Investigation of An Early Prediction System of Cardiac Arrest Using Machine Learning Techniques

MUHAMMAD AFNAN BIN MOHAMMAD NASIR

BACHELOR OF ELECTRICAL ENGINEERING (ELECTRONICS) WITH HONOURS

UNIVERSITI MALAYSIA PAHANG

UNIVERSITI MALAYSIA PAHANG

DECLARATION OF THESIS AND COPYRIGHT

Author's Full Name : MUHAMMD AFNAN BIN MOHAMMAD NASIR

Date of Birth : 8 DISEMBER 1997

Title : INVESTIGATION OF AN EARLY PREDICTION SYSTEM OF CARDIAC ARREST USING MACHINE LEARNING TECHNIQUES

Academic Session : ____

I declare that this thesis is classified as:

☐	CONFIDENTIAL	(Contains confidential information under the Official Secret Act 1997)*
☐	RESTRICTED	(Contains restricted information as specified by the organization where research was done)*
☑	OPEN ACCESS	I agree that my thesis to be published as online open access (Full Text)

I acknowledge that Universiti Malaysia Pahang reserves the following rights:

1. The Thesis is the Property of Universiti Malaysia Pahang
2. The Library of Universiti Malaysia Pahang has the right to make copies of the thesis for the purpose of research only.
3. The Library has the right to make copies of the thesis for academic exchange.

Certified by:

(Student's Signature)

971208-03-7171
New IC/Passport Number
Date: 17/6/2022

(Supervisor's Signature)

DR. AHMAD AFIF BIN MOHD FAUDZI
Date: 27 June 2022

NOTE: * If the thesis is CONFIDENTIAL or RESTRICTED, please attach a thesis declaration letter.

THESIS DECLARATION LETTER (OPTIONAL)

Librarian,
Perpustakaan Universiti Malaysia Pahang,
Universiti Malaysia Pahang,
Lebuhraya Tun Razak,
26300, Gambang, Kuantan.

Dear Sir,

CLASSIFICATION OF THESIS AS RESTRICTED

Please be informed that the following thesis is classified as RESTRICTED for a period of three (3) years from the date of this letter. The reasons for this classification are as listed below.

Author's Name
Thesis Title

Reasons (i)

(ii)

(iii)

Thank you.

Yours faithfully,

____
(Supervisor's Signature)

Date:

Stamp:

Note: This letter should be written by the supervisor, addressed to the Librarian, *Perpustakaan Universiti Malaysia Pahang* with its copy attached to the thesis.

SUPERVISOR'S DECLARATION

I hereby declare that I have checked this thesis and in my opinion, this thesis is adequate in terms of scope and quality for the award of Bachelor of Electrical Engineering (Electronics) with Honours

(Supervisor's Signature)

Full Name : DR. AHMAD AFIF BIN MOHD FAUDZI

Position : Senior lecturer

Date : 27 June 2022

(Co-supervisor's Signature)

Full Name :

Position :

Date :

Universiti Malaysia PAHANG
Engineering • Technology • Creativity

STUDENT'S DECLARATION

I hereby declare that the work in this thesis is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at Universiti Malaysia Pahang or any other institutions.

_____ ________

(Student's Signature)

Full Name : MUHAMMAD AFNAN BIN MOHAMMAD NASIR

ID Number : EA18069

Date : 17/6/2022

Investigation of An Early Prediction System of Cardiac Arrest Using Machine Learning Techniques

MUHAMMAD AFNAN BIN MOHAMMAD NASIR

Thesis submitted in fulfillment of the requirements
for the award of the
Bachelor of Electrical Engineering (Electronics) with Honours

College of Engineering
UNIVERSITI MALAYSIA PAHANG

JUNE 2022

ACKNOWLEDGEMENTS

All praise to the almighty God with His blesses and chances for me as I can finish this project and thesis writing in order for me to finish my Bachelor of Electrical Engineering (Electronics) with Honours, Alhamdulillah.

In this opportunity, I would like to give a big gratitude and thank to my beloved parent. A thousand of thanks for both of you that support me without hesitate until I can finish my project. Their moral support gives me hope and push me to exceed my boundary in making this thesis happen.

Also, I would like to thank and give my gratefulness for my supervisor, Dr. Ahmad Afif Bin Mohd Faudzi who taught me and lead me until I can successfully finish the project given in the meantime. His guidelines based on his experience in researching skills and knowledge help me a lot in doing this project and writing this thesis.

My next gratitude will be for my Academic Advisor, Ts.Dr. Asrul Bin Adam as he helps me a lot during my ups and downs while finishing this project. His supportive character always calms me and help me in making good decisions throughout this project development. Not to forget all staff in College in Engineering, specifically the Department of Electrical Engineering that gives their knowledge and experience to makes me understands more about my project.

To my friends who is together with me throughout this final year project, our sleepless night, our joy and cry. From the bottom of my heart, I would like to thank all of you for the support and encourage to me, lets pray for our best in future.

Thank you.

ABSTRACT

The increase in popularity for wearable technologies has opened the door for an Internet of Things (IoT) solution to healthcare. One of the most prevalent healthcare problems today is the poor survival rate of out-of-hospital sudden cardiac arrests. Not only that, most of the conventional device are also wired and unfriendly device. Most of the people nowadays have no alert and better awareness about their health condition. The objective of this study is to present a multisensory system using IoT that can collect physical activity heart rates and body temperatures that can alert about their health condition. For this study, we implemented an embedded sensory system with a Low Energy Bluetooth communication module to discreetly collect electrocardiogram and body temperature data using a smartphone in a common environment. To identify approaching heart illness using Machine learning techniques, a preliminary design of a cloud-based heart disease prediction system was developed. An effective machine learning approach created from a separate examination of many machine learning algorithms in WEKA should be applied for the correct identification of heart disease. Random Forest algorithm be used in this study which is got the best performance with 83% accuracy then the other algorithms in WEKA. This algorithm was applied in the Python using Google Colab to make prediction of the sudden cardiac arrest. As the result, to make a prediction user need to set their data health in the Python using Random Forest algorithm to detect either they have a heart disease or not.

ABSTRAK

Peningkatan populariti untuk teknologi boleh pakai telah membuka pintu untuk penyelesaian Internet Perkara (IoT) kepada penjagaan kesihatan. Salah satu masalah penjagaan kesihatan yang paling lazim hari ini ialah kadar kelangsungan hidup yang lemah akibat serangan jantung mengejut di luar hospital. Bukan itu sahaja, kebanyakan peranti konvensional juga adalah peranti berwayar dan tidak mesra. Kebanyakan orang pada masa kini tidak mempunyai kesedaran dan kesedaran yang lebih baik tentang keadaan kesihatan mereka. Objektif kajian ini adalah untuk mempersembahkan sistem multisensori menggunakan IoT yang boleh mengumpul degupan jantung aktiviti fizikal dan suhu badan yang boleh memberi amaran tentang keadaan kesihatan mereka. Untuk kajian ini, kami melaksanakan sistem deria terbenam dengan modul komunikasi Bluetooth Tenaga Rendah untuk mengumpul data elektrokardiogram dan suhu badan menggunakan telefon pintar dalam persekitaran biasa. Untuk mengenal pasti penyakit jantung yang menghampiri menggunakan teknik pembelajaran Mesin, reka bentuk awal sistem ramalan penyakit jantung berasaskan awan telah dibangunkan. Pendekatan pembelajaran mesin yang berkesan yang dicipta daripada pemeriksaan berasingan bagi banyak algoritma pembelajaran mesin dalam WEKA harus digunakan untuk mengenal pasti penyakit jantung yang betul. Algoritma Random Forest digunakan dalam kajian ini yang mendapat prestasi terbaik dengan ketepatan 83% berbanding algoritma lain dalam WEKA. Algoritma ini digunakan dalam Python menggunakan Google Colab untuk membuat ramalan serangan jantung secara tiba-tiba. Hasilnya, untuk membuat ramalan pengguna perlu menetapkan kesihatan data mereka dalam Python menggunakan algoritma Random Forest untuk mengesan sama ada mereka mempunyai penyakit jantung atau tidak.

TABLE OF CONTENT

DECLARATION		
TITLE PAGE		
ACKNOWLEDGEMENTS		**ii**
ABSTRACT		**iii**
ABSTRAK		**iv**
TABLE OF CONTENT		**v**
LIST OF TABLES		**viii**
LIST OF FIGURES		**ix**
LIST OF SYMBOLS		**x**
LIST OF ABBREVIATIONS		**xi**
CHAPTER 1 INTRODUCTION		**1**
1.1	Overview	1
1.2	Project Background	2
1.3	Problem Statement	3
1.4	Objective	4
1.5	Scope and Limitation of project	5
1.6	Research Challenges	5
CHAPTER 2 LITERATURE REVIEW		**6**
2.1	Introduction	6
2.2	Cardiac Arrest	6
2.3	Machine Learning Algorithm	7

2.4 Wearable & Mobile Healthcare System 8

2.5 Heart Disease Prediction 8

CHAPTER 3 METHODOLOGY 9

3.1 Introduction 9

3.2 Block Diagram 9

3.3 Flow Chart 10

3.4 System Architecture Proposal 11

3.5 Device Hardware 12

3.5.1 Circuit Diagram 12

3.5.2 Arduino Pro Mini 13

3.5.3 ESP8266 Wi-fi Module 14

3.5.4 MAX30102 Sensor 15

3.5.5 Organic Light Emitting Diode (OLED) 128”32 Screen display 16

3.6 Machine Learning Algorithms 17

3.6.1 Naïve Bayes 17

3.6.2 Artificial Neural Networks (ANN) 18

3.6.3 Support Vector Machine (SVM) 18

3.6.4 Random Forest 19

3.6.5 Simple Logistic Regression 19

3.7 Data Source 20

3.8 Attribute Documentation 21

CHAPTER 4 RESULTS AND DISCUSSION 22

4.1 Hardware Design 22

4.2 Machine Learning Algorithm Performance Analysis 23

4.3 Algorithm Analysis Performance 25

4.4 Heart Disease Prediction Using Python 26

CHAPTER 5 CONCLUSION 27

5.1 Conclusion 27

5.2 Summary 28

5.3 Future Recommendation 29

REFERENCES 30

APPENDIX A LIST OF DATA SET 33

APPENDIX B CODING FOR WIFI MODULE 35

APPENDIX C CODING FOR ARDUINO 40

APPENDIX D CODING PYTHON FOR PREDICTION 46

LIST OF TABLES

Table 3.1 Arduino Pro Mini Key Features 14

Table 3.2 MAX30102 Sensor Pinout 16

Table 3.3 Input Attributes used for model formation and validation 20

Table 4.1 Performance analysis of five machine learning algorithms (in %). 24

LIST OF FIGURES

Figure 3.1 Block diagram of wireless early prediction system of cardiac arrest	10
Figure 3.2 Flow chart of the project system.	11
Figure 3.3 Flow of the project	12
Figure 3.4 Circuit Diagram of The Project	13
Figure 3.5 Arduino pro mini	13
Figure 3.6 Arduino pro mini pinot	14
Figure 3.7 ESP8266 Wi-fi Module	14
Figure 3.8 ESP8266 Pinout	15
Figure 3.9 MAX30102 Sensor	15
Figure 3.10 OLED 128“32 Screen display	16
Figure 3.11 Artificial neural network	18
Figure 3.12 Support vector machine operation	19
Figure 4.1 Circuit Diagram for Hardware Device	22
Figure 4.2 Result accuracy of the Analysis Algorithm	24
Figure 4.3 Weka display screen showing different parameters for Random Forest.	25
Figure 4.4 Result from Python	26

LIST OF SYMBOLS

%	Percentage
^{o}C	Degree Celcius

LIST OF ABBREVIATIONS

CVD	Cardiovascular disease
WHO	World health organization
IoT	Internet of things
IP	Internet protocol
ANN	Artificial Neural Network
BAS	Body Area Sensor
WHMS	Wireless Health Monitoring System
SEM	Structural Equation Modeling
CCHS	Canadian Community Health Survey
FCM	Fuzzy Cognitive Maps
WEKA	Waikato Environment for Knowledge Analysis
PPG	Photoplethysmography

CHAPTER 1

INTRODUCTION

1.1 Overview

The heart is the most essential component of the circulatory system, which also includes the lungs. It is a muscular organ that pumps blood throughout the body. A process of blood vessels, such as veins, arteries, and capillaries, is also part of the cardiovascular system. These blood arteries supply blood throughout the body. Problems in normal heart blood flow cause a variety of cardiac disorders known as cardiovascular diseases (CVD). The main cause of dying globally is heart failure. From World Health Organization (WHO) report, heart attacks and strokes cause There are 17.5 million deaths globally. Over 75 percent of cardiovascular disease kills occur in middle- and low-income country. In addition, strokes and heart problems account about 80% of CVD-related mortality [1]. So, early diagnosis of cardiac irregularities as well as solutions for the detection of heart illnesses will save a lot of people's lives and assist doctors in designing an efficient treatment plan, thus lowering the death cardiovascular disease mortality rate. Because of the improvement of modern healthcare systems, a large amount of patient information is now available for use in creating prediction Cardiovascular models illnesses. Data mining, also known as machine learning, is a way of discovering new knowledge by evaluating large amounts of data from many perspectives. " Data mining is the non-trivial retrieval of implicit, previously unknown, and possibly good information from data" [2]. Medical institutions create a massive quantity of data on illness diagnosis, patients, and so on. Data mining employs a variety of approaches to uncover data with patterns and insights or relationships. As a result, in this research, a machine learning approach for development of a heart failure prediction system is given, and it is verified on two freely accessible prediction of heart disease datasets.

Another feature of this work is the introduction of a method for monitoring cardiac patients based on the Internet of Things (IoT) technique and employing several sensor systems for physiological data and an Arduino microcontroller. Sensor networks usually use Internet of Things (IoT) technology to merge, analyse, and transmit data through one node to the next. IoT is a relatively new and fast increasing technology in which various sensors/data collectors may detect, share, and interact across a public networks, Internet Protocol (IP), or private networks. After a certain period of time, the sensors collect data, evaluate it, and apply it to trigger the next action, resulting in a cloud-based intelligent system for monitoring, making plans, and decision making. IoT device, like embedded technology, give an information exchange between nodes or the Web, and it is anticipated that 8 to 50 billion gadgets will be associated by 2020.[3].

1.2 Project Background

Many research findings emphasised the significance and advantages of employing a regression heart disease detection and prediction system. The use use of ai technology in illness diagnosis systems, particularly identification of heart problems systems, enhances the efficiency of other commonly used models, such as models provided by the in CVD diagnosis and prediction, the American College of Cardiology/American Heart Association (ACC/AHA) [4]. Zhao, Wang, and Nakahira examined the possibility and associated issues of providing better In 2011, they also provided human patient care system services provided a study IoT medical technology direction [5].

This study intends to create a Decision Support System for heart disease identification that employs the data mining approach with the highest exactness and performance between Nave Bayes, Support Vector Machine, Simple Logistic Regression, Random Forest, and Artificial Neural Network (ANN) are some of the techniques used among others. It is feasible to detect the chances of getting heart disease by utilising many arterial system factors for example age, blood pressure, ECG readings, gender, and blood sugar [6]. A examination of several machine learning algorithms was shown in order to derive the algorithm that have the highest performance in the diagnosis and detecting of heart disease. This algorithm also use medical factors like age of the user, their blood pressure, gender, ECG readings, blood sugar, and so on as input and outputs the chances of impending heart attack. This suggested system consists of the strategy and build of a web-based Mobile application that uses an accurate machine to identify heart illness approach. It can be a highly important tool in diagnosing cardiac problems for doctors, patients, and medical students. After moving from the hospital to the patient's home, 24 hour monitoring of the patient's health is required for the diagnosis of deadly physiological abnormalities and symptoms such as heart attack. The patient may use this application to input the current parameters of heart disease from any location on the application's architecture and examine the risk level of developing heart disease. Every non-real-time variable, like blood sugar, serum cholesterol, and result of ECG, will be also certain metrics are provided in the doctor's recommended report, such as chest pain Kind of exercise-induced angina should be self-measured by the user on a routine basis and manually entered into the interactive platform. If the programme detects a highly dangerous condition, the patient may interact with any registered doctor with a heart illness can be discovered by make a medical check-up.

Critical care is constantly monitored cardiac patients is available in hospitals, although patients are generally released from the hospital without direct observation. For at least a week, these patients must have their health condition regularly checked to decrease the possibility of unexpected effects. As a result, another objective of this research is the empirical design and implementation of a prototype of a continuous real-time heart health monitoring system utilising Arduino-based sensors that can be commercially produced and attached to the person's body. Using the Wi-Fi module, sensor data would be sent to the server and preserved in the server database. The sensor data may be updated on the server every 10 seconds, and users with smartphones can use this application from anywhere to see their current health status. If any of the blood pressure, temperature, or SpO2 sensors exceed the threshold value, users will immediately receive an notification message on the programme as well as on his phone. Furthermore, the system allows family members and caretakers to access the patient's real-time data.

1.3 Problem Statement

From year to year, people who had respiratory distress are getting increase due to environmental condition. Examples of respiratory distress are cardiac arrest, short of breath and now is Covid-19 pandemic. The demand of heath monitoring device to monitoring the vital sign among the hospital and home monitoring is getting rise. But the device is not very privilege on alert vital sign drop. Here, where the problem that produce idea of wireless early prediction system of cardiac arrest will invent are:

- Conventional device are wired device and unfriendly to user.

As we know, at hospital, the pulse oximeter device is wired device that attach along with the monitor during the value investigation. The photoplethysmogram sensor must connect to the monitor port to power the sensor for reading purposes. This makes the monitoring system device been unfriendly to be used where the user needs to bring along with the monitor everywhere the person is move.

- No better awareness of their health status and early medical warnings.

The person who is under asthma attack phase sometimes does not realize when their oxygen level was dropping, and heart rate was higher and need medication such as take their inhaler to overcome the shortness of breath. This often happens to children and elderly when they in deep sleep.

- No alert info when impending cardiac arrest

Most of device that been in the market such as smartwatch does not have alert system when oxygen level and pulse rate was reduced in certain rate which from normal range to range that need had attention.

1.4 Objective

The purpose of this proposed project of wireless early prediction system of cardiac arrest is to overcome circumstances on monitoring vital in real - time condition. The objectives of this project are:

i. To invent friendly and wearable health monitoring using IoT.

ii. To measure heart rate, temperature and oxygen level of body using Arduino to get data set for provide early medical warnings.

iii. To present a multisensory system using a smart IoT system that can collect Body Area Sensor (BAS) data using machine learning algorithm for early warning cardiac arrest prediction.

1.5 Scope and Limitation of project

In this invention of wristband pulse oximetry monitoring system for detect the cardiac arrest, purposely to alert user and guardian or parent on the vital sign during respiratory distress issue. Data that produce are collected from the human body directly (saturation of oxygen in blood, pulse rate and body temperature) using the MAX30102 biometric sensor and microcontroller to analyses the data collected then the output from the data analysis will send to application using cloud server (Blynk).

This propose project is focus on heart attack or cardiac arrest issues. This propose project is recommended use for self- monitoring and for home-monitoring system device. But then, this propose project also has it limitation where the device only can take data for one person per time. Then, measurement of data value will be store under one name at cloud server application. Plus, device and cloud server application must connect with Wi-Fi to store the data measurement value for further use. Also, the measurement value will slightly not accurate when the user not wearing improper way and have difference surface of skin.

1.6 Research Challenges

Once it comes to IoT systems based on eHealth, we want to design a system that is different and stands out. In doing so, we encountered a number of research hurdles, which are stated below:

i. Establishing between the Iot system and the mobile app through a low-energy communication path

ii. Data gathering, processing, and visualizing in real time

iii. Using an integrated sensor system to predict cardiac problems.

iv. Sending a message of concern to a caregiver or emergency contact.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

Given that the topic under investigation is a highly popular area of research that is expanding on a daily basis as technology improves, there has been a variety of work done to something we might relate. Many people are now conducting eHealth research, and many businesses have benefited from this work by developing system connects patients with doctors all over the world. Many research have proposed various ways for utilising machine learning to predict cardiac disease and stress. However, the majority of them are unable to anticipate in real time and do not give a cohesive framework.

2.2 Cardiac Arrest

Heart disease is a broad phrase that encompasses a variety of illnesses, accidents, and ailments affecting the heart and blood arteries. The symptoms of heart illness varies depending on the kind of heart ailment [8]. Congenital cardiac disease refers to the creation and operation of the heart as a result of premature heart maturation during birth [9]. Congestive heart disease arises when the heart unable to pump enough blood to the body's various organs. Coronary heart failure, also known as ischemic heart dysfunction, is the most common kind of heart problem [10]. Ischaemic heart failure is a disorder characterised by heart damage caused by decreased blood flow, which increases fatty deposits on blood vessel cladding that provide oxygen to the cardiac muscles, adding to their stiffness [11].

2.3 Machine Learning Algorithm

Several study findings emphasised the significance and benefits of using a machine learning-based heart disease detection and prediction system. The use of artificial intelligence in disease detection systems, particularly cardiac disease detection systems, enhances the performance of other commonly used models, such as models provided by the American College of Cardiology/American Heart Association (ACC/AHA) in CVD detection and prediction [4]. Zhao, Wang, and Nakahira examined the possibility and associated issues of providing better services of a human health management system in 2011, and they also provided a study direction of medical technology on IoT [5]. They investigated a wide range of health-related sensors and technologies. They found certain difficulties that needs to be resolved. Chiuchisan and Geman designed the home monitoring and decision assistance system in 2014 [12]. This approach helped in the home surveillance, diagnosis, medical prescriptions, medical care, rehabilitation, and personal growth of Parkinson's disease patients. The Wireless Health Monitoring System (WHMS) has received a lot of attention during the last decade and get interest from the scientific community and industry. Several Machine Learning algorithms and classifier performances, such as the weighted associative Classifier quality, such as that of the weighted associative classifier, has been shown to improve in the identification of cardiac defects [13].

There are several data mining tools available for implementing machine learning algorithms. WEKA, TANAGRA, RapidMiner, MATLAB, Apache Mahout, and other popular free and open source technologies are listed here [14]. WEKA version 3.8.6 is one of them, and it is utilised in this study. WEKA stands for Waikato Environment for Knowledge Learning. It is a computer software created at New Zealand's University of Waikato to identify knowledge from raw data. It is free and open source software distributed under the GNU General Public License. WEKA can perform a variety of data mining tasks, including classification, regression, feature selection, data pre-processing, clustering, and visualizing. WEKA was initially built in C, but it was later rewritten in Java and is now compatible with the majority of computing platforms. It is simple to use, with a graphical interface that allows for easy setup and operation.

2.4 Wearable & Mobile Healthcare System

Wearable devices are used in many IoT systems. Qardiocore is a well-known and high-quality heart monitor that records and shows a user's full heart health on their smartphones [15]. The gadget produces highly precise readings and is one of the best items on the market for displaying real-time ECG graphs. It does, however, only identify heart abnormalities and allows users to exchange data with their doctors. It does not provide the patient/user with the option of alerting a qualified doctor in real time when their heart is in a critical situation, and it does not detect heart attacks. Neither of the techniques mentioned above has the capability of predicting heart attacks

A real-time mobile healthcare system had provided the monitoring elderly individuals indoors or outside areas in [3]. The system's basic components were a signal sensor, as well as a mobile. The bio-signal sensor data was supplied to an intelligent server for data collecting delivered to an intelligent server through GPRS/UMTS network. The device may monitor the older patient's movement, vital signs, location, and condition from a distance. Suggested a fully working wireless body area network (WBAN) system [16]. The developed system collected sensor-derived clinical data using medical bands. The author picked several medical bands to reduce interference between sensors and other existing equipment.

2.5 Heart Disease Prediction

Manpreet Singh et al. [17] created a heart disease prediction system using Structural Equation Modelling (SEM) and Fuzzy Cognitive Maps (FCM). They verified the data from the 2012 Canadian Community Health Survey (CCHS). They made use of twenty significant characteristics. SEM was used to build the weight matrix for the FCM model, which subsequently predicted the likelihood of cardiovascular illnesses. The association between CCC 121 and 20 attributes establishes a SEM model; CCC 121 is a variable that identifies whether or not the responder has heart disease. Prajakta Ghadge et al. [10] used big data to build an efficient heart attack prediction system. Because of its great occurrence, heart attacks must be identified quickly and properly. The primary purpose of this investigation was to develop an intelligent cardiac arrest prediction system model that makes use of big data and data mining modelling approaches. This technology might extract hidden facts about heart illness from a specific heart disease database.

CHAPTER 3

METHODOLOGY

3.1 Introduction

On this work, 2 types of the dataset will be use to apply on the algorithm which is the data from my hardware device and the existing data from Cleveland Heart Disease dataset. Next, effective machine learning method was chosen from those available in a Java-based open access data mining platform (WEKA) to measure and choose the best performance of the algorithm. I compare about 5 algorithm that will explain later. Then, employing the best algorithm into the Python to make a prediction of the heart disease for the each person.

3.2 Block Diagram

This proposed project of wireless early prediction system of cardiac arrest uses just one input sensor, the MAX30102 biometric sensor, which continuously measures the percentage of oxygen in blood, the value of pulse rate, and body temperature. The signal data is managed by the Arduino Pro Mini, which functions as a microcontroller to evaluate the sensor input. The data of vital signs is shown at two output user interfaces, one at the OLED screen display and one at the Blynk application cloud server, using Wi-Fi modulation implemented in the ESP8266. Finally, the smartphone's alert notification alarm will activate when it detects an increase or drop in the range of any of the parameters. Figure 3.1 below shows the block diagram on the development of wireless early prediction system of cardiac arrest.

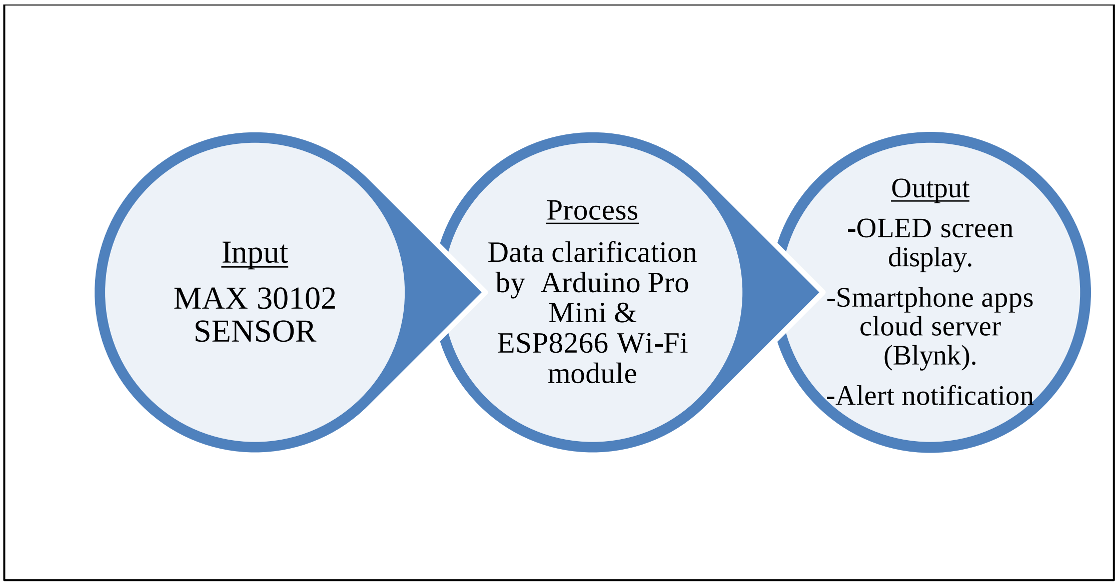

Figure 3.1 Block diagram of wireless early prediction system of cardiac arrest

3.3 Flow Chart

This project's entire operation flow includes of all elements that are input, process, and output. To begin, the suggested project of wireless early prediction system of cardiac arrest employs the MAX30102 sensor, which measures vital indicators including oxygen saturation in blood, pulse rate, and body temperature. The data obtained will then be computed in the microcontroller of the Arduino Pro Mini and the ESP8266 Wi-Fi module using the data signal processing that has been integrated in the controller. The data will be analysed by the controller. In typical resting time, the normal saturation of blood oxygen in blood must be greater than 97 percent, and the pulse rate must be greater than 60bpm to 100bpm. If the vital sign result is normal, the value will be displayed on the device's screen as well as on the Blynk application cloud server. Furthermore, if the data detects an abnormal measurement, the data collected will be shown, and an alert message will be activated, producing an alarm to inform the user. For normal conditions, the device will remain ON but in sleep mode after 5 seconds of displaying data. While, for abnormal condition warning notice, an alarm and alert display will be generated until the measurement data returns to normal. The flow chart of the system process is shown in Figure 3.2 below.

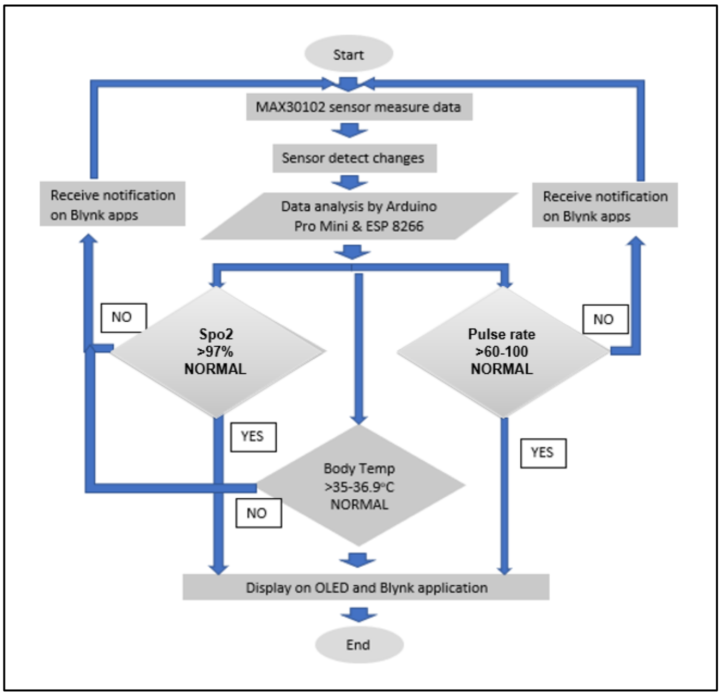

Figure 3.2 Flow chart of the project system.

3.4 System Architecture Proposal

This part presents an a summary of the suggested method and shows most of the elements, approaches, and tools employed in the system's development. A wearable device is required to assess the health status of patients in order to construct a smartphone-based application. To construct the continuous patient monitoring system, hardware components such as Arduino, several biomedical sensors, a wifi module, and a display monitor are required. Figure 1 depicts a block representation of the whole system workflow. All data from the device will be delivered to Blynk Apps that are linked to the WiFi module. If the system detects abnormal data, the app will notify the user. Data from the device is required to generate a massive dataset for use on the machine learning algorithms. After selecting the a strong algorithm with the highest accuracy and performance measurements from WEKA, that will be used in the creation of the Python in Google Colab for the detection and prediction of heart attacks.

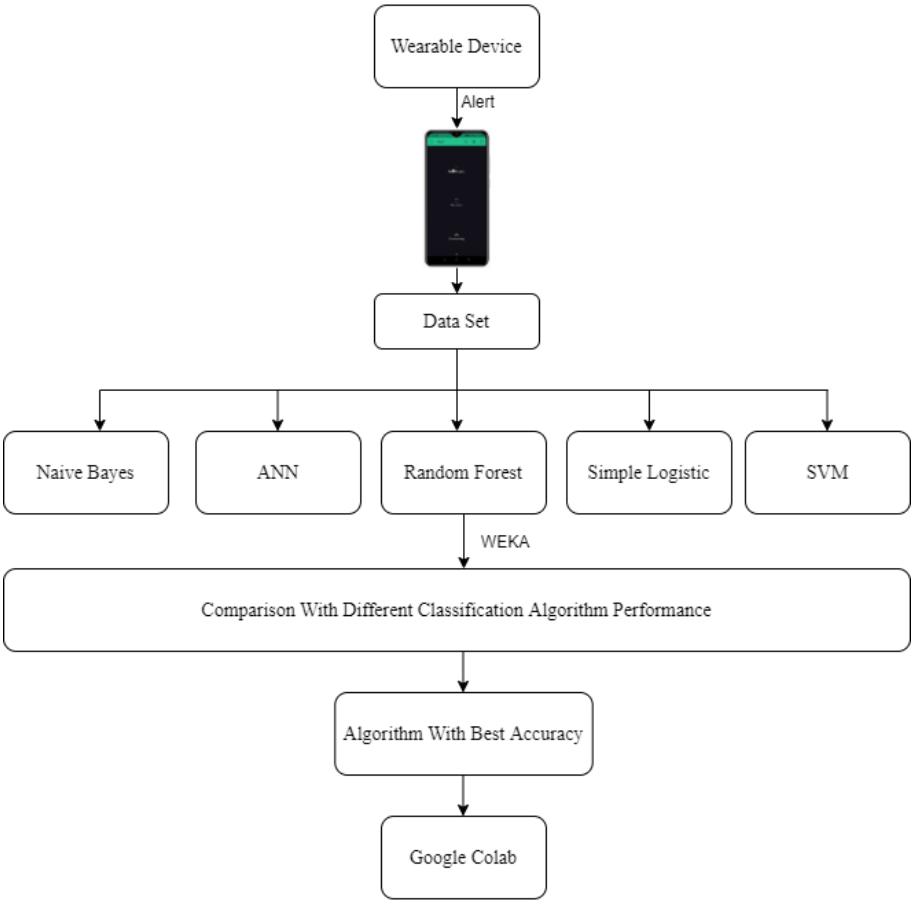

Figure 3.3 Flow of the project

3.5 Device Hardware

3.5.1 Circuit Diagram

The hardware electronics component of the design of a wireless heart attack prediction system is depicted in Figure 3.3 below. To begin, the battery was attached to an Arduino Pro Mini, which was then connected to an ESP822 Wi-Fi module to power up the system. Following that, the MAX30102 sensor is connected to the microcontroller, whose data will be processed throughout the measuring system. Furthermore, the ESP8266 Wi-Fi module will display readings on the screen display and on the user interface of the Blynk application cloud server. Apart from that, the Blynk Apps will generate an alert signal if the system detects abnormal changes in vital signs measurement data.

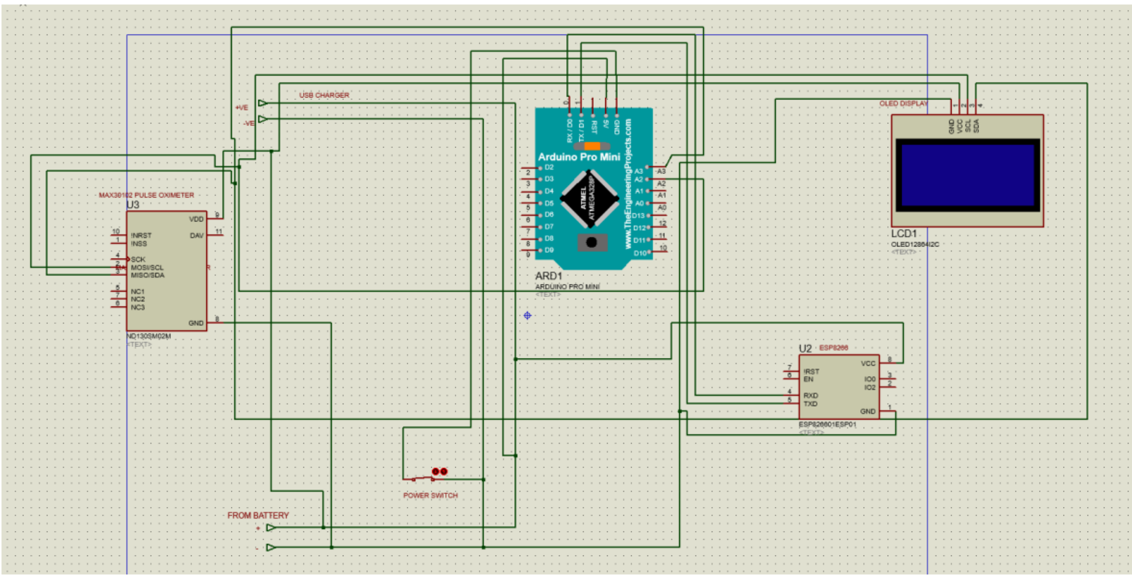

Figure 3.4 Circuit Diagram of The Project

3.5.2 Arduino Pro Mini

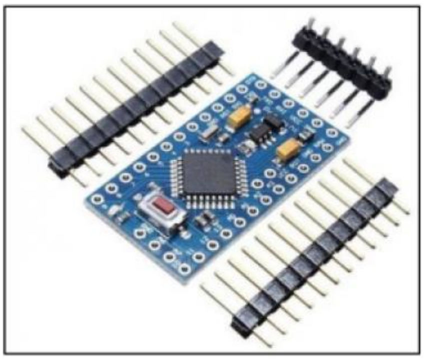

Figure 3.5 Arduino pro mini

The arduino pro small is a microcontroller board based on the Atmega32. In terms of general capability, the arduino pro small is quite similar to the arduino UNO, but the key distinction is its size and built-in programmer. There is just one on the Arduino Pro Mini. There are two types of Arduino Pro Mini available: one powered by 5V and running at 16MHz, and one powered by 3.3V and running at 8MHz [18].

Table 3.1 Arduino Pro Mini Key Features

Arduino Pro Mini Key Features		
No.	**Feature**	**Value**
1	Microcontroller	ATmega328
2	Operating Frequency/Crystal Oscillator	16MHz/8MHz
3	Digital I/O Pins	14
4	Analog Pins	8
5	PWM(Pulse Width Modulation) Pins	6
6	Built-in Programmer	Not available.
7	USB Port	Not available.
8	Flash Memory	32KB
9	SRAM	2KB
10	EEPROM	1KB
11	Bootloader	0.5KB in Flash Memory.

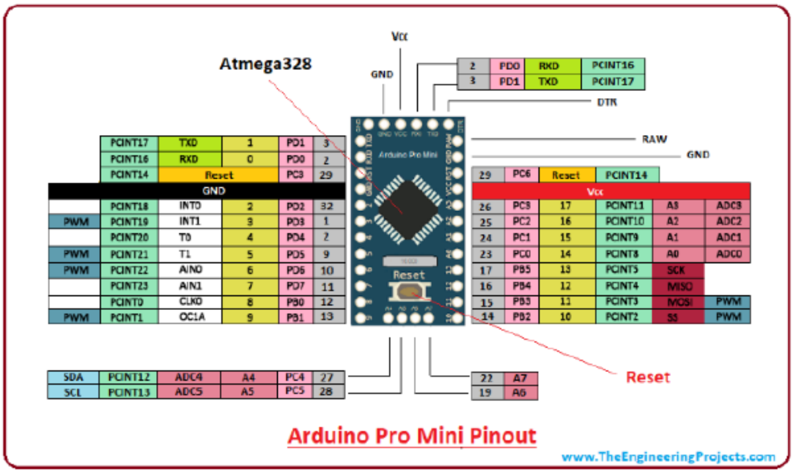

Figure 3.6 Arduino pro mini pinot

3.5.3 ESP8266 Wi-fi Module

Figure 3.7 ESP8266 Wi-fi Module

The ESP8266 Wi-Fi System is self SOC that includes an integrated TCP/IP protocol stack will link with every microcontroller may connect to your Wi-Fi connection. The ESP8266 will be used either host an application or delegate all Wi-Fi connection tasks to another application programmer Each ESP8266 module is pre-configured with an AT instruction set firmware, which means you can just connect it to an Arduino device and receive almost the same Wi-Fi-ability as a Wi-Fi Shield [19]. The ESP8266 module is a small board that has a large and growing user base. This module includes enough on-board storage and processing to communicate with sensors and other application-specific devices via its GPIOs yet needing little loading and load over execution. The ESP8266 enables APSD for VoIP applications and Bluetooth persistence interfaces, has self-calibrated RF that operates in any situation, and requires no extra RF components.

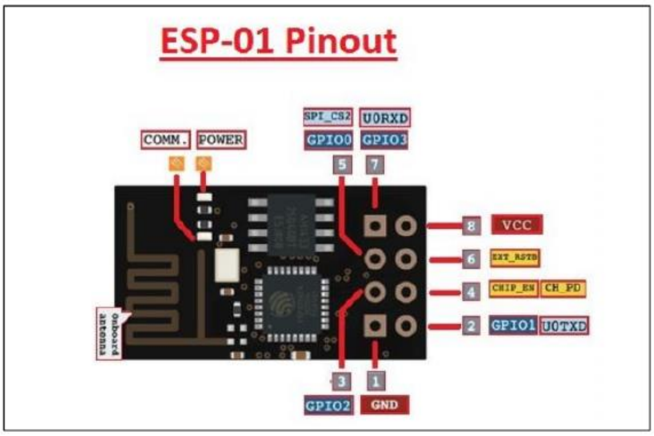

Figure 3.8 ESP8266 Pinout

3.5.4 MAX30102 Sensor

Figure 3.9 MAX30102 Sensor

The MAX30102 is a biosensor module that has heart rate monitor component and embedded pulse oximetry. The package includes internal LEDs, photodetectors, optical components, and low distortion and ambient light refusal electronics. This sensor is a full system solution that simplifies design of mobile and wearable devices. The MAX30102 is supplied by a single 1.8V supply and an extra 3.3V supply for the internal LEDs [20]. For communication, a standard I2C-compatible interface is employed. Through software, the module may be shut off with 0 percent standby current, leaving the power rails constantly enabled.

Table 3.2 MAX30102 Sensor Pinout

Pin Name	Description
VIN	1.8V-5V main power supply input terminal
SCL	Connect the I2C bus clock.
SDA	Data delivered via the I2C bus
INT	Pin interrupt
RD	RED of the MAX30102, LED ground terminal- commonly not connected
IRD	The IR, LED grounding of the MAX30102 – commonly not connected
GND	Ground wire

3.5.5 Organic Light Emitting Diode (OLED) 128”32 Screen display

Figure 3.10 OLED 128“32 Screen display

OLED display screens are the most recent advancement in display technology, having been introduced in flat-panel displays. It can produce light with great viewing angle, brightness, and contrast without requiring a backlight in a very tiny and well-organized package. This display has a 128 × 32 pixel revolution with white pixels on a black backdrop. The SSD1306 controller is used in modules that are compatible with the controller's software libraries. This display contains a 12C interface, which requires an MCU to interact with a location that has strong library support and an efficient operating system.

3.6 Machine Learning Algorithms

To validate the model performance, more than twenty machine learning algorithms by using the UCI Machine Learning Repository Dataset and Statlog Database, WEKA was utilised to predict cardiac disease. [21][22]. Those with better accuracy which is more than 80% are taken into account for performance measurement, and they are explained detailed below.

3.6.1 Naïve Bayes

Naive Bayes is a pretty effective predictive modelling method. It performs statistical classification that assumes no attribute dependency and attempts to optimise the marginal likelihood in identifying the class. In theory, this classifier has the lowest error rate, although this may not always be the case. Assumptions based on class conditional independence and a lack of sufficient probability data lead to inaccuracies. This model is related with two types of probabilities, both of which may be derived directly from the training dataset:

a) The likelihood of each class.

b) Each class's conditional probability with each x value.

According to Bayesian theorem $P(A \mid B) = P(A) * P(B / A) / P(B)$, where $P(B \mid A) = P(A \cap B) / P(A)$ [23]. Based on the earlier formula, a Bayesian classifier calculates the conditional possibility of an instance belonging to each class, and the instance is categorised as belonging to the class with the greatest conditional likelihood. After calculating these probabilities, the probabilistic model may be used to generate predictions using additional data using the Nave Bayes Theorem. When data is real-valued, it is more likely to have a Gaussian distribution (bell curve). As a result, these probabilities are easily approximated. Naive Bayes is that since it assumes that each input variable is independent. This classifier method employs conditional independence, which means it assumes that the value of an attribute on a particular result of class is unrelated to the values of those variables.

3.6.2 Artificial Neural Networks (ANN)

Artificial neural networks, also known as Multilayer Perceptrons, are biologically inspired and can represent exceedingly complicated non-linear processes. ANNs are one of the most important tools in machine learning. They are brain-oriented technology designed to emulate how people learn, as the word "neural" implies. Neural networks are made up of three layers: input, output, and hidden. A hidden layer is generally composed of units that convert the input into a form that the output layer can manage. ANNs are useful at detecting patterns that are too complex or ambiguous for a human programmer to extract and train the machine to identify. Neural networks have been used since the 1940s, and they have been a major aspect of artificial intelligence in recent decades as a result of the development of a new approach known as "backpropagation," which allows networks when the results don't really fit the creator's expectations, they know how to change their secret layers of neurons [24]. Figure 2 depicts the interaction of the layers.

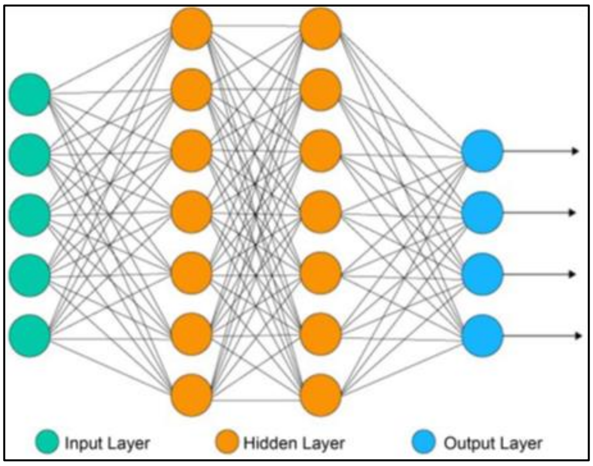

Figure 3.11 Artificial neural network

3.6.3 Support Vector Machine (SVM)

SVM is a method for ramifying both linear and nonlinear data. It employs a non-linear mapping approach to convert the training data to a big dimension. In SVM, a hyperplane is a type of line that splits the input variable space. The hyperplane has the ability to split elements in the input parameter space that have a class of 0 or 1 [25]. In two dimensions, this may be seen as a line, and it is expected that this line can entirely separate each input point. The margin is the distance between the hyperplane and the following data locations. The appropriate hyperplane is the line with the greatest margin of difference between the two classes. These are known as support vectors because they serve to define or sustain the hyperplane. In fact, an optimization algorithm calculates the values for the parameters that optimize the margin. The procedure of feature transformation is visualized in Fig 3.

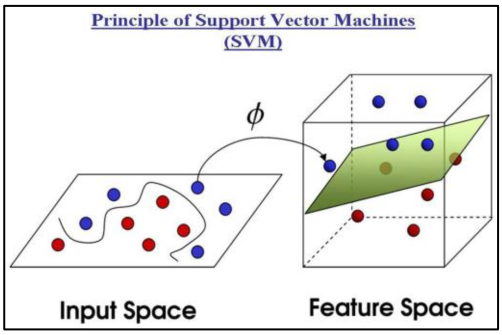

Figure 3.12 Support vector machine operation

3.6.4 Random Forest

The Random Forest is well-known and very effective machine learning technique. It is a example of machine learning method known as Bagging or Bootstrap Aggregation. The bootstrap is a very strong statistical technique for estimating a value from a data sample, such as mean. A large number of data samples are obtained, the mean is computed, and then all of the mean values are averaged to provide a better estimate of the true mean value [26]. The similar procedure is employed in bagging, however instead of calculating the mean of each data sample, decision trees are commonly utilised. Several samples of training data are explored here, and models are produced for each data sample. When a predictions for any data is required, each model makes a prediction, which is then averaged to provide a better estimate of the true output value.

3.6.5 Simple Logistic Regression

Simple logistic regression is one of the machine learning approaches derived from the study of statistics. This approach is suitable for binary classification, which is result are classified into two groups. Logistic regression is same as linear regression in that the purpose is to determine the coefficient result for all input value. In contrast to linear regression, the prediction of the output is made using a non-linear function known as a logistic function. The logistic function changes all result from 0 and 1. The logistic regression predictions are choosed to calculate the probability of a data value belonging to class 0 or class 1 [27]. This may be necessary in cases where greater reasoning for a prediction is required. When variables irrelevant to the output variable and qualities connected to one another are eliminated, logistic regression performs better.

3.7 Data Source

Two publicly accessible heart illness databases with the same kind and amount of features are combined to create a larger dataset with greater accuracy. The Cleveland Heart Disease dataset [21] has 303 records, while the device hardware dataset has 30 records with 13 comparable attributes, as shown in Table 1. By comparable characteristics, it was indicated that these features are employed for heart disease diagnosis and prediction in both the Cleveland Heart Disease dataset and the device hardware dataset. Both datasets were combined in order to improve the robustness of the suggested model. Instances with missing values are deleted after combining these two databases. The remaining 329 examples are utilised for model validation. WEKA version 3.8.6 was utilised for pre-processing and categorization utilising the following Machine learning methods.

Table 3.3 Input Attributes used for model formation and validation

No.	Attribute	Description
1	Age	Age of patients in year
2	Gender	1 for male, 0 for Female
3	Cp	Chest pain type (typical angina = 1, atypical angina = 2, non-anginal pain = 3, asymptomatic = 4)
4	Trestbps	Resting blood sugar (mm/Hg)
5	Chol	Serum cholesterol in mg/dl
6	Fbs	Fasting blood sugar > 120 mg/dl (true = 1, false = 0)
7	Restecg	Resting electrocardiographic results (normal = 0, having ST-T wave

		abnormality = 1, left ventricular hypertrophy = 2)
8	Thalach	Maximum heart rate
9	Exang	Exercise-induced angina
10	Old Peak	ST depression induced
11	Slope	Slope of the peak exercise ST segment
12	Ca	Number of major vessels
13	Thal	0 if normal, 2 if fixed defect, 3 if reverse defect.

3.8 Attribute Documentation

The dataset utilised for the data mining application has 13 distinct types of input attributes linked to various cardiovascular system variables. A key property was created for the construction of the Cloud-based classification and assume system. For selecting the two classed groups, predictable qualities are assigned a numerical scale which is have heart disease or not. Table 1 displays the input qualities, which include a total of 13 physiological parameters collected from the two previously stated datasets and utilised to predict any form of cardiac disease.

a) Input Attribute

b) Predictable Attribute

if the value is 0 means they do not have heart disease. However if the value got 1 means they have a heart disease.

On the classification process, all machine learning algorithms anticipate this property based on input attributes. This attribute value is separated into two groups: value 0 indicates that the user do not have heart disease, while values 1 - 4 are regarded as value 1, indicating that the user has heart disease. The processes of result analysis and performance monitoring are subsequently performed by categorising the dataset information into these two groups.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Hardware Design

The hardware electronics component of the design of a wireless early prediction system of cardiac arrest is depicted in Figure 3.3 below. To begin, the battery was attached to an Arduino Pro Mini, which was then connected to an ESP822 Wi-Fi module to power up the system. Following that, the MAX30102 sensor is connected to the microcontroller, whose data will be processed throughout the measuring system. Furthermore, the ESP8266 Wi-Fi module will display readings on the screen display and on the user interface of the Blynk application cloud server. Apart from that, the Blynk Apps will generate an alert signal if the system detects abnormal changes in vital signs measurement data. To generate and connect the hardware later, I will follow this circuit diagram to complete my hardware later.

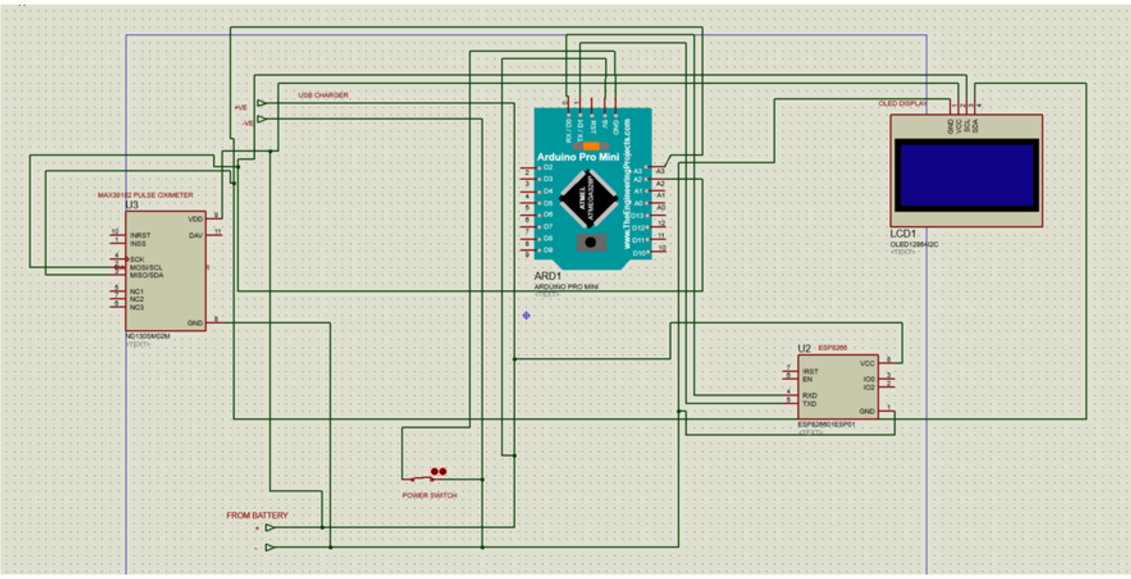

Figure 4.1 Circuit Diagram for Hardware Device

4.2 Machine Learning Algorithm Performance Analysis

There are several methods available that may be applied to the dataset. There are more than 10 algorithms are applied to the dataset, but only five major methods are evaluated and discussed, with an accuracy level of more than 80%. For testing the performance of the machine learning methods, the algorithms should be applied to the data set using 10-fold Cross-Validation on WEKA version 3.8.6. For performance analysis, the following metrics are computed [28]:

$$Precision = \frac{tp}{tp + fp}$$

$$Recall = \frac{tp}{tp + fn}$$

$$F_{score} = \frac{2 \times precision \times recall}{precision + recall}$$

$$Accuracy = \frac{tp + tn}{n}$$

$$Sensitivity = \frac{tp}{tp + fn}$$

$$Specificity = \frac{tn}{tn + fp}$$

n = Total number of instances, tp = true positive, tn = true negative, fp = false positive, fn = false negative.

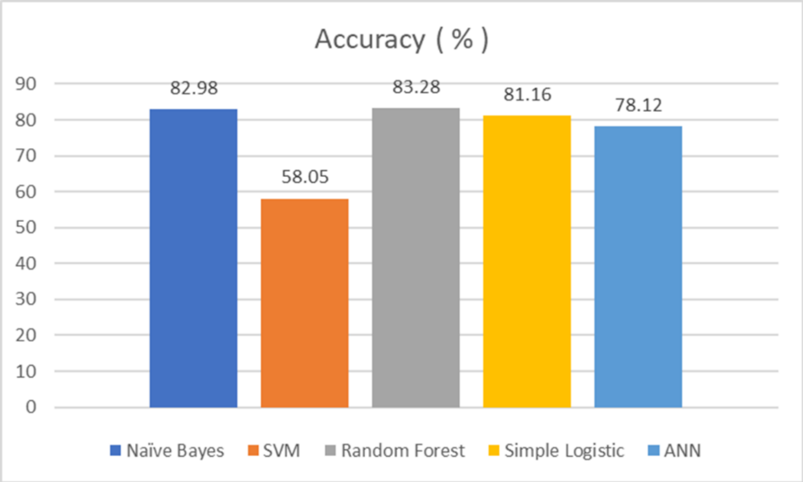

Figure 4.2 Result accuracy of the Analysis Algorithm

Table 4.1 Performance analysis of five machine learning algorithms (in %).

Evaluation Criteria	**Naïve Bayes**	**SVM**	**Random Forest**	**Simple Logistic**	**ANN**
Accuracy	82.98	58.05	83.28	81.16	78.12
Sensitivity	83	58.1	83.3	81.2	78.1
Specificity	79.3	97.6	84.2	83.6	79.4
False Out	17	42.2	16.7	18.9	21.9
Miss Rate	20.7	81.7	17.7	21.3	23.2
Precision	83.2	71.4	83.3	81.2	78.1
F-score	83.0	50.2	83.3	81.2	78.1

```
00:02:02 - trees.RandomForest

Bagging with 100 iterations and base learner

weka.classifiers.trees.RandomTree -K 0 -M 1.0 -V 0.001 -S 1 -do-not-check-capabilities

Time taken to build model: 0.18 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         274               83.2827 %
Incorrectly Classified Instances        55               16.7173 %
Kappa statistic                          0.6656
Mean absolute error                      0.2616
Root mean squared error                  0.348
Relative absolute error                 52.3179 %
Root relative squared error             69.5866 %
Total Number of Instances              329

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.823    0.158    0.839      0.823    0.831      0.666    0.910     0.915     0
                 0.842    0.177    0.827      0.842    0.835      0.666    0.910     0.900     1
Weighted Avg.    0.833    0.167    0.833      0.833    0.833      0.666    0.910     0.908

=== Confusion Matrix ===

   a   b   <-- classified as
 135  29 |   a = 0
  26 139 |   b = 1
```

Figure 4.3 Weka display screen showing different parameters for Random Forest.

4.3 Algorithm Analysis Performance

Table 4.1 shows that Naive Bayes, Random Forest, and Simple Logistic models have greater accuracy rates of more than 80%, making them significant models for biological systems of illness detection and prediction. In addition, several analytical criteria are explored in this study to establish the best performing model for the databases under consideration. Random Forest exceeds both Naive Bayes and Simple Logistic models in terms of accuracy, sensitivity, specificity, precision, and F-score. Furthermore, Random Forest has the lowest miss rate. The quantity of features also influences the model's categorization and prediction capabilities. According to the unpublished results of this work, as the number of features was decreased, the model performance metrics were similarly decreased, with Random Forest still giving the greatest performance. The same study was repeated in Python, and Random Forest again exceeded the models with Radial Basis Kernel Function. As a result, when all 13 characteristics are examined, Random Forest is the most efficient algorithm to be used on the heart disease prediction system, as discovered in our study. Previous studies [13][29][30] comparing several machine learning algorithms discovered that no system achieved an accuracy level of more than 80% in heart disease prediction using the same quantity and variety of features as employed in this study. Furthermore, two datasets with the same quantity and kind of characteristics were integrated in this study. As a result, the chosen model will be more robust than those presented in previous research.

4.4 Heart Disease Prediction Using Python

After comparing the performance of the best algorithm, Random Forest, I implement Random Forest Algorithm into the model classifier at Python using Google Colab to detect heart problems. From the Python also can measure the accuracy of the training and test data set. Moreover, to make a prediction for 1 dataset or 1 person, I use a ‘reshapped’ coding. This coding means to predict only 1 data. Next, I need to input the data for 1 person and the Python will predict whether the person have a heart disease or do not have heart disease.

```
model = RandomForestClassifier()
# training the RandomForestclassifier model with Training data
model.fit(X_train, Y_train)
# accuracy on training data
X_train_prediction = model.predict(X_train)
training_data_accuracy = accuracy_score(X_train_prediction, Y_train)
print('Accuracy on Training data : ', training_data_accuracy)
# accuracy on test data
X_test_prediction = model.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
print('Accuracy on Test data : ', test_data_accuracy)

input_data = (32,1,0,119,132,0,1,137,1,0.6,2,0,2)

# change the input data to a numpy array
input_data_as_numpy_array= np.asarray(input_data)

# reshape the numpy array as we are predicting for only on instance
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

prediction = model.predict(input_data_reshaped)
print(prediction)

if (prediction[0]== 0):
  print('The Person does not have a Heart Disease')
else:
  print('The Person has Heart Disease')
```

```
Accuracy on Training data :  1.0
Accuracy on Test data :  0.8181818181818182
[0]
The Person does not have a Heart Disease
```

Figure 4.4 Result from Python

CHAPTER 5

CONCLUSION

5.1 Conclusion

The main objective of this project was to invent friendly and wearable health monitoring using IoT and develop system and prediction of the cardiac arrest by using machine learning techniques. The data from the device and existing data of the body health condition is the most important things to create the dataset of the system. Next, WEKA software was be used to make analysis of the best performance of the algorithm that need to apply into the Python by using google colab to make a prediction of the heart disease of the patients. Random Forest algorithm be used in this project that have best performance which is got 83% accuracy then the other algorithms. So to make the prediction of the cardiac arrest or heart disease, the user need to insert their data in the Python with the Random Forest algorithm. From there the user will no and alert about their health condition either they have heart disease or not. If the user detect they have heart disease, they need to check with the doctor immediately to prevent worst condition.

5.2 Summary

On this work, an effective machine learning method was chosen from among those available in a Java-based open access data mining platform (WEKA) to detect the existence or possibility of cardiac disease in a big dataset. Then, utilising an Arduino-based microcontroller system, a continuous cardiac monitoring system design was presented. The following are the suggested system's step-by-step design methods and the overall system's workflow:

- A main wireless patient health monitoring system is constructed utilising Arduino with temperature, spo2 and blood pressure sensors to gather real-time patient physiological data and detect the patient's severe condition.

- All sensor data are automatically recorded and will be update on the server database after a set period (every 10 seconds), and if any undesired sensor value is detected, they will get an alert message with the current sensor data on his phone.

- If the abnormal reading is constant, the user should seek medical attention in a hospital before having cardiac arrest.

- The patient may manually enter all heart disease data on the dataset, as well as sensor data from the device, such as heartbeat. As a result, they can identify whether or not they have heart disease.

- Upload the dataset into WEKA to compare the accuracy and performance of several data mining algorithms in detecting heart disease.

- Selection of the best algorithm based on model performance attributes for developing heart disease prediction in Google Colab.

- To anticipate the condition of another patient's cardiac arrest, just enter the data set into the algorithm in Google Colab.

5.3 Future Recommendation

In the future, a Photoplethysmography (PPG)-based blood pressure sensor module or electronic sphygmomanometer may be attached to the Arduino and broadcast real-time data to the server. This sensor is not included in the intended patient monitoring system due to the lack of a clinically recognised system at this time, despite the fact that much research is being conducted on the development of a PPG-based blood pressure monitor [31]. Though a cloud-based cardiac disease detection application concept is depicted here, future work will focus on the establishment of a dedicated server and data base for this sort of patient monitoring application. If the suggested application is successfully integrated and developed, it will be made accessible in the Android Play store. As a result, every patient or doctor from any location of the world will be able to install and utilise this programme for heart disease prediction, as detailed in [32]. This method may be utilised for illness patient monitoring in addition to heart disease.

REFERENCES

[1] H. Animesh, K. M. Subrata, G. Amit, M. Arkomita, and A. Mukherje, “Heart Disease Diagnosis and Prediction Using Machine Learning and Data Mining Techniques: A Review,” *Adv. Comput. Sci. Technol.*, 2017.

[2] J. Patel, T. Upadhyay, and S. Patel, “Heart Disease Prediction Using Machine learning and Data Mining Technique,” *IJCSC*, 2017.

[3] A. B. Chavanpatil and P. S. S. Sonawane, “To Predict Heart Disease Risk and Medications Using Data Mining Techniques With an IoT Based Monitoring System For Post Operative Heart Disease Patients .,” *Procedia Int. J. Emerg. Trends Technol.*, 2017.

[4] S. F. Weng, J. Reps, J. Kai, J. M. Garibaldi, and N. Qureshi, “Can Machine-learning improve cardiovascular risk prediction using routine clinical data?,” *PLoS One*, 2017, doi: 10.1371/journal.pone.0174944.

[5] W. Zhao, C. Wang, and Y. Nakahira, “Medical application on internet of things,” *IET Conf. Publ.*, vol. 2011, no. 586 CP, pp. 660–665, 2012, doi: 10.1049/cp.2011.0751.

[6] M. Shouman, T. Turner, and R. Stocker, “Using data mining techniques in heart disease diagnosis and treatment,” 2012, doi: 10.1109/JEC-ECC.2012.6186978.

[7] S. M. R. Islam, D. Kwak, M. H. Kabir, M. Hossain, and K. S. Kwak, “The internet of things for health care: A comprehensive survey,” *IEEE Access*, 2015, doi: 10.1109/ACCESS.2015.2437951.

[8] Purushottam, K. Saxena, and R. Sharma, “Efficient heart disease prediction system using decision tree,” 2015, doi: 10.1109/CCAA.2015.7148346.

[9] S. Palaniappan and R. Awang, “Intelligent heart disease prediction system using data mining techniques,” 2008, doi: 10.1109/AICCSA.2008.4493524.

[10] H. Sharma and M. A. Rizvi, “Prediction of Heart Disease using Machine Learning Algorithms: A Survey International,” *Int. J. Recent Innov. Trends Comput. Commun.*, 2017.

[11] A. Golande and T. Pavan Kumar, “Heart disease prediction using effective machine learning techniques,” *Int. J. Recent Technol. Eng.*, 2019.

[12] I. Chiuchisan and O. Geman, “An Approach of a Decision Support and Home Monitoring System for Patients with Neurological Disorders using Internet of Things Concepts Faculty of Electrical Engineering and Computer Science,” *Wseas Trans. Syst.*, 2014.

[13] J. Soni, U. Ansari, D. Sharma, and S. Soni, “Predictive Data Mining for Medical Diagnosis: An Overview of Heart Disease Prediction,” *Int. J. Comput. Appl.*, 2011, doi: 10.5120/2237-2860.

[14] M. Kumari and S. Godara, “Comparative Study of Data Mining Classification Methods in Cardiovascular Disease Prediction,” *Int. J. Comput. Sci. Trends Technol.*, 2011.

[15] “Smart Wearable ECG EKG Monitor - QardioCore.” https://www.qardio.com/qardiocore-wearable-ecg-ekg-monitor-iphone/ (accessed Jun. 25, 2022).

[16] M. R. Yuce, “Implementation of wireless body area networks for healthcare systems,” *Sensors Actuators, A Phys.*, 2010, doi: 10.1016/j.sna.2010.06.004.

[17] M. Singh, L. M. Martins, P. Joanis, and V. K. Mago, “Building a cardiovascular disease predictive model using structural equation model & fuzzy cognitive map,” 2016, doi: 10.1109/FUZZ-IEEE.2016.7737850.

[18] “Arduino Pro Mini | Arduino Documentation | Arduino Documentation.” https://docs.arduino.cc/retired/boards/arduino-pro-mini (accessed Jun. 17, 2022).

[19] “WiFi Module - ESP8266 (4MB Flash) - WRL-17146 - SparkFun Electronics.” https://www.sparkfun.com/products/17146 (accessed Jun. 17, 2022).

[20] “MAX30102--High-Sensitivity Pulse Oximeter and Heart-Rate Sensor for Wearable Health,” Accessed: Jun. 17, 2022. [Online]. Available: www.maximintegrated.com.

[21] “UCI Machine Learning Repository: Heart Disease Data Set.” http://archive.ics.uci.edu/ml/datasets/Heart+Disease (accessed Jun. 17, 2022).

[22] “uci/statlog-heart | Workspace | data.world.” https://data.world/uci/statlog-heart/workspace/file?filename=heart.dat.txt (accessed Jun. 17, 2022).

[23] “Naive Bayes Classifier in Machine Learning - Javatpoint.” https://www.javatpoint.com/machine-learning-naive-bayes-classifier (accessed Jun. 17, 2022).

[24] “Artificial Neural Network (ANN) in Machine Learning - DataScienceCentral.com.” https://www.datasciencecentral.com/artificial-neural-network-ann-in-machine-learning/ (accessed Jun. 17, 2022).

[25] “SVM | Support Vector Machine Algorithm in Machine Learning.” https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/ (accessed Jun. 17, 2022).

[26] "Understanding Random Forest. How the Algorithm Works and Why it Is… | by Tony Yiu | Towards Data Science." https://towardsdatascience.com/understanding-random-forest-58381e0602d2 (accessed Jun. 17, 2022).

[27] "Introduction to Machine Learning Algorithms: Linear Regression | by Rohith Gandhi | Towards Data Science." https://towardsdatascience.com/introduction-to-machine-learning-algorithms-linear-regression-14c4e325882a (accessed Jun. 17, 2022).

[28] P. S. Kumar and S. Pranavi, "Performance analysis of machine learning algorithms on diabetes dataset using big data analytics," *2017 Int. Conf. Infocom Technol. Unmanned Syst. Trends Futur. Dir. ICTUS 2017*, vol. 2018-Janua, no. Iddm, pp. 508–513, 2018, doi: 10.1109/ICTUS.2017.8286062.

[29] C. S.Dangare and S. S. Apte, "Improved Study of Heart Disease Prediction System using Data Mining Classification Techniques," *Int. J. Comput. Appl.*, 2012, doi: 10.5120/7228-0076.

[30] H. D. Masethe and M. A. Masethe, "Prediction of heart disease using classification algorithms," 2014.

[31] M. Elgendi, Y. Liang, and R. Ward, "Toward Generating More Diagnostic Features from Photoplethysmogram Waveforms," *Diseases*, 2018, doi: 10.3390/diseases6010020.

[32] F. Ahmed, "An Internet of Things (IoT) Application for Predicting the Quantity of Future Heart Attack Patients," *Int. J. Comput. Appl.*, 2017, doi: 10.5120/ijca2017913773.

APPENDIX A
LIST OF DATA SET

```
@relation heart
@attribute 'age' numeric
@attribute 'sex'{'0','1'}
@attribute 'cp' numeric
@attribute 'trestbps' numeric
@attribute 'chol' numeric
@attribute 'fbs' numeric
@attribute 'restecg' numeric
@attribute 'thalach' numeric
@attribute 'exang' numeric
@attribute 'oldpeak' numeric
@attribute 'slope' numeric
@attribute 'ca' numeric
@attribute 'thal' numeric
@attribute 'diagnosis' {'0','1'}
@data
63,1,3,145,233,1,0,150,0,2.3,0,0,1,1
37,1,2,130,250,0,1,187,0,3.5,0,0,2,1
41,0,1,130,204,0,0,172,0,1.4,2,0,2,1
56,1,1,120,236,0,1,178,0,0.8,2,0,2,1
57,0,0,120,354,0,1,163,1,0.6,2,0,2,1
57,1,0,140,192,0,1,148,0,0.4,1,0,1,1
56,0,1,140,294,0,0,153,0,1.3,1,0,2,1
44,1,1,120,263,0,1,173,0,0,2,0,3,1
52,1,2,172,199,1,1,162,0,0.5,2,0,3,1
57,1,2,150,168,0,1,174,0,1.6,2,0,2,1
54,1,0,140,239,0,1,160,0,1.2,2,0,2,1
48,0,2,130,275,0,1,139,0,0.2,2,0,2,1
49,1,1,130,266,0,1,171,0,0.6,2,0,2,1
64,1,3,110,211,0,0,144,1,1.8,1,0,2,1
58,0,3,150,283,1,0,162,0,1,2,0,2,1
50,0,2,120,219,0,1,158,0,1.6,1,0,2,1
58,0,2,120,340,0,1,172,0,0,2,0,2,1
66,0,3,150,226,0,1,114,0,2.6,0,0,2,1
43,1,0,150,247,0,1,171,0,1.5,2,0,2,1
69,0,3,140,239,0,1,151,0,1.8,2,2,2,1
59,1,0,135,234,0,1,161,0,0.5,1,0,3,1
44,1,2,130,233,0,1,179,1,0.4,2,0,2,1
42,1,0,140,226,0,1,178,0,0,2,0,2,1
61,1,2,150,243,1,1,137,1,1,1,0,2,1
40,1,3,140,199,0,1,178,1,1.4,2,0,3,1
71,0,1,160,302,0,1,162,0,0.4,2,2,2,1
```

```
57,1,0,110,335,0,1,143,1,3,1,1,3,0
55,0,0,128,205,0,2,130,1,2,1,1,3,0
61,1,0,148,203,0,1,161,0,0,2,1,3,0
58,1,0,114,318,0,2,140,0,4.4,0,3,1,0
58,0,0,170,225,1,0,146,1,2.8,1,2,1,0
67,1,2,152,212,0,0,150,0,0.8,1,0,3,0
44,1,0,120,169,0,1,144,1,2.8,0,0,1,0
63,1,0,140,187,0,0,144,1,4,2,2,3,0
63,0,0,124,197,0,1,136,1,0,1,0,2,0
59,1,0,164,176,1,0,90,0,1,1,2,1,0
57,0,0,140,241,0,1,123,1,0.2,1,0,3,0
45,1,3,110,264,0,1,132,0,1.2,1,0,3,0
68,1,0,144,193,1,1,141,0,3.4,1,2,3,0
57,1,0,130,131,0,1,115,1,1.2,1,1,3,0
57,0,1,130,236,0,0,174,0,0,1,1,2,0
25,1,0,113,130,0,1,123,0,0.8,0,0,1,0
24,1,0,93,125,0,1,115,0,1.2,0,0,1,0
24,0,0,100,130,0,0,135,0,0,2,0,1,0
25,0,0,89,119,0,1,141,1,0.4,2,0,1,0
31,1,0,108,132,0,1,145,0,0,2,0,1,0
35,1,0,117,136,0,0,135,0,0.5,1,0,2,0
25,0,0,99,110,0,0,139,0,1.4,1,0,2,0
26,0,0,81,124,0,1,130,0,1.4,2,2,1,0
26,0,0,91,126,0,0,125,1,0,2,0,2,0
30,1,0,100,130,0,1,145,1,1.6,2,0,1,0
31,1,0,110,131,0,1,148,0,0.8,2,0,1,0
25,1,0,89,125,0,1,139,1,0.8,2,0,2,0
35,1,1,121,135,0,0,147,0,1.5,2,0,3,0
31,0,1,115,131,0,0,134,0,0.2,1,2,1,0
30,0,0,117,130,0,0,139,0,3,2,0,1,0
26,1,0,99,126,0,1,129,0,0.4,1,0,2,0
31,1,0,110,131,0,1,140,0,0,2,1,2,0
27,1,0,97,127,0,0,148,1,0.2,0,0,1,0
29,0,0,89,129,0,1,149,0,0,2,1,2,0
32,0,1,114,132,0,0,146,0,0,2,0,1,0
33,1,1,121,133,0,1,140,1,0,1,0,2,0
29,1,0,120,129,0,1,139,1,0,2,0,1,0
26,1,0,97,126,0,1,130,0,0.5,2,0,2,0
25,0,0,99,125,0,0,128,1,0.4,1,0,1,0
35,1,1,116,135,0,1,140,1,1.8,2,0,2,0
32,1,0,119,132,0,1,137,1,0.6,2,0,2,0
```

APPENDIX B
CODING FOR WIFI MODULE

```
#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleStream.h>
const char* ssid = "server";
const char* pass = "12345678";
char auth[] = "LHR79XN3ljZJ5E-wLVntJC740ZzpTppa";
char server[] = "139.59.206.133";
int ALARM4=0;
int ALARM3=0;
int SPO2x=0;
int BPMx=0;
String DATA="";
int P1=0, P2=0, P3=0, P4=0;
int Rly1=0, Rly2=0, Rly3=0, Rly4=0, Rly5=0;
int led1x=0,led2x=0,led3x=0,led4x=0;
int TotalUse=0;
int TotalAvai=0;
float Temp1=30.1423;
float PH=7;
float Temp2=30.2;
String Flat;
String Flon;
String Temp1x="";
String PHx="";
String Temp2x="";
String Temp1y="";
String PHy="";
String Temp2y="";
String Temp3y="";
String Temp3x="";
String Temp4y="";
String Temp4x="";
String Temp5y="";
String Temp5x="";
String Temp6y="";
String Temp6x="";
String Temp7y="";
String Temp7x="";
String Temp8y="";
String Temp8x="";
String Temp9y="";
String Temp9x="";
String Temp10y="";
String Temp10x="";
```

```
int Timer=0;
int Mode=0;
int DataIn=0;
int ALERT=0;
WiFiClient wifiClient;
bool connectBlynk()
{
  wifiClient.stop();
  return wifiClient.connect(server, BLYNK_DEFAULT_PORT);
}
void connectWiFi()
{
  Serial.print("Connecting to ");
  Serial.println(ssid);
  if (pass && strlen(pass)) {
    WiFi.begin((char*)ssid, (char*)pass);
  } else {
    WiFi.begin((char*)ssid);
  }
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
}
//-----------------------Manage Virtual Pin---------------------------
BLYNK_WRITE(V10)
{
  int pinValue = param.asInt(); // assigning incoming value from pin V1 to a variable
  Rly1=pinValue;
  // process received value
}
BLYNK_WRITE(V11)
{
  int pinValue1 = param.asInt(); // assigning incoming value from pin V1 to a variable
  Rly2=pinValue1;
}
BLYNK_WRITE(V12)
{
  int pinValue3 = param.asInt(); // assigning incoming value from pin V1 to a variable
  Rly3=pinValue3;
  // process received value
}
BLYNK_WRITE(V13)
{
  int pinValue4 = param.asInt(); // assigning incoming value from pin V1 to a variable
  Rly4=pinValue4;
}
BLYNK_WRITE(V14)
```

```
{
  int pinValue5 = param.asInt(); // assigning incoming value from pin V1 to a variable

  Rly5=pinValue5;
}

void setup()
{
  // Debug console
  Serial.begin(9600);
  connectWiFi();
  connectBlynk();
  Blynk.begin(wifiClient, auth);
}
void loop()
{
  if (WiFi.status() != WL_CONNECTED) {
    connectWiFi();
    return;
  }
  if (!wifiClient.connected()) {
    connectBlynk();
    return;
  }
  Blynk.run();
Timer++;
  if (Rly3==1)
  }
if (Timer > 1000){
  Timer=0;
}
while (Serial.available()) {
    // get the new byte:
    char inChar1 = (char)Serial.read();
  if (inChar1 == '*') {
      DataIn++;
    }
if (inChar1 == 'X'){
  Blynk.notify("High body temperature!");
}
if (inChar1 == 'Y'){
  Blynk.notify("High BPM!!");
}
    while (DataIn > 0){
        while (Serial.available()) {
    // get the new byte:
    char inChar = (char)Serial.read();
    if (inChar == '*') {
```

```
    DataIn++;
   }
   if (inChar != '*' && inChar != '#' && DataIn==1) {
    Temp1x+=inChar;
   }
   if (inChar != '*' && inChar != '#' && DataIn==2) {
    Temp2x+=inChar;
   }
   if (inChar != '*' && inChar != '#' && DataIn==3) {
    Temp3x+=inChar;

   }
    if (inChar != '*' && inChar != '#' && DataIn==4) {
    Temp4x+=inChar;
   }
    if (inChar != '*' && inChar != '#' && DataIn==5) {
    Temp5x+=inChar;
   }
    if (inChar != '*' && inChar != '#' && DataIn==6) {
    Temp6x+=inChar;
   }
    if (inChar != '*' && inChar != '#' && DataIn==7) {
    Temp7x+=inChar;
   }
    if (inChar != '*' && inChar != '#' && DataIn==8) {
    Temp8x+=inChar;
   }
    if (inChar != '*' && inChar != '#' && DataIn==9) {
    Temp9x+=inChar;
   }
    if (inChar != '*' && inChar != '#' && DataIn==10) {
    Temp10x+=inChar;
   }
   if (inChar == '#') {
    DataIn=0;
    Temp1y=Temp1x;   PHy=PHx;     Temp2y=Temp2x;  Temp3y=Temp3x;
Temp4y=Temp4x;
    Temp5y=Temp5x;
    Temp6y=Temp6x;
    Temp7y=Temp7x;
    Temp8y=Temp8x;
    Temp9y=Temp9x;
    Temp10y=Temp10x;
    Temp1x="";
    PHx="";   Temp2x="";
    Temp3x="";
    Temp4x="";
    Temp5x="";
```

```
     Temp6x="";
     Temp7x="";
     Temp8x="";
     Temp9x="";
     Temp10x="";
    Blynk.virtualWrite(V0, Temp1y);
    Blynk.virtualWrite(V1, Temp2y);
    Blynk.virtualWrite(V2, Temp3y);
    SPO2x=Temp2y.toInt();
    BPMx=Temp1y.toInt();
    if (SPO2x<=96){
      if (ALARM3==0){
        ALARM3=1;
        Blynk.notify("LOW OXYGEN!!");
      }
    }
     if (SPO2x>96){
      ALARM3=0;
     }
      if (BPMx<=60){
      if (ALARM4==0){
        ALARM4=1;
        Blynk.notify("LOW BPM!!");
      }
    }
     if (BPMx>60){
      ALARM4=0;
     }
    }
        }
    }
}
}
```

APPENDIX C
CODING FOR ARDUINO

```
#include <I2Cdev.h>
#include <U8glib.h>
#include <Adafruit_GFX.h>
#include <Adafruit_GrayOLED.h>
#include <Adafruit_SPITFT.h>
#include <Adafruit_SPITFT_Macros.h>
#include <gfxfont.h>
#include <Adafruit_SSD1306.h>
#include <heartRate.h>
#include <MAX30105.h>
#include <spo2_algorithm.h>
#include <Wire.h>
#include "MAX30105.h"
#include <SoftwareSerial.h>
#include <SPI.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <EEPROM.h>
#include "heartRate.h"
MAX30105 particleSensor;
SoftwareSerial ss(2, 3); //(RX,TX)
Adafruit_SSD1306 display(OLED_RESET);
int ALARM=0;
int ALARM1=0;
float TD=0;
float TDm=0;
int CheckR=0;
int Beat=0;
float OldIrValue=0;
int BeatCount=0;
float VLDL=0;
float CH=0;
float LD=0;
float HD=0;
float CHx=0;
float LDx=0;
float HDx=0;
float Tr=0;
const byte RATE_SIZE = 4; //Increase this for more averaging. 4 is good.
byte rates[RATE_SIZE]; //Array of heart rates
byte rateSpot = 0;
long lastBeat = 0; //Time at which the last beat occurred
int Tcount=0;
float beatsPerMinute,AvgRead,Glucose;
```

```
int beatAvg,i;
float AvgMax,AvgMaxR,AvgMin,AvgMinR,Reading;
int MODE=0;
float NewBPM=0;
float Tempx=0;
long previousMillis = 0;
long interval = 3000;
long previousMillis1 = 0;
long interval1 = 10000;
//---------------------------------------------
  long UpperThreshold = 518;
    long LowerThreshold = 490;
    long reading = 0;
    float BPM = 0.0;
    bool IgnoreReading = false;
    bool FirstPulseDetected = false;
    unsigned long FirstPulseTime = 0;
    unsigned long SecondPulseTime = 0;
    unsigned long PulseInterval = 0;
    int MyTimer=0;
//------------------------------------------------
static const unsigned char PROGMEM logo16_glcd_bmp[] =
{ B00000000, B11000000,
  B00000001, B11000000,
  B00000001, B11000000,
  B00000011, B11100000,
  B11110011, B11100000,
  B11111110, B11111000,
  B01111110, B11111111,
  B00110011, B10011111,
  B00011111, B11111100,
  B00001101, B01110000,
  B00011011, B10100000,
  B00111111, B11100000,
  B00111111, B11110000,
  B01111100, B11110000,
  B01110000, B01110000,
  B00000000, B00110000 };
#if (SSD1306_LCDHEIGHT != 32)
#error("Height incorrect, please fix Adafruit_SSD1306.h!");
#endif
void setup()
{
  delay(3000);
  Serial.begin(9600);
  ss.begin(9600);
  Serial.println("Initializing...");
  pinMode(ledg,OUTPUT);
```

```
  pinMode(ledr,OUTPUT);
digitalWrite(ledg,HIGH);
digitalWrite(ledr,HIGH);
  // Initialize sensor
  if (!particleSensor.begin(Wire, I2C_SPEED_FAST)) //Use default I2C port, 400kHz speed
  {
    Serial.println("MAX30105 was not found. Please check wiring/power. ");
    while (1);
  }
  Serial.println("Place your index finger on the sensor with steady pressure.");
  particleSensor.setup(); //Configure sensor with default settings
  particleSensor.setPulseAmplitudeRed(0x0A); //Turn Red LED to low to indicate sensor is
running
  particleSensor.setPulseAmplitudeGreen(0x0A); //Turn off Green LED
display.begin(SSD1306_SWITCHCAPVCC, 0x3C);  // initialize with the I2C addr 0x3C (for
the 128x32)
  display.display();
  delay(100);
        display.clearDisplay();
    display.setTextSize(2);
  display.setTextColor(WHITE);
  display.setCursor(0,0);
  display.println("  WELCOME..");
  //display.println("  Initialize Data");
  display.setTextColor(BLACK, WHITE); // 'inverted' text
  display.display();
  delay(800);
  display.clearDisplay();
digitalWrite(ledg,LOW);
digitalWrite(ledr,LOW);
 // display.clearDisplay();
}
void loop()
{
  float temperature = particleSensor.readTemperature();
  long irValue = particleSensor.getIR();
  Reading=irValue/1000;
float SPO2;
       SPO2=(irValue/1000);
       SPO2=(SPO2/100*11) + 92;
       if (SPO2>100){
        SPO2=100;
        }
Tcount++;
if (Tcount>=10){
  temperature=(temperature*10) + 10;
   ss.print("*");
  ss.print(AvgMax);
```

```
   ss.print("*");
   ss.print(temperature,1);
    ss.print("*");
       ss.print(Glucose);
       ss.print("*");
       ss.print(SPO2);
     ss.println("#");
     Tcount=0;
 }
if (temperature<100){
  Tempx=temperature+2;
}
  unsigned long currentMillis = millis();
  unsigned long currentMillis1 = millis();
  if(currentMillis - previousMillis > interval) {
    previousMillis = currentMillis;
 BeatCount=Beat;
 Beat=0;
 BPM=BeatCount*60 * 0.07547;
 if (BPM>50 && BPM<160){
  NewBPM=BPM;
}
if (irValue < 40000){
     display.clearDisplay();
    display.setTextSize(2);
  display.setTextColor(WHITE);
  display.setCursor(0,0);
  display.println("TIME");
  display.setTextColor(BLACK, WHITE); // 'inverted' text
  display.display();
  delay(500);
  }
  if (MyTimer>=10 && irValue>40000){
      AvgMaxR=AvgMax;
      AvgMinR=AvgMin;
      UpperThreshold=AvgMaxR-3;
      LowerThreshold=AvgMinR+3;
  MyTimer=0;  AvgMax=0; AvgMin=100;
   display.clearDisplay();
    display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(0,0);
  display.print("BPM:");
  display.setTextSize(2);
  display.println(NewBPM);
  display.setTextSize(1);
  display.print("T(C):");
  display.setTextSize(1);
```

```
  display.println(Tempx);
 display.setTextSize(1);
  display.print("SPO2:");
  display.setTextSize(1);
  display.println(SPO2);
  display.setTextColor(BLACK, WHITE); // 'inverted' text
  display.display();
  ss.print("*");
ss.print(NewBPM);
ss.print("*");
ss.print(SPO2);
ss.print("*");
ss.print(Tempx);
ss.println("#");
}
  }
  if (irValue > 40000){
    TD++;
    if (irValue>OldIrValue+25 || irValue<OldIrValue-25){
      Beat++;
      OldIrValue=irValue;
      TDm=TD*100;
      TD=0;
    }
   if (NewBPM>120){
    if (ALARM==0){
      ALARM=1;
      ss.println("Y");
    }
   }
   if (NewBPM>30 && NewBPM<120){
    ALARM=0;
   }
   if (Tempx>=37){
    if (ALARM1==0){
      ALARM1=1;
      ss.println("X");
    }
   }
   if (Tempx>=35 && Tempx<37){
    ALARM1=0;
   }
  }
  if (MyTimer<10  && irValue>40000){
  MyTimer++;
Serial.print(NewBPM);
Serial.print("\t");
Serial.println(Tempx);
```

```
    Serial.print("\t");
  Serial.print(temperature,1);
  Serial.print("\t");
  Serial.print(SPO2,1);
    Serial.print("\t");
    Serial.print(SBP);
    Serial.print("\t");
    Serial.print(DBP);
}
while (ss.available()) {
    char inChar = (char)ss.read();
   if (inChar == 'A') {
    MODE=1;
   }
  }
  while (Serial.available()) {
    char inChar1 = (char)Serial.read();
   if (inChar1 == 'A') {
    MODE=1;
   }
  }
  }
```

APPENDIX D
CODING PYTHON FOR PREDICTION

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
```

Data Collection and Processing

```
[ ] # loading the csv data to a Pandas DataFrame
    heart_data = pd.read_csv('/content/heart.csv')
```

```
[ ] # print first 5 rows of the dataset
    heart_data.head()
```

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1

```
[ ] # print last 5 rows of the dataset
    heart_data.tail()
```

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
324	29	1	0	120	129	0	1	139	1	0.0	2	0	1	0
325	26	1	0	97	126	0	1	130	0	0.5	2	0	2	0
326	25	0	0	99	125	0	0	128	1	0.4	1	0	1	0
327	35	1	1	116	135	0	1	140	1	1.8	2	0	2	0
328	32	1	0	119	132	0	1	137	1	0.6	2	0	2	0

```
[ ] # number of rows and columns in the dataset
    heart_data.shape

    (329, 14)
```

```
[ ] # getting some info about the data
    heart_data.info()

    <class 'pandas.core.frame.DataFrame'>
    RangeIndex: 329 entries, 0 to 328
    Data columns (total 14 columns):
     #   Column    Non-Null Count  Dtype
    ---  ------    --------------  -----
     0   age       329 non-null    int64
     1   sex       329 non-null    int64
     2   cp        329 non-null    int64
     3   trtbps    329 non-null    int64
     4   chol      329 non-null    int64
     5   fbs       329 non-null    int64
     6   restecg   329 non-null    int64
     7   thalachh  329 non-null    int64
     8   exng      329 non-null    int64
     9   oldpeak   329 non-null    float64
     10  slp       329 non-null    int64
     11  caa       329 non-null    int64
     12  thall     329 non-null    int64
     13  output    329 non-null    int64
    dtypes: float64(1), int64(13)
    memory usage: 36.1 KB
```

```
[ ] # checking for missing values
    heart_data.isnull().sum()

    age         0
    sex         0
    cp          0
    trtbps      0
    chol        0
    fbs         0
    restecg     0
    thalachh    0
    exng        0
    oldpeak     0
    slp         0
    caa         0
```

```
[ ] # statistical measures about the data
    heart_data.describe()
```

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
count	329.000000	329.000000	329.000000	329.000000	329.000000	329.000000	329.000000	329.000000	329.000000	329.000000	329.000000	329.000000	329.000000	329.000000
mean	52.343465	0.677812	0.905775	129.501520	236.951368	0.136778	0.534954	148.659574	0.331307	1.010638	1.410334	0.689970	2.249240	0.501520
std	11.170544	0.468027	1.018247	18.624519	59.071918	0.344136	0.523380	22.361898	0.471400	1.137381	0.623730	1.003617	0.647609	0.500759
min	24.000000	0.000000	0.000000	81.000000	110.000000	0.000000	0.000000	71.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	45.000000	0.000000	0.000000	119.000000	203.000000	0.000000	0.000000	133.000000	0.000000	0.000000	1.000000	0.000000	2.000000	0.000000
50%	54.000000	1.000000	1.000000	130.000000	235.000000	0.000000	1.000000	150.000000	0.000000	0.600000	1.000000	0.000000	2.000000	1.000000
75%	60.000000	1.000000	2.000000	140.000000	271.000000	0.000000	1.000000	165.000000	1.000000	1.600000	2.000000	1.000000	3.000000	1.000000
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	2.000000	4.000000	3.000000	1.000000

```
[ ] # checking the distribution of Target Variable
    heart_data['output'].value_counts()

    1    165
    0    164
    Name: output, dtype: int64
```

1 --> Defective Heart

0 --> Healthy Heart

Splitting the Features and Target

```
[ ] X = heart_data.drop(columns='output', axis=1)
    Y = heart_data['output']
```

```
[ ] print(X)

       age  sex  cp  trtbps  chol  fbs  restecg  thalachh  exng  oldpeak  slp  \
    0   63    1   3     145   233    1        0       150     0      2.3    0
    1   37    1   2     130   250    0        1       187     0      3.5    0
    2   41    0   1     130   204    0        0       172     0      1.4    2
```

Model Training

Random Forest Claasifier

```
[ ] model = RandomForestClassifier()
```

```
[ ] # training the RandomForest model with Training data
    model.fit(X_train, Y_train)

    RandomForestClassifier()
```

Model Evaluation

Accuracy Score

```
[ ] # accuracy on training data
    X_train_prediction = model.predict(X_train)
    training_data_accuracy = accuracy_score(X_train_prediction, Y_train)
```

```
[ ] print('Accuracy on Training data : ', training_data_accuracy)

    Accuracy on Training data :  1.0
```

```
[ ] # accuracy on test data
    X_test_prediction = model.predict(X_test)
    test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
```

```
[ ] print('Accuracy on Test data : ', test_data_accuracy)

    Accuracy on Test data :  0.8181818181818182
```

```
[ ] model = RandomForestClassifier()
    # training the RandomForestclassifier model with Training data
    model.fit(X_train, Y_train)
    # accuracy on training data
    X_train_prediction = model.predict(X_train)
    training_data_accuracy = accuracy_score(X_train_prediction, Y_train)
    print('Accuracy on Training data : ', training_data_accuracy)
    # accuracy on test data
    X_test_prediction = model.predict(X_test)
    test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
    print('Accuracy on Test data : ', test_data_accuracy)

    input_data = (32,1,0,119,132,0,1,137,1,0.6,2,0,2)

    # change the input data to a numpy array
    input_data_as_numpy_array= np.asarray(input_data)

    # reshape the numpy array as we are predicting for only on instance
    input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

    prediction = model.predict(input_data_reshaped)
    print(prediction)

    if (prediction[0]== 0):
      print('The Person does not have a Heart Disease')
    else:
      print('The Person has Heart Disease')
```

```
Accuracy on Training data :  1.0
Accuracy on Test data :  0.8181818181818182
[0]
The Person does not have a Heart Disease
```