defines a function reportCondition for the ambulance to report patient conditions. It opens a file dialog to select a dataset, connects to the hospital server, sends patient data, and receives and displays the predicted condition.
- Graphical User Interface (GUI): The GUI components include a button for the ambulance to report patient conditions and a text widget to display information and results.

10.2 Dataset description:

The dataset contains features of cardiovascular health. Here's a brief description of each column:

- age: This represents the age of an individual.
- sex: This column represents the gender of an individual, encoded as binary values (e.g., 0 for female, 1 for male).
- cp: This column represents chest pain type, possibly categorized into different levels.
- trestbps: This column represents the resting blood pressure of an individual.
- chol: This represents the serum cholesterol level of an individual.
- fbs: This is the fasting blood sugar level, encoded as binary values (e.g., 0 for normal, 1 for high).
- restecg: This is the resting electrocardiographic results, indicating different states or conditions.
- thalach: This the maximum heart rate achieved during an exercise test.
- exang: This column represents exercise-induced angina, possibly encoded as binary values (e.g., 0 for no, 1 for yes).
- oldpeak: This represents a depression induced by exercise relative to rest in the ST segment of the electrocardiogram.
- slope: This represent the slope of the peak exercise ST segment.
- ca: This column represent the number of major vessels colored by fluoroscopy.
- thal: This is the Thallium stress test result, providing information about blood flow to the heart.

— target: This is a target variable, indicates an individual has a heart disease (1) or not (0).

Together, these columns suggest that the dataset is related to cardiovascular health and used for tasks such as predicting the presence or absence of heart disease based on the provided features

10.3 Results description:

Hospital server:

This figure depicts the primary interface of the hospital server application for managing and processing ambulance-related data intelligently.

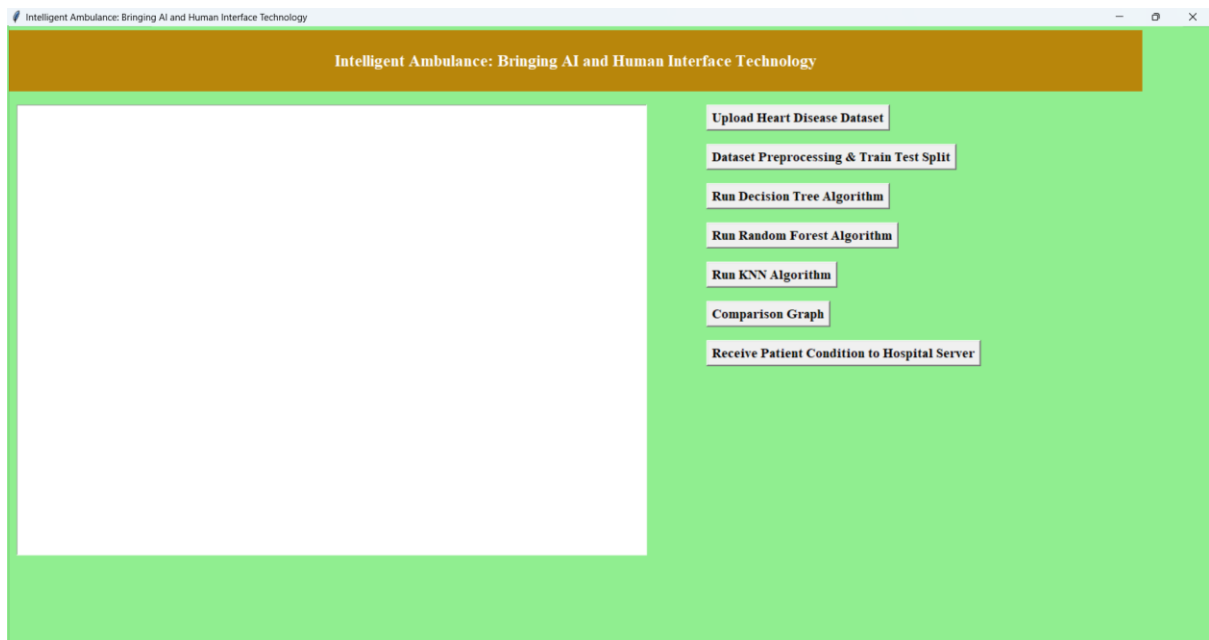


Figure 1: Main GUI application of proposed intelligence ambulance server side.

The figure 2 shows a screen or window within the GUI where users (possibly administrators or analysts) can select a dataset for analysis or model training.

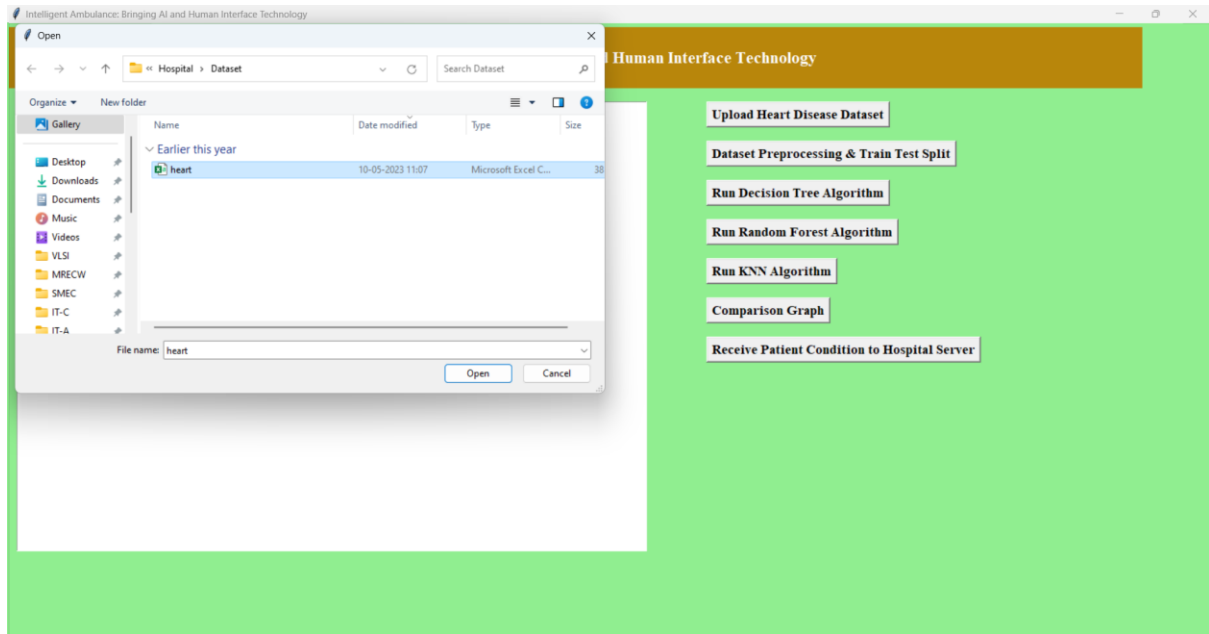


Figure 2: Selecting the dataset in the GUI application.

The figure 3 presents the content of the selected dataset along with a count plot illustrating the distribution of target columns, providing a quick overview of the data.

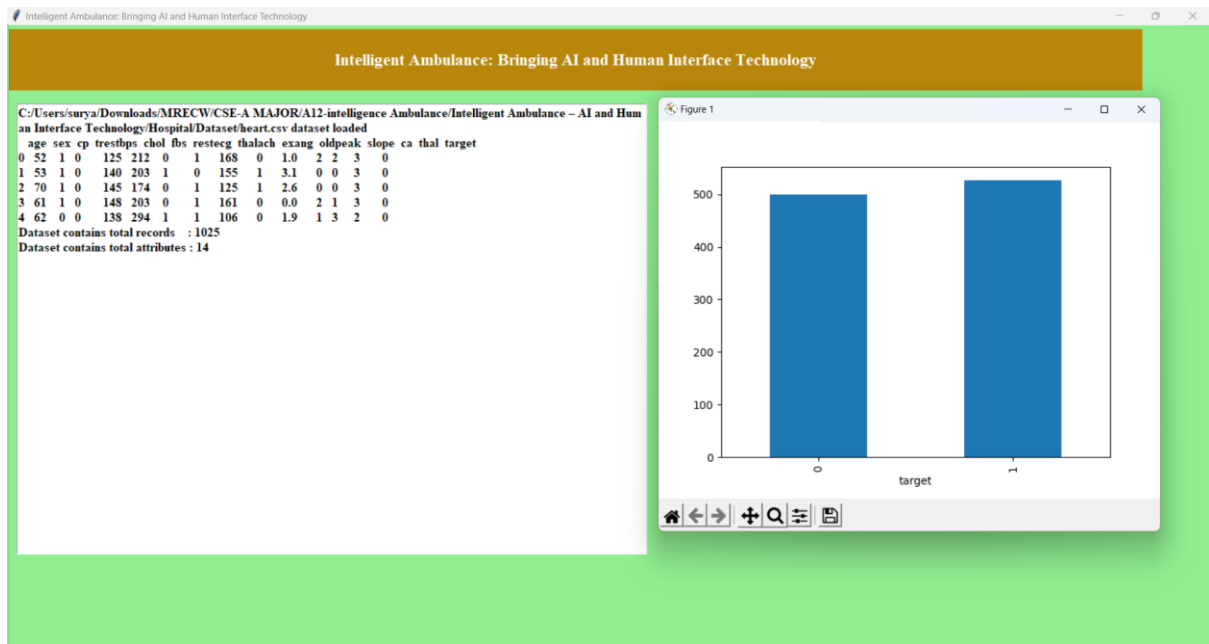


Figure 3: Displays the dataset and count plot of target columns.

The figure 4 displays the dataset after undergoing preprocessing steps, such as cleaning, transforming, or feature engineering, to prepare the data for analysis or model training.

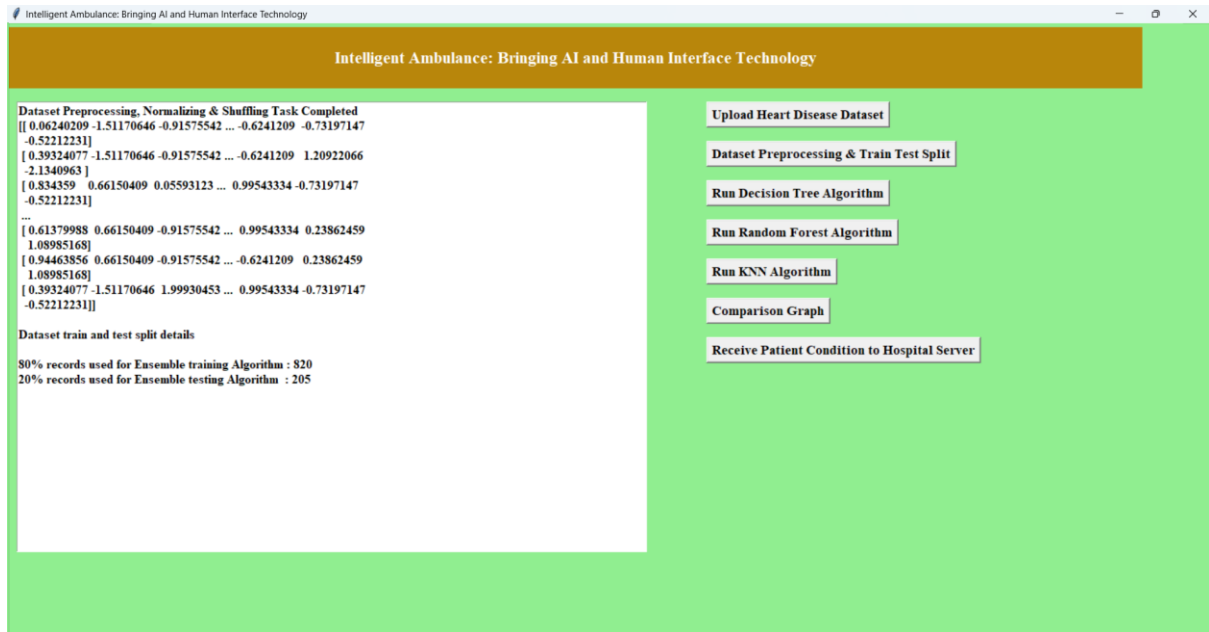


Figure 4: Presents the pre-processed data from the dataset.

This figure 5 shows the evaluation metrics, such as accuracy, precision, recall, and a confusion matrix, specifically for a Decision Tree model applied to the dataset.

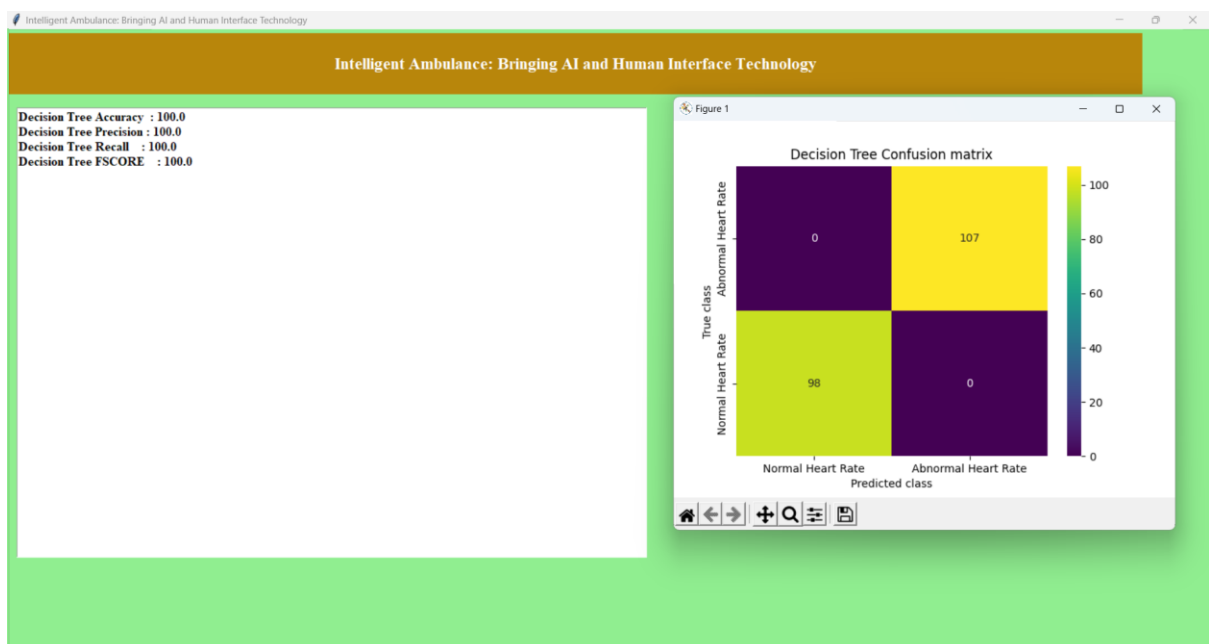


Figure 5: Displays the performance evaluation and confusion matrix of the decision tree model.

Similar to Figure 5, The figure 6 provides performance evaluation metrics and a confusion matrix, but for a Random Forest model.

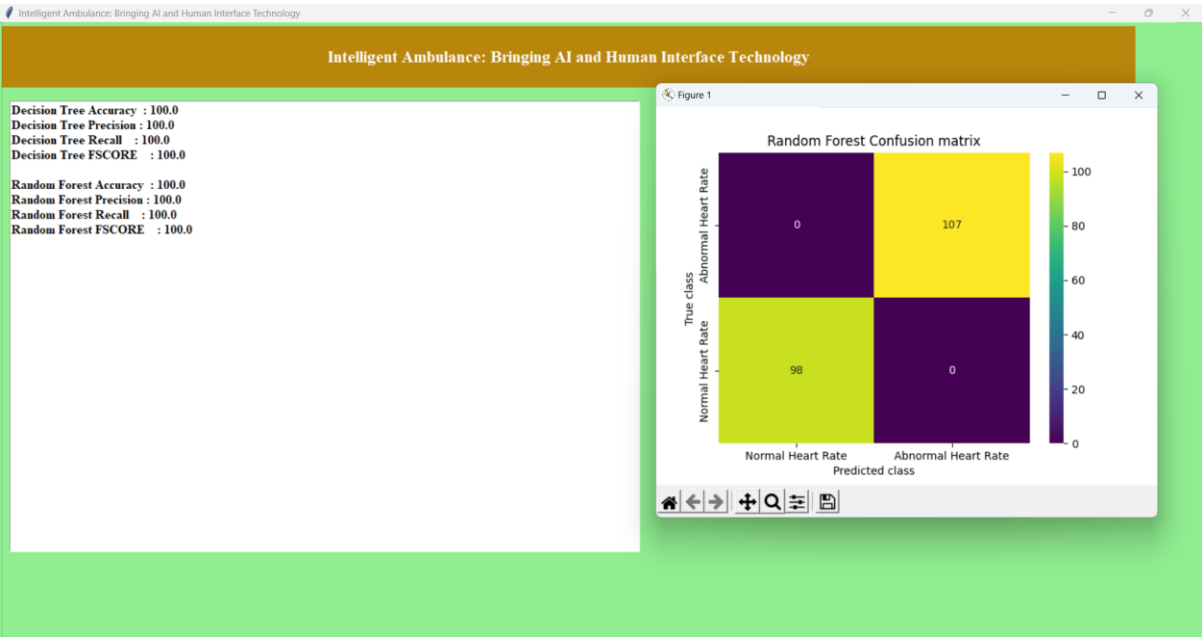


Figure 6: Displays the performance evaluation and confusion matrix of the random forest model.

The figure 7 shows performance evaluation metrics and a confusion matrix, but for a K-Nearest Neighbors (KNN) model.

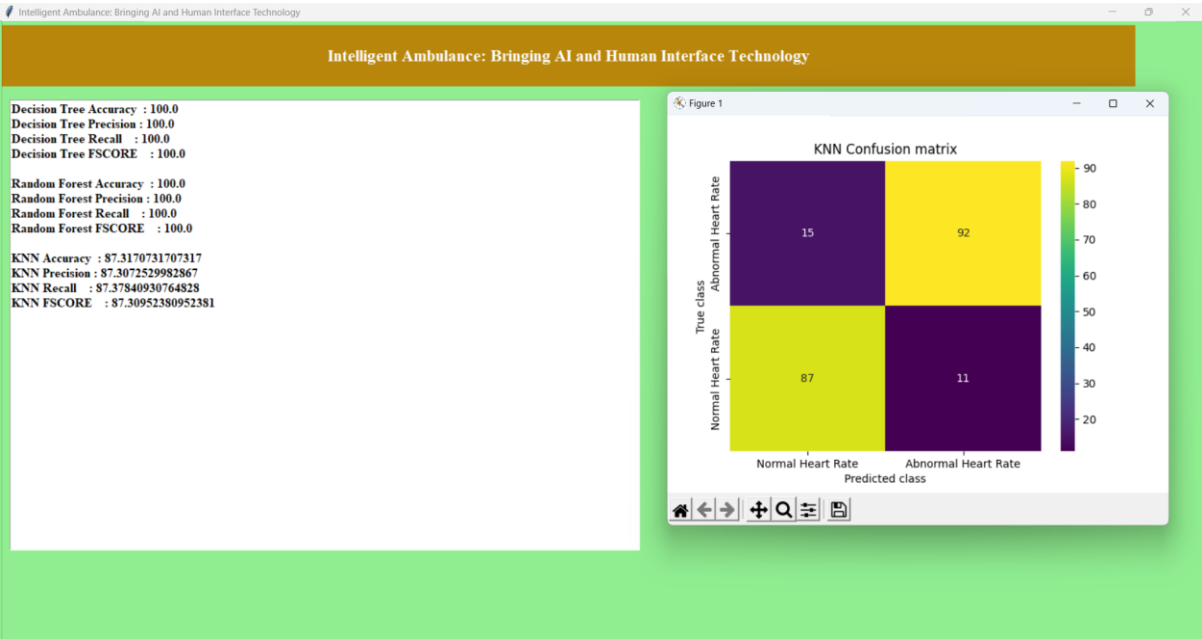


Figure 7: Displays the performance evaluation and confusion matrix of the KNN model.

The figure 8 provides a comparative analysis of performance metrics across the Decision Tree, Random Forest, and KNN models, helping users make informed decisions about model selection.



Figure 8: Displays the comparison of performance metrics in Decision Tree, RFC and KNN models.

The figure 9 illustrates the process of initiating or starting a cloud server through the graphical user interface, allowing seamless integration with cloud computing resources.



Figure 9: Displays the start of cloud server in the GUI console.

Ambulance reporting side:

The figure 10 represents the primary interface for the ambulance reporting side, providing a user-friendly platform for ambulance personnel to interact with the system.

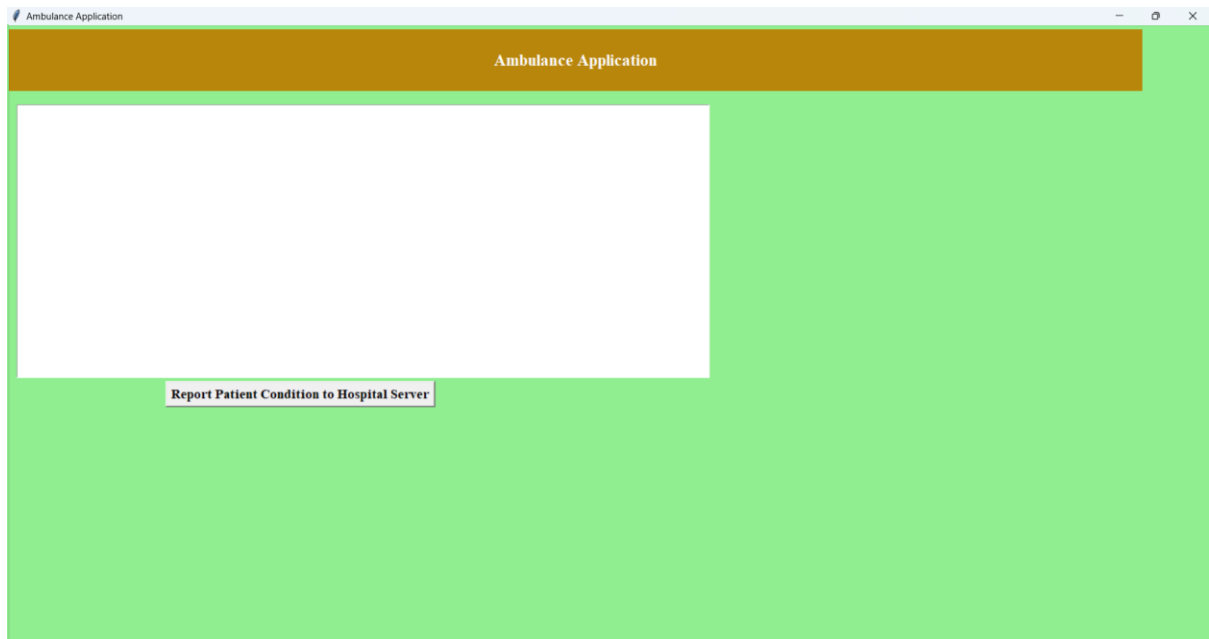


Figure 10: Main GUI application of ambulance side.

The figure 11 shows a screen where ambulance personnel can select a test dataset and initiate the reporting process to the hospital server, potentially sending relevant data for analysis.

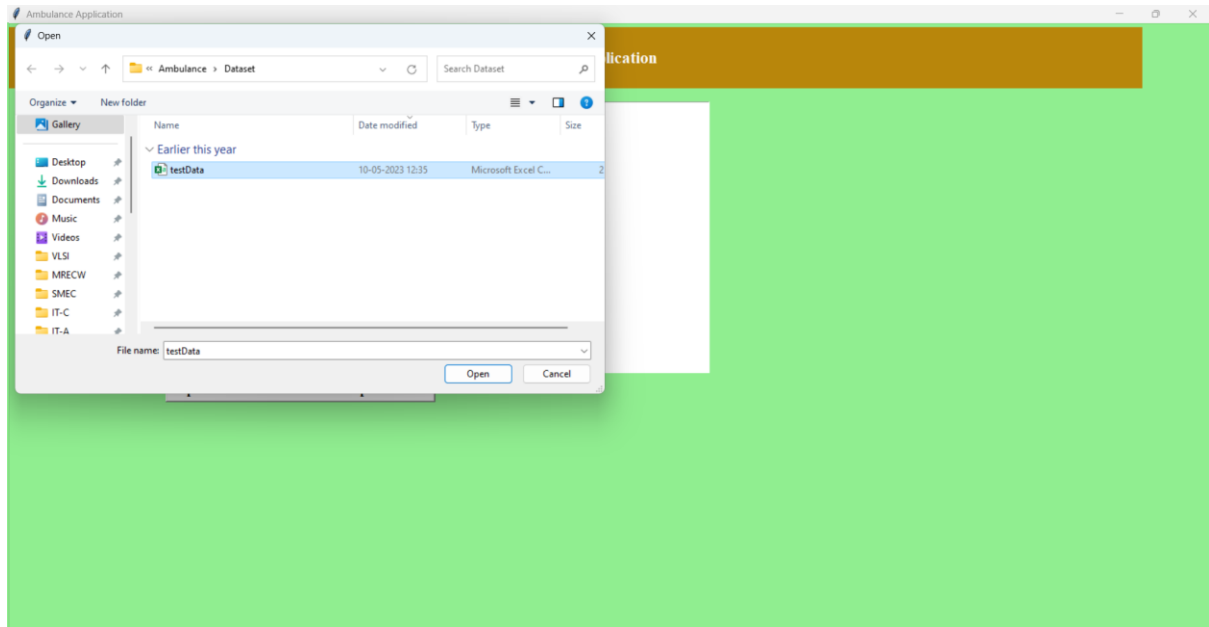


Figure 11: Selecting the test dataset and reporting to hospital server.

The figure 12 displays the outcomes or predictions generated by the server model based on the test data received from the ambulance side. It allows ambulance personnel to view and act upon the results.

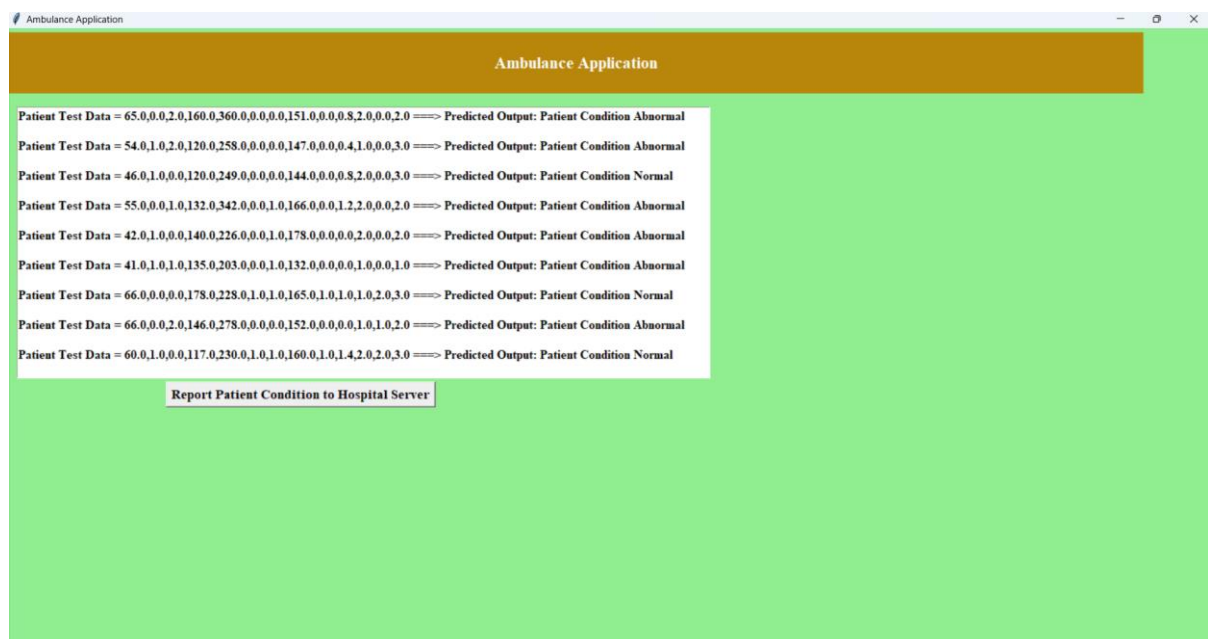


Figure 12: Represents the predication of test data by the server model.

CHAPTER 11

CONCLUSION AND FUTURE SCOPE

The Intelligence Ambulance project, integrating AI and human interaction technologies, has showcased the potential for significantly improving emergency medical services. The incorporation of artificial intelligence into ambulance systems, coupled with advanced human interaction technologies, has resulted in a more intelligent and responsive healthcare delivery system. The project has demonstrated successful outcomes in terms of faster response times, accurate patient assessment, and improved communication between healthcare providers and patients. The synergy between AI-driven decision support systems and human expertise has the potential to transform emergency medical care, enhancing both efficiency and patient outcomes.

Future Scope:

Advanced AI Diagnostics: Expanding the capabilities of AI within the ambulance for advanced diagnostics as a key focus. Integrating AI algorithms that can quickly analyze medical data, such as ECG readings or vital signs, can assist paramedics in making more informed decisions about patient care during transit.

Telemedicine Integration: Enhancing the connectivity of the Intelligence Ambulance to enable seamless integration with telemedicine platforms can enable real-time consultation with healthcare professionals. This can be particularly valuable in critical situations where expert advice is needed to guide on-the-spot medical decisions.

IoT for Health Monitoring: Integrating Internet of Things (IoT) devices for continuous health monitoring during transportation can provide a more comprehensive view of the patient's condition. Wearable sensors or connected medical devices can transmit real-time data to the ambulance, enabling proactive intervention.

Augmented Reality (AR) for Assistance: Incorporating augmented reality (AR) technologies can assist paramedics in various tasks, such as locating veins for intravenous access or providing step-by-step procedural guidance. AR can enhance the precision and speed of medical interventions in the field.

Predictive Analytics for Resource Allocation: Implementing predictive analytics to forecast emergency medical service demands can optimize resource allocation. AI algorithms can analyze

historical data to predict high-demand areas or times, allowing for better deployment of ambulance services.

Enhanced Communication Systems: Improving communication systems within the ambulance, including secure and high-bandwidth connections, can facilitate real-time data exchange between the ambulance and healthcare facilities. This can ensure that the receiving hospital is well-prepared for the incoming patient.

Human-Machine Collaboration Training: Training programs for paramedics and emergency responders can be developed to enhance their skills in collaborating with AI systems. This includes understanding AI recommendations, interpreting AI-generated insights, and making informed decisions based on the combined input from AI and human expertise.

Community Outreach and Education: Establishing community outreach programs to educate the public about the capabilities of the Intelligence Ambulance and how to use emergency services efficiently can contribute to improved overall emergency response outcomes.

Regulatory Compliance and Ethical Considerations: As AI technologies in healthcare evolve, ensuring compliance with regulatory standards and addressing ethical considerations, such as patient privacy and data security, will be crucial. Future work can focus on developing frameworks that adhere to healthcare regulations and ethical guidelines.

In summary, the Intelligence Ambulance project presents a transformative approach to emergency medical services, and future developments can further enhance its capabilities, ensuring that the integration of AI and human interaction technologies continues to contribute to more effective and responsive healthcare delivery in emergency situations.

REFERENCES

- [1] Akash Bansode, Sanket Thakare, Sarthak Pawar, Subodh Wavhal and D.S. Rakshe, Smart Ambulance Management Application Using Cloud, IJARIIIE-ISSN(O)-2395-4396, Vol-8, Issue-3 2022.
- [2] Divya Ganesh, Gayathri Seshadri, Sumathi Sokkanarayanan, Panjavarnam Bose, Sharanya Rajan and Mithileysh Sathiyarayanan, “Automatic Health Machine for COVID-19 and Other Emergencies,” in IEEE-2021.
- [3] Gargi Beri, Pankaj Ganjare, Amruta Gate, Ashwin Channawar, Vijay Gaikwad, “Intelligent Ambulance with Traffic Control”, Upper Indira Nagar, Bibvewadi, Pune, ISSN: 2454-5031, Volume 2 - Issue 5, May 2016. Design & Development of Intelligent Ambulance Concept – AI and Human Interface Technology Section A-Research paper 186 Eur. Chem. Bull. 2023,12(Special Issue 9), 177-188
- [4] Ms. Aisha Meethian, Althaf B.K., Athinan Saeed, Ligin Abraham, Mohammed Samran, “IoT Based Traffic Control System with Patient Health Monitoring For Ambulance”, ISSN:2395-5252, Volume 4, Issue 8, August 2022.
- [5] Himadri Nath Saha , Neha Firdaush Raun, Maitrayee Saha, “Monitoring Patient’s Health with Smart Ambulance system using Internet of Things (IOTs)”, IEEE 2017.
- [6] Timothy Malche, Sumegh Tharewal, Pradeep Kumar Tiwari and Mohammad Aman Ullah, “Artificial Intelligence of Things- (AIoT) Based Patient Activity Tracking System for Remote Patient Monitoring”, 2022.
- [7] Erik Alonso, Unai Irusta, Elisabete Aramendi and Mohamud R. Daya, “A Machine Learning Framework for Pulse Detection During Out-of-Hospital Cardiac Arrest”, 2020.
- [8] Yuanyuan Pan, Minghuan Fu, Biao Cheng, Xuefei Tao and Jing Guo, “Enhanced Deep Learning Assisted Convolutional Neural Network for Heart Disease Prediction on the Internet of Medical Things Platform”, 2020.
- [9] M. Sheetal Singh, Prakash Choudhary, “Stroke Prediction using Artificial Intelligence”, 2017.
- [10] Santhana Krishnan J. and Geetha S., “Prediction of Heart Disease using Machine Learning Algorithms” ICICT, 2019.

- [11] Akash Bansode, Sanket Thakare, Sarthak Pawar, Subodh Wavhal and D.S. Rakshe, “Smart Ambulance Management Application Using Cloud”, IJARIE, ISSN(O) - 2395- 4396, Vol-8 Issue- 3, 2022.
- [12] Aiswarya G, Anjali U K, Amal Mathew, Alen Benny, Jamshid C T, Prof. Tinimol Andrews, “Android based Remote Health Monitoring System”, IJERT, Volume 9, Issue 13,2021.
- [13] M. Dhinakaran , Khongdet Phasinam , Joel Alanya Beltran , Kingshuk Srivastava , D. Vijendra Babu , and Sitesh Kumar Singh, A System of Remote Patients’ Monitoring and Alerting Using the Machine Learning Technique, Journal of Food Quality, 8 February 2022. [14] Sakshi Dhuria , Mohd. Yusuf Khan , Rupal Mishra , Shivam Narsingh , Satya Prakash Singh, “Ambulance Tracking with Patient Health Monitoring by the use of GPS and GSM”,2021.
- [14] Gowthami.P, “an Android based Patient monitoring System”, vol. 3, IJIRAE, 2016.
- [15] P.Ponsudha ,Haritha.K ,Gayathri.D ,HarshithaShree.P ,S wetha.C, “Efficient ambulance service with real time ambulance monitoring syatem” International Journal of Applied Engineering Research, 2019.
- [16] M Sanjay Karanth, Bindhu kumar K T, Gururaj Reddy J, Manoj K, Veda. B, “Smart ambulance with patient monitoring”, 2015.
- [17] Mery Subito1, Alamsyah , and Ardi Amir, “Web-Based Wireless Monitoring System on Patient’s Vital Sign”, 2019.
- [18] Tia Gao, Logan K. Hauenstein, Alex Alm, David Crawford, Cassius K. Sims, Azmat Husain, and David M, “Vital Signs Monitoring and Patient Tracking Over a Wireless Network”, 2016
- [20] Amogh Powar , Seema Shilvant , Varsha Pawar, Pratiksha Shetgaonkar, Shailendra Aswale, “Data Mining & Artificial Intelligence Techniques for Prediction of Heart attack”, 2019.

Project Details				
Academic Year		2023-2024		
Title of the Project		REVOLUTIONIZING EMERGENCY HEALTHCARE IN DEVELOPING INDIA: AN AI-INTEGRATED AMBULANCE SYSTEM FOR TIMELY INTERVENTION IN CRITICAL CONDITIONS		
Name of the Students and Hall Ticket No.		P. SAIRAM (20RA1A0572) V. SAI BHARATH (20RA1A0566) P. SAI ROHIT (20RA1A0568)		
Name of the Guide		DR. S. KAVITHA		
Project PO Mapping				
Name of Course from which Principles are applied in this Project	Related Course Outcome Number	Description of the application	Page Number	Attained
Python Programming Software Engineering (C313)	C313.1	Students described the basis for their problem statement.	1- 4	PO2
Machine Learning, Python Programming, Data Mining (C322, C411)	C322.2, C411.2	Students explained about An AI-Integrated Ambulance system for Timely Intervention In Critical Conditions	5-6	PO1
Software Engineering, Python Programming (C313)	C313.3	Students identified the existing system and its Drawbacks and proposed a Solution to it.	7-15	P02,P03
Design Patterns, Software Engineering, (C313, C222)	C313.2, C222.3,	Students explain the flow of the project using UML diagrams.	16-21	P03,P05, PO9, PSO3
Software Engineering (C313)	C313.1	Students identified the hardware and software required for the project	35	PO5

Python Programming, Data Mining(C411)	C411.2	Students explained about python programming language and decided the packages for the solution of the problem.	22-33	PO2, PO3,PO4
Python Programming		Students Developed code for the problem Statement.	40-53	PO3,PO4,PO5
Future Scope		Students explained about how they would like to take their project further and develop it as their future scope	63	PO12,PS02
Bibliography		Listed the references from which the literature was collected	65-66	PO8,PO12
ENG		Prepared the thesis and intermediate progress reports and explained to the review panel. Also, continuously caught up with guide and explained the progress.		PO9,PO10

SIGNATURE OF STUDENT

SIGNATURE OF INTERNAL GUIDE