A

DISSERTATION REPORT

ON

Machine Learning based Diabetes Prediction using Decision Tree J48

SUBMITTED TO

MIT ART, DESIGN & TECHNOLOGY UNIVERSITY, PUNE

IN THE PARTIAL FULFLLMENT OF REQUIREMENTS

FOR THE AWARD OF THE DEGREE

OF

MASTER OF TECHNOLOGY

(ELECTRONICS AND COMMUNICATION ENGINEERING)

(MICROELECTRONICS & VLSI DESIGN)

by

Aftab Nadaf

Exam No: MITU21MTMV0001

Under the Guidance of

Prof. Sandip B. Shrote

Department of Electronics & Communication Engineering

MIT School of Engineering & Sciences

MIT Art, Design & Technology University, Pune

Academic Year 2023-2024

Department of Electronics & Communication Engineering

MIT School of Engineering & Sciences

MIT Art, Design & Technology University, Pune

Academic Year 2023-2024

CERTIFICATE

This is to certify that the Project report entitled **“Machine Learning based Diabetes Prediction using Decision Tree J48”** submitted by **Aftab Mehboob Nadaf** is a bonafide work carried out by him, under the supervision of **Prof. Sandip B. Shrote** and submitted towards the partial fulfilment of MIT-ADT University’s requirement, for the award of the degree of Master of Technology in Electronics and Communication (Microelectronics & VLSI Design).

(Prof. Sandip B. Shrote) **Guide**	**(Prof. Dr. Shankar J. Gambhire)** **PG Coordinator**

(Prof. Dr. Shubhangi Joshi)
Head ECE

(Prof. Dr. Virendra V. Shete)
Director
MIT School of Engineering & Sciences, Pune

Sign
External Examiner

Place: Pune
Date:

DECLARATION

We hereby declare that the project entitled "Machine Learning based Diabetes Prediction Using Decision Tree J48" submitted by us to MIT ADT University School of engineering Pune during the degree of Master of Engineering in the department of Electronics & Communication Engineering is a record of bonafide work carried out by us under the guidance of **Prof. Sandip B. Shrote** We further declare that the work reported in this project has been submitted either in part or in full for award of any degree in this institute.

AFTAB NADAF	MITU21MTMV0001

Place: Pune

Date:

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of the task would be put incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crown all the efforts with success.

We whole heartedly thank my project guide **Prof. Sandip B. Shrote (Guide)** for his consistent guidance, expert academic support throughout the project. Without his great concepts & inspiration it would have not been possible.

It is my greatest pleasure to thank **Prof. Dr. Shubhangi Joshi** (**Head of the Department, Electronics & Communication Engineering, MIT School of Engineering**) for providing us the heart full encouragement support and allowing us to work in such a resourceful lab of this esteemed institute and thereby fulfilling one of my dreams.

We thank our postgraduate (PG) coordinator **Prof. Dr. Shankar J Gambhire** for their support and guidance throughout the project and for giving us the inspiration for choosing an appropriate project title

We show our gratitude to our honorable **Director, Prof. Dr. Virendra V. Shete,** for providing all facilities and support.

I thank my parents for their emotional and financial support which they provided me during this project.

I thank all faculties and staff members who directly or indirectly helped us in the completion of this project.

AFTAB NADAF (MITU21MTMV0001)

ABSTRACT

The Project ‘Machine Learning based Diabetes Prediction Using Decision Tree J48’aims to an innovative side in healthcare technology. At its core lies the designed system that has the predictive analysis of machine learning, specifically employing the Decision Tree J48 algorithm. The hardware foundation is based upon the Raspberry Pi Model B+, functioning as the central part. Collaborating seamlessly, a PIC microcontroller facilitates analog-to-digital conversion, ensuring precise data from temperature and glucose sensors. The human-computer interaction is streamlined through a user-friendly interface, featuring a keypad and an LCD display. On the software front, the project has of Microsoft Visual Studio Code as the primary Python development environment. The machine learning library empowered by the scikit-learn module, and the algorithmic implementation of the Decision Tree J48 for predictions. In this both sensor and data set modes has been implemented. Users are empowered to contribute diverse datasets, fostering a personalized dimension to the prediction process. This consists of hardware and software, including by the seamless interaction of sensors, microcontrollers, and machine learning algorithms, showcases the project's approach. Beyond a technological feat, this project addresses a critical need in healthcare early diabetes risk detection.

CONTENTS

List of Figures

List of Abbreviations

Abbreviation	**Full Form**
ADC	Analog to Digital Converter
SK Learn	Sci kit learn
PIC	Peripheral Interface Controller
ML	Machine Learning
LM	Linear Monolithic
RPI	Raspberry Pi
GPIO	General Purpose Input/Output

CHAPTER 1: INTRODUCTION

CHAPTER 1: INTRODUCTION

1.1 INTRODUCTION

The goal of this Project is to create an automated system that can accurately forecast a patient's risk of developing diabetes in the future using a decision tree algorithm. An up-and-coming area of study in data science, machine learning is concerned with how computers might pick up new skills via observation and repetition.

To increase a computer system's performance on a given job, researchers investigate the algorithms and mathematical models known as "machine learning" (ML). To draw inferences and judgments without being explicitly trained to do so, machine learning algorithms construct a mathematical model out of sample data. This data is referred to as training data.

High blood glucose levels are the root cause of diabetes. Symptoms of high blood sugar include the need to urinate more often, drink more water, and eat more frequently. Diabetic complications include vision loss, renal failure, limb loss, cardiovascular disease, and stroke. Our bodies convert the food we consume into glucose, or sugar. When this happens, insulin production from the pancreas should begin. Insulin acts as a key to unlock our cells, allowing glucose entry and subsequent usage of the glucose for energy. However, this method is ineffective for those with diabetes.

The newest member of the Raspberry Pi 3 family is the Model A+. It has the same 64-bit quad-core 1.4 GHz CPU as the Raspberry Pi 3 Model B+, as well as the same dual-band 2.4 GHz and 5 GHz wireless LAN and Bluetooth 4.2/BLE.

The glucosensor and temperature sensor in this project measure the user's actual blood glucose and core body temperature, respectively. This project's primary controller is a Raspberry Pi, which receives data from the sensors through an interface with a PIC microcontroller. The user may choose between an automated and a manual mode through a set of choose switches. The decision tree J48 method relies on values entered by the user through a 4X4 keypad.

A Raspberry Pi computer is the tool for the job. Raspberry Pi's inbuilt 'Linux' is what's responsible for making this happen. The controlling system receives predictions generated by machine learning and the Decision Tree J48 Algorithm.

The temperature sensor and the glucose levels sensor are two of the sensors we're employing in this project. Unfortunately, the analog output from these sensors is incomprehensible to the raspberry pi. To make the analog data usable by the Raspberry Pi, a microcontroller converts it to digital form and subsequently to serial. You may toggle between sensor mode and data set mode using the switch. If you choose the sensor mode, it will read and collect data, and then you may feed that data set into it. The data set is compared to what's already in the raspberry pi's RAM.

To load the data set, SK The learn module includes both the decision tree algorithm and the random forest method by default. In this case, we demploy a decision tree approach using J48, which gives us the illness name as an output, just as you said. Here, we have a 32GB class 10 ScanDisk memory card that comes with the Raspberry Pi. After installing Raspbian Buster 32 bit, the operating system of choice for the Raspberry Pi, we next install the necessary modules.

In this project, we provide two options for carrying out the procedure. One option is to use a mode on the sensor that allows for the collection of dynamic data, up to a certain maximum. Alternatively, you may switch to "data set mode." The 4x4 keypad allows you to choose from 6 different data sets. The Raspberry Pi rereads the data set, compares it to the saved data sets, and displays the result of this decision tree on an LCD screen.

1.2 PROJECT OVERVIEW:

The term "embedded system" refers to a computer system that combines hardware and software to carry out a specific function. Microprocessors and microcontrollers are two of the most common types of embedded devices. Microprocessors are sometimes referred to as "general purpose processors" since all they do is take in data, process it, and then output the results. A microcontroller, on the other hand, does more than just receive data as inputs; it also processes the data, interfaces it with other devices, regulates it, and outputs the outcome.

The "Machine Learning Based Diabetes Prediction using Decision Tree J48 Algorithm" project using the ARM-11 processor is unique in its ability to use the Decision tree algorithm for early prediction of diabetes in a patient. An up-and-coming area of study in data science, machine learning is concerned with how computers might pick up new skills via observation and repetit

1.3 PROBLEM STATEMENT

The rising problem of diabetes has the need for innovative and proactive healthcare solutions. In this context, the project aims to develop an embedded system leveraging the ARM-11 processor and the Raspberry Pi Model B+ to predict the risk of diabetes in individuals. The primary focus is on employing the J48 Decision Tree Algorithm, a powerful machine learning tool, to analyze data collected from temperature and blood glucose sensors. However, the analog nature of the sensor output poses a challenge, necessitating the use of a microcontroller for conversion to digital data.

1.4 FEATURES

1. **Analog-to-Digital Conversion:** The project faces the challenge of translating analog signals from temperature and glucose sensors into a format usable by the Raspberry Pi. A microcontroller plays a crucial role in this conversion, ensuring accurate data representation.
2. **Algorithm Implementation:** The successful integration and implementation of the J48 Decision Tree Algorithm on the Raspberry Pi pose a significant challenge. The algorithm must be optimized to handle patient data effectively and provide accurate predictions of diabetes risk.
3. **User Interface:** A 4x4 keypad serves as the input interface, allowing users to input records and toggle between automated and manual modes. Designing an intuitive and user-friendly interface that accommodates diverse data sets and modes is a critical challenge.
4. **Data Collection Modes:** The project introduces two modes for data collection – dynamic data collection and data set mode and sensor mode. Managing these modes efficiently, including selecting data sets and comparing them to stored records, requires careful consideration and programming.
5. **Machine Learning Application:** The successful integration of machine learning into the embedded system for early diabetes prediction demands meticulous attention. Ensuring that the machine learning model is accurate, reliable, and capable of analyzing diverse patient data is a key challenge.

1.5 OBJECTIVES

1. Machine Learning Application: Develop and integrate a machine learning model using the J48 Decision Tree Algorithm for accurate diabetes risk prediction.
2. Algorithm Implementation: Implement the J48 Decision Tree Algorithm on the ARM-11 processor to process patient data effectively.
3. Sensor Integration: Integrate temperature and blood glucose sensors into the system, overcoming the challenge of analog-to-digital conversion for seamless data acquisition.
4. User Input Interface: Implement a user-friendly interface using a 4x4 keypad, enabling users to input records and navigate between automated and manual modes effortlessly.
5. Mode Selection: Provide a toggle mechanism for users to switch between automated and manual modes, ensuring flexibility in data collection and prediction. Create interactive dashboard to tweak every minute setting to get better result according to user.

1.6 Software: -

i. Microsoft Visual Studio Code (For Python Code)
ii. SCI-KIT Learn Module. (Machine Learning Library)
iii. Raspberry pi OS

1.7 Hardware: -

i. Raspberry PI B+
ii. PIC Microcontroller (8 bit)
iii. Temperature sensor LM35
iv. Glucose sensor
v. LCD display
vi. Keypad
vii. Selection switch

1.8 Brief about software used:

i. Microsoft Visual Studio Code (For Python Code):

Microsoft Visual Studio Code, commonly known as VS Code, is a free and open-source code editor developed by Microsoft. It provides a highly customizable and lightweight environment for writing and debugging code in various programming languages, including Python. VS Code supports features like syntax highlighting, code completion, debugging tools, and extensions that enhance its functionality for different programming tasks.

ii. SCI-KIT Learn Module (Machine Learning Library):

Scikit-learn is a popular machine learning library for Python. It provides simple and efficient tools for data analysis and modeling, including various machine learning algorithms for classification, regression, clustering, and more. In the context of the project, scikit-learn is likely used to implement machine learning algorithms, including the J48 Decision Tree Algorithm.

iii. Raspberry Pi OS:

Raspberry Pi users may install Raspbian, a Linux distribution based on Debian. Raspbian is available in several different releases, such as Raspbian Buster and Raspbian Stretch. Since 2015, the Raspberry Pi Foundation has publicly distributed it as the default OS for their line of Raspberry Pi single-board computers. Mike Thompson and Peter Green developed Raspbian as a side project. Construction on the first phase ended in June 2012.

The OS is currently in its early stages of development. Raspbian has been fine-tuned for the Raspberry Pi line's modest ARM processors.

In the most recent version of Raspbian, PIXEL (Pi Improved X-Window Environment, Lightweight) serves as the default desktop environment. It's based off Open box, a stacking window manager, and a customized version of the LXDE desktop environment. As of the most recent release, the package now includes a lightweight version of Chromium in addition to the standard version, the computer algebra tool Mathematica, and Minecraft Pi.

Different versions of the Raspbian operating system exist. We're employing a variety of operating systems, including Raspbian Jessie.

Raspberry Pi OS, formerly known as Raspbian, is the official operating system for the Raspberry Pi single-board computers. It is a Debian-based Linux distribution optimized for the Raspberry Pi hardware. Raspberry Pi OS provides the necessary environment and tools to run applications, including Python scripts, on the Raspberry Pi. In the project, it serves as the operating system for the Raspberry Pi, facilitating the execution of the diabetes prediction system.

1.9Brief about hardware used:

i. Raspberry Pi B+:

The Raspberry Pi B+ is a model of the Raspberry Pi single-board computer. It features a 64-bit quad-core CPU, wireless LAN, Bluetooth capabilities, and GPIO (General Purpose Input/Output) pins. In the project, the Raspberry Pi B+ likely serves as the main controller for collecting data from sensors, running machine learning algorithms, and making predictions.

ii. PIC Microcontroller (8-bit):

The PIC (Peripheral Interface Controller) microcontroller is an 8-bit microcontroller unit (MCU) developed by Microchip Technology. Microcontrollers are embedded systems that combine processing capabilities with input/output peripherals. In the project, the PIC microcontroller is for to convert analog signal to digital signal and likely interfaces with sensors (temperature sensor, glucose sensor) and communicates with the Raspberry Pi.

iii. Temperature Sensor:

A temperature sensor is a device that measures the ambient temperature of its surroundings. In the project, the temperature sensor is used to measure the user's core body temperature. The data collected from this sensor is likely a crucial input for the diabetes prediction system.

iv. Glucose Sensor:

These sensors are devices that measure specific physiological parameters related to diabetes. The glucose sensor measures blood glucose levels, which is a key factor in diabetes. Current and voltage sensors may be used for various purposes, potentially related to power management, or monitoring specific electrical characteristics.

v. LCD Display:

LCD (Liquid Crystal Display) is a type of flat panel display that uses liquid crystals to modulate light and display information. In the project, the LCD display is likely used to present information, such as the results of the decision tree algorithm or user interface feedback.

vi. Keypad:

Explanation: A keypad is an input device with a set of keys or buttons. In the project, the 4x4 keypad likely allows users to input information or make selections, such as choosing data sets or modes (automated/manual).

vii. Selection Switch:

A selection switch is a type of switch that allows users to toggle between different modes or options. In the project, the selection switch may be used to choose between automated and manual modes or other operational settings. These hardware components collectively form the embedded system designed for the diabetes prediction project. They work in conjunction to collect data, run machine learning algorithms, and provide feedback through the mentioned interfaces and displays.

Fig:1.1 Raspberry Pi 3 B+

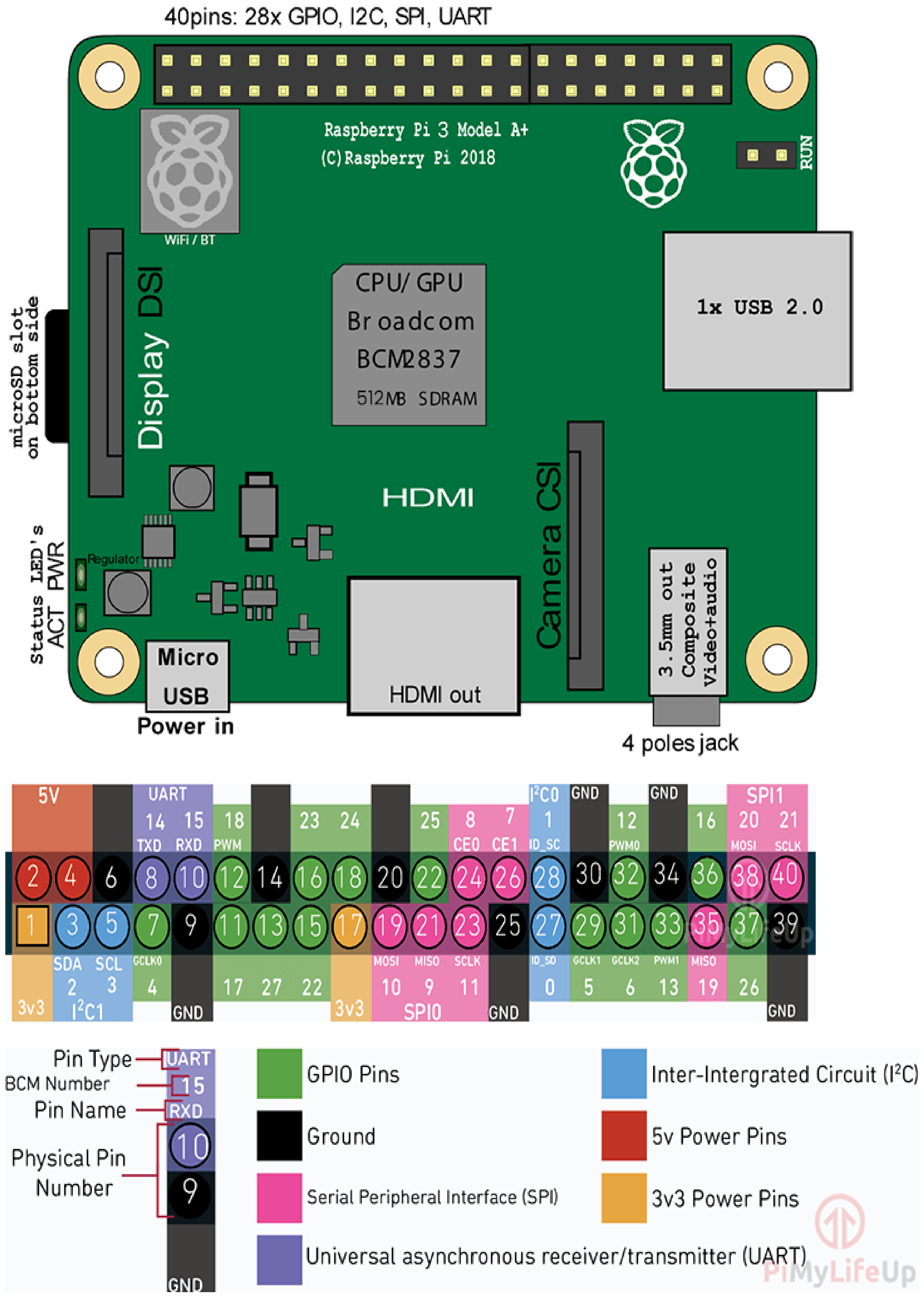


Fig 1.2 Pin Diagram of Raspberry

The Raspberry Pi Model B+ plays a pivotal role in this project, serving as the primary controller that processes to achieve accurate diabetes risk predictions. Here's a detailed look at how the Raspberry Pi is utilized:

1. **Hardware Integration:**
   - **Processing Power:** The Raspberry Pi Model A+ boasts a 64-bit quad-core 1.4 GHz CPU, providing ample processing power to handle data processing, machine learning computations, and system control.
   - **Wireless Connectivity:** With dual-band 2.4 GHz and 5 GHz wireless LAN, the Raspberry Pi facilitates wireless communication, enabling data transfer and interaction with other devices.
   - **Bluetooth Capability:** The Bluetooth 4.2/BLE support enhances connectivity options, potentially allowing the Raspberry Pi to communicate with other Bluetooth-enabled devices or peripherals.
2. **Operating System:**
   - **Linux-Based OS:** The Raspberry Pi runs on a built-in Linux operating system. Linux provides a stable and customizable platform for running applications, managing peripherals, and interfacing with external devices.
3. **Data Reception and Processing:**
   - **Data Acquisition:** The Raspberry Pi receives data from sensors—glucose sensor and temperature sensor—through an interface with a PIC microcontroller. This data includes crucial information related to glucose levels and body temperature.
   - **Analog-to-Digital Conversion:** Analog data from sensors, which is initially incomprehensible to the Raspberry Pi, undergoes conversion to digital format by a microcontroller. This ensures that the data becomes usable by the Raspberry Pi for further processing.
4. **Decision Tree Algorithm Integration:**
   - **Machine Learning Library:** The Raspberry Pi is equipped with the necessary software components, including the Scikit-Learn library. This library incorporates the Decision Tree J48 algorithm, enabling the Raspberry Pi to perform machine learning tasks.
   - **Algorithm Implementation:** The Decision Tree J48 algorithm is applied to the collected data for training and learning patterns related to diabetes risk. The Raspberry Pi oversees the execution of the algorithm.
5. **User Interaction and Display:**
   - **Input Interface:** Users interact with the system through a 4x4 keypad, selecting modes (automated/manual) and providing inputs.
   - **Output Display:** The Raspberry Pi communicates the results of the decision tree algorithm to users through an LCD screen. This ensures a user-friendly interface for displaying predictions and relevant information.

6. **Flexibility and Control:**
   - **Mode Switching:** Users have the flexibility to switch between automated and manual modes using dedicated switches. This feature enhances user control over data collection and prediction processes.
7. **System Integration:**
   - **Interfacing with Modules:** The Raspberry Pi acts as a central hub, interfacing with various hardware components, including sensors, microcontrollers, and the LCD screen. It integrates these elements into a cohesive system for diabetes risk prediction.

CHAPTER 2: LITERATURE SURVEY

CHAPTER 2: LITERATURE SURVEY

1. **"Pattern Recognition and Machine Learning" by Christopher M. Bishop (2006):**

   This comprehensive textbook covers a broad spectrum of topics in pattern recognition and machine learning. It introduces fundamental concepts such as Bayesian networks, neural networks, and support vector machines. The book emphasizes a probabilistic approach to machine learning, providing a solid theoretical foundation. Practical examples and case studies are included to illustrate the application of machine learning algorithms.

2. **"Machine Learning" by Tom M. Mitchell (1997):**

   In Mitchell's book provides a comprehensive introduction to machine learning, covering topics from concept learning to reinforcement learning. It delves into the theoretical foundations of machine learning algorithms and their practical applications. The book emphasizes the importance of learning from data and adapting to new information. Case studies and examples help readers understand the real-world implications of machine learning.

3. **"Induction of Decision Trees" by J. Ross Quinlan (1986):**

   Quinlan's paper introduces the concept of decision trees and the C4.5 algorithm for inducing them. It outlines the methodology for constructing decision trees from labeled training data. The paper discusses the use of entropy and information gain as measures for selecting the best attributes for splitting the data. Practical examples and experiments illustrate the effectiveness of decision trees in classification tasks.

4. **"The Elements of Statistical Learning: Data Mining, Inference, and Prediction" by Trevor Hastie, Robert Tibshirani, and Jerome Friedman (2009):**

   "The Elements of Statistical Learning" is a comprehensive textbook that covers various aspects of statistical learning, including decision trees. The book explores the theory and application of decision trees for predictive modeling. It introduces key concepts such as CART (Classification and Regression Trees) and discusses ensemble methods like bagging and boosting.

   Practical examples and case studies illustrate the application of decision trees in real-world scenarios.

5. **"Machine Learning Techniques for Diabetes Prediction" by K. Srinivasan et.al. (2017):**

   This paper delves into the application of various machine learning techniques for predicting diabetes. It likely explores different algorithms, methodologies, and features used in the prediction process. The focus may include the use of relevant health data and variables in developing predictive models. The paper discusses the implications of machine learning in the context of diabetes management and healthcare

6. **"Scalable and Accurate Deep Learning with Electronic Health Records" by A. Rajkomar et.al. (2018):**

   This paper likely focuses on the application of deep learning techniques in analyzing electronic health records. It may discuss the scalability and accuracy of deep learning models when applied to large-scale healthcare datasets. The paper might explore the challenges and opportunities associated with using electronic health records for training deep learning models. Considerations for model interpretability and real-world implications may be discussed.

7. **"Deep Patient: An Unsupervised Representation to Predict the Future of Patients from Electronic Health Records" by R. Miotto et.al. (2018):**

   This paper likely focuses on the development of unsupervised representations for predicting future health outcomes from electronic health records. It may discuss the architecture and methodology of the "Deep Patient" model, emphasizing the unsupervised learning aspect. The paper might explore the potential applications of unsupervised deep learning in patient risk prediction. Considerations for ethical and privacy-related aspects of utilizing electronic health records may be discussed.

8. **"Ensemble Selection from Libraries of Models" by R. Caruana et.al. (2004):**

   This paper likely explores ensemble selection strategies, where models from a library are selectively combined to form an ensemble. It may discuss the motivation for ensemble selection and the criteria used to choose models for inclusion in the ensemble. The paper could present experimental results showcasing the effectiveness of ensemble selection. Considerations for model diversity and performance evaluation may be addressed.

9. **"Diabetes Prediction using Ensemble Learning based on Decision Trees" by S. Das and D. Saha (2016):**

   This paper likely focuses on the development of a diabetes prediction system using ensemble learning, specifically based on decision trees. It may discuss the design and implementation of the ensemble learning approach, including considerations for decision tree models. The paper could present experimental results, showcasing the effectiveness of the proposed system in predicting diabetes. Considerations for data preprocessing, feature selection, and model evaluation may be discussed.

10. **"Learning from Imbalanced Data" by H. He and E. A. Garcia (2009):**

    This paper likely focuses on the challenges posed by imbalanced data in machine learning and proposes techniques for learning from such data. It may discuss methods to address the bias introduced by imbalanced classes and how these methods impact model training. The paper could cover various strategies for improving the performance of models trained on imbalanced datasets. Considerations for model evaluation in the context of imbalanced data may also be discussed**.**

11. **"SMOTE: Synthetic Minority Over-sampling Technique" by N. V. Chawla et.al. (2002):**

    This paper introduces the Synthetic Minority Over-sampling Technique (SMOTE) as a method for addressing imbalanced datasets. It likely details the SMOTE algorithm and how it generates synthetic samples for the minority class to balance the class distribution. The paper may discuss experimental results showcasing the effectiveness of SMOTE in improving model performance on imbalanced data. Considerations for the impact of synthetic data on model generalization may also be addressed.

CHAPTER 3: METHODOLOGIES

CHAPTER 3: METHODOLOGIES

3.1 Machine Learning Methodology:

- **Objective:** The primary goal is to predict the risk of diabetes in individuals using machine learning techniques.
- **Algorithm:** The J48 Decision Tree Algorithm is employed for predicting diabetes. This algorithm recursively splits the dataset based on features to create a tree-like structure, facilitating decision-making.
- **Training Data:** The machine learning model is trained using a dataset that includes information about patients' temperature and blood glucose levels. The dataset likely comprises both positive (patients with diabetes) and negative (patients without diabetes) instances.
- **Feature Selection:** Relevant features, such as temperature and glucose levels, are selected as input variables for the decision tree algorithm. Feature selection is crucial for building an effective predictive model.
- **Data Collection:** Data is collected from sensors (temperature sensor, glucose sensor) to obtain real-time information about the patient's health status. The collected data is used to make predictions and refine the decision tree model.
- **Model Evaluation:** The performance of the machine learning model is likely evaluated using metrics such as accuracy, precision, recall, and F1 score. This evaluation ensures that the model provides reliable predictions.
- **Data Preprocessing:** The project may involve preprocessing steps such as handling missing data, normalizing features, and addressing imbalances in the dataset to enhance the performance of the machine learning model.
- **Scikit-Learn Module:** The scikit-learn machine learning library is used to implement the J48 Decision Tree Algorithm. Scikit-learn provides tools for data preprocessing, model training, and evaluation, making it a valuable resource for the machine learning aspect of the project.

3.2 Embedded System Methodology:

1. **Objective:** The embedded system is designed to collect real-time health data, process it using machine learning, and provide predictions through a user-friendly interface.
2. **Raspberry Pi as the Controller:** The Raspberry Pi B+ serves as the central controller, responsible for receiving data from sensors, running the machine learning model, and displaying results. The Raspberry

Pi interacts with both hardware (sensors, LCD, keypad) and software components.

3. **PIC Microcontroller Interface:** The PIC microcontroller acts as an interface between the Raspberry Pi and analog sensors. It converts analog sensor data to digital form and facilitates communication with the Raspberry Pi, ensuring that the data is in a usable format.
4. **User Interaction:** The system allows user interaction through a 4x4 keypad and a selection switch. Users can choose between automated and manual modes, select data sets, and potentially input additional information through the keypad.
5. **LCD Display:** The LCD display serves as the output interface, providing users with the results of the decision tree algorithm. It enhances user interaction by displaying relevant information in a readable format.
6. **Mode Switching:** The system provides two modes of operation - automated and manual. The mode can be toggled using a selection switch. In automated mode, real-time data is collected, and predictions are generated. In manual mode, users may choose from pre-existing data sets for analysis.
7. **Real-time Data Acquisition:** The system supports real-time data acquisition from sensors, allowing users to monitor and analyze health data dynamically. The collected data is then used to make predictions using the decision tree algorithm.
8. **Integration of Hardware and Software:** The project integrates hardware components (Raspberry Pi, PIC microcontroller, sensors) with software components (machine learning algorithm, user interface) to create a comprehensive embedded system for diabetes prediction.

3.3 Data Set Selection and Evaluation:

1. **Kaggle**:

   Kaggle is an online platform for data science and machine learning competitions. It hosts a diverse range of datasets and challenges, allowing data scientists and machine learning practitioners to participate in competitions, collaborate, and showcase their skills.

2. **Training Dataset:**

   The training dataset is obtained from Kaggle, indicating that it originates from one of the competitions or datasets available on the platform. Composition, A training dataset typically consists of a set of labeled examples, where each example includes input features (such as temperature and glucose levels in this project) and corresponding labels indicating whether a patient has diabetes or not. Purpose is the training dataset is used to teach the machine learning model how to make predictions. The model learns patterns and relationships in the data by adjusting its parameters during the training process.

3. **Features and Labels:**

   Features are the input variables used to make predictions, and labels are the corresponding outcomes (in this case, whether a patient has diabetes). The model learns to map features to labels based on the patterns present in the training data

4. **Importing to scikit-learn:**

   Data Loading: Scikit-learn provides various functions for loading and working with datasets. The load_... or fetch_... functions in scikit-learn, such as load_diabetes or load_digits, are commonly used to load built-in datasets. However, for datasets from external sources like Kaggle, you would need to load them manually into a format that scikit-learn understands.

5. **Data Preparation:**

   Once the dataset is loaded, it needs to be preprocessed to ensure it aligns with the requirements of the machine learning algorithm. This may involve handling missing values, scaling features, and splitting the data into training and testing sets.

6. **Machine Learning Model:**

   Algorithm: In this project, the J48 Decision Tree Algorithm from the scikit-learn library is used. This algorithm constructs a decision tree based on the patterns in the training data, enabling the model to make predictions about diabetes risk.

7. **Training Process:**

   The model is trained on the Kaggle-derived training dataset using the fit method in scikit-learn. During training, the algorithm optimizes its internal parameters to minimize the difference between predicted outcomes and actual labels in the training data.

8. **Evaluation:**

   Testing Data, in addition to the training dataset, there is likely a separate testing dataset used to evaluate the model's performance. The testing dataset is distinct from the training data, helping to assess how well the model generalizes to new, unseen data.

9. **Metrics:**

   Evaluation metrics, such as accuracy, precision, recall, and F1 score, are used to measure the performance of the model on the testing dataset. These metrics provide insights into how well the model predicts diabetes risk.

10. Data Range:

Defined six distinct types of outcomes based on ranges of glucose levels. Each outcome corresponds to a specific range, and these ranges are used to categorize individuals into different health states. Here's an elaboration on each outcome and its associated glucose level range in Milligrams per decilitre (mg/dL):

i. **Excellent (0-90):**
Individuals falling within the glucose level range of 0 to 90 mg/dL are classified as "Excellent." This range typically represents a healthy blood sugar level.

ii. **Normal (90-120):**
The "Normal" category encompasses individuals with glucose levels ranging from 90 to 120 mg/dL. This range is considered normal and indicates a stable blood sugar level.

iii. **Borderline (120-130):**
Individuals with glucose levels in the range of 120 to 130 mg/dL are classified as "Borderline." This range suggests a potential deviation from normalcy, warranting attention.

iv. **Prediabetes (130-150):**
The "Prediabetes" category includes individuals with glucose levels ranging from 130 to 150 mg/dL. This range signifies an elevated blood sugar level that may precede the onset of diabetes.

v. **Diabetes (150-180):**
Individuals with glucose levels in the range of 150 to 180 mg/dL are categorized as having "Diabetes." This range indicates a higher and potentially unhealthy blood sugar level requiring medical attention.

vi. **Dangerous (180 and above):**
The "Dangerous" category includes individuals with glucose levels equal to or exceeding 180 mg/dL. This range represents a critical condition, and individuals falling into this category may be at risk of severe health complications. Urgent medical intervention is typically required.

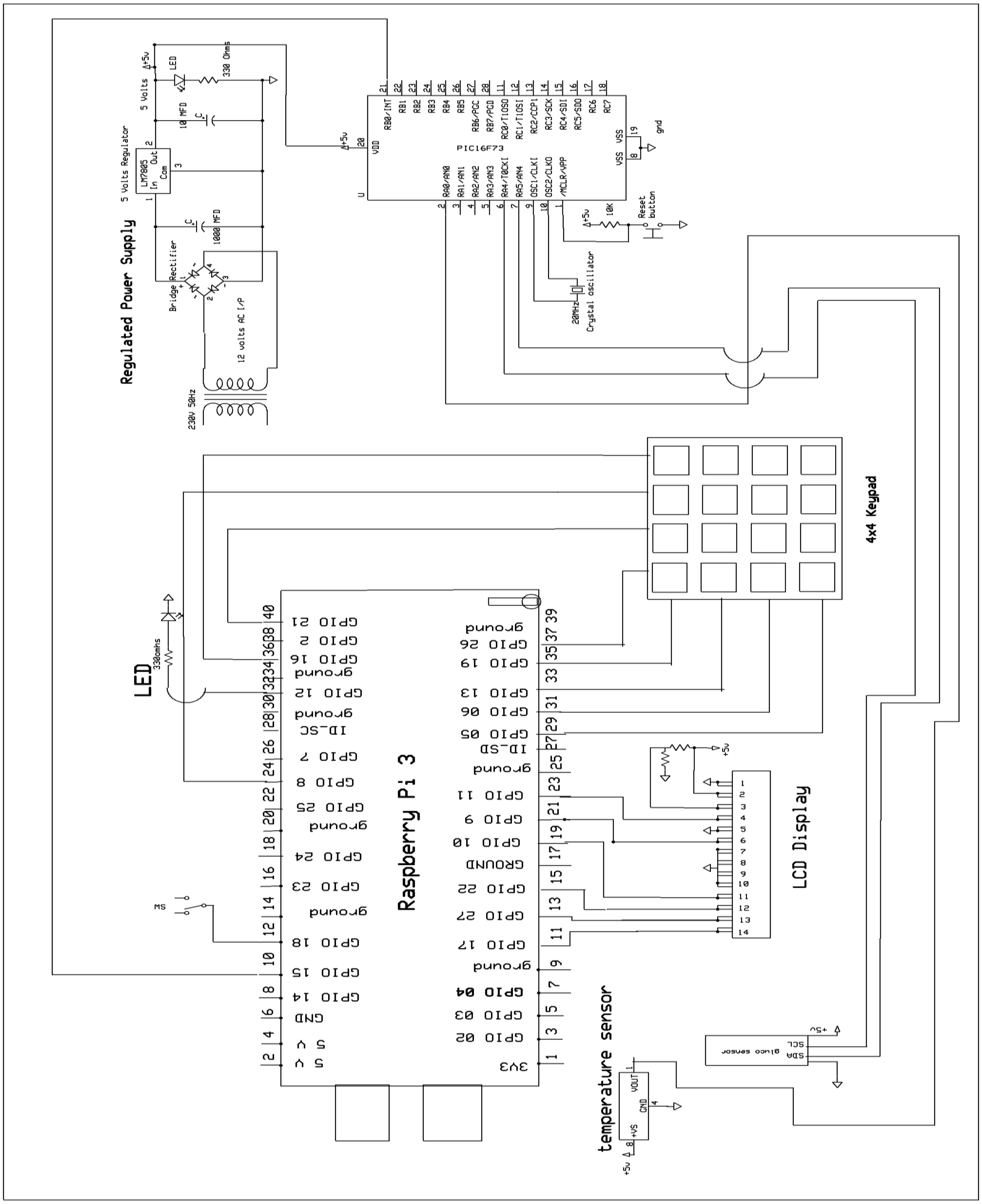


FIG 3.1: Complete Circuit Implementation

3.4 Brief about hardware connections:

1. Glucose Sensor:

A glucose sensor is a specialized device designed to measure the concentration of glucose (sugar) in a solution, typically in bodily fluids like blood. In the context of your diabetes prediction system, the glucose sensor is instrumental in obtaining real-time data about the user's blood glucose levels. Why is the Glucose Sensor Used? Diabetes Monitoring: High blood glucose levels are a key indicator of diabetes. By continuously monitoring glucose levels, the system can provide insights into the user's health status. Early Detection: Regular monitoring allows for early detection of abnormal glucose levels, enabling proactive measures and intervention before the onset of severe complications. Machine Learning Training Data, the glucose sensor provides crucial data for training the machine learning model. The historical patterns of glucose levels contribute to the model's ability to predict future trends and identify potential diabetes risk factors. Decision Making, Glucose levels are a critical factor in the decision-making process of the decision tree algorithm. The system uses this information to assess the user's risk of developing diabetes.

2. Temperature Sensor LM35:

A temperature sensor is a device that measures the ambient temperature of its surroundings. In this project, the temperature sensor is likely employed to monitor the user's core body temperature.

Why is the Temperature Sensor Used?

Health Monitoring for Core body temperature is a vital sign that reflects the body's overall health. Abnormalities in temperature can indicate various health conditions, including infections or metabolic issues. Correlation with Diabetes is a connection between diabetes and changes in body temperature. Monitoring temperature can provide additional context to the machine learning algorithm, contributing to a more comprehensive health assessment.

Decision Tree Algorithm Input is like the glucose sensor, the temperature sensor's data becomes an input for the decision tree algorithm. It helps the system make more informed predictions about the user's health status.

Enhancing Predictive Accuracy is Combining multiple health parameters, including temperature, improves the overall accuracy of the predictive model. It allows for a more holistic evaluation of the user's health condition.

3. PIC Microcontroller (8-bit):

   Connection with Raspberry Pi is for Establish communication between the PIC microcontroller and the Raspberry Pi through a serial interface, such as UART. This connection enables data exchange between the microcontroller and the Raspberry Pi, facilitating seamless collaboration.

4. ADC Conversion:

   Leverage the microcontroller's ADC capabilities to convert analog sensor data (glucose and temperature) into digital form for processing. This step is crucial for transforming real-world sensor measurements into a format that the machine can comprehend.

5. Mode Switch and Keypad:

   Connect the mode switch and 4x4 keypad to appropriate General-Purpose Input/Output (GPIO) pins on the microcontroller. These connections allow the microcontroller to receive user input and control the system's operation mode. Ensure proper voltage levels compatibility between the microcontroller and Raspberry Pi. This compatibility is essential for maintaining stable communication between the two components.

6. Raspberry Pi:

   Connection with PIC Microcontroller:

   Establish a serial communication link between the Raspberry Pi and the PIC microcontroller using GPIO pins. This connection enables the Raspberry Pi to receive data from the microcontroller and vice versa, fostering data exchange.

7. LCD Screen:

   Connect the LCD screen to the Raspberry Pi using GPIO pins or any supported display interface, such as SPI or I2C. This connection allows the Raspberry Pi to convey information to users through the display screen. Ensure proper voltage levels compatibility between the Raspberry Pi and the PIC microcontroller. Additionally, install the necessary libraries or drivers on the Raspberry Pi to enable seamless interfacing with the LCD screen.

8. Keypad:

   Connection with PIC Microcontroller:

   Connect the 4x4 keypad to appropriate GPIO pins on the PIC microcontroller. These connections enable the microcontroller to receive user input from the keypad, facilitating user interaction with the system. Ensure proper voltage levels compatibility. Implement keypad scanning routines in the microcontroller code to effectively interpret and respond to user inputs.

9. Power Supply:

   Power for Sensors and Microcontroller:

   Provide a suitable power supply for the sensors, PIC microcontroller, and any supporting components.

   This ensures that all components receive the required power for accurate and consistent operation.

10. Power for Raspberry Pi:

    Ensure the Raspberry Pi has a stable power supply. It might be powered separately or through the same power source as the sensors. A stable power supply is essential for the reliable performance of the Raspberry Pi.

3.5 Decision Tree J48:

Decision Tree J48, also known as C4.5, is a popular algorithm used in machine learning for both classification and regression tasks. Developed by Ross Quinlan, it builds a tree-like structure where each internal node represents a decision based on the values of a particular feature, and each leaf node represents the predicted outcome. The algorithm is particularly well-suited for classification problems.

How J48 Works:

1. Node Splitting:

J48 employs a top-down, recursive approach. It starts with the entire dataset and selects the feature that best splits the data into subsets. This process continues recursively for each subset until a stopping criterion is met.

2. Attribute Selection:

At each node, the algorithm decides which attribute (feature) to use for splitting. It selects the attribute that maximizes the information gain or minimizes the impurity of the subsets, ensuring the resulting branches are as pure as possible.

3. Stopping Criteria:

The recursive splitting continues until a predefined stopping condition is met. This could be reaching a certain depth of the tree, achieving a minimum number of instances per leaf, or other criteria to prevent overfitting.

4. Leaf Node Prediction:

Once the tree is constructed, new instances traverse the tree from the root to a leaf node. The majority class in the leaf node is then assigned as the predicted outcome.

5. Application in the Diabetes Prediction System:

In this project, the Decision Tree J48 algorithm is used for predicting the risk of diabetes based on the input data, which includes glucose levels and temperature. Here's how it fits into the system:

6. Data Collection:

Glucose and temperature data are collected using sensors connected to the PIC microcontroller.

7. Data Processing:

The microcontroller processes the analog sensor data and converts it into a digital format. It also handles user inputs from the keypad and mode switch.

8. Machine Learning Model Training:

The digital sensor data is then sent to the Raspberry Pi, where the Scikit-Learn library, including the Decision Tree J48 algorithm, is utilized. The collected data is split into training and testing sets.

9. Decision Tree Training:

The Decision Tree J48 algorithm is applied to the training set to learn the patterns and relationships between glucose levels, temperature, and the likelihood of diabetes.

10. Prediction:

When a new set of data is received (glucose and temperature levels), the Decision Tree J48 algorithm is used to predict the likelihood of diabetes. The result is displayed on the LCD screen.

11. User Interaction:

The 4x4 keypad allows users to interact with the system, choosing between automated and manual modes, providing input, and initiating data collection or prediction.

Terminologies Related to Decision Tree Algorithms

I. Root Node: Separate similar nodes are created from this one. It's a symbol for the full data set.

II. Splitting: It's the act of separating a node into several new nodes.

III. Interior Nodes: Each one stands for a distinct attribute-checking procedure.

IV. Branches: They have access to the results of the examinations.

V. Leaf Nodes: When further division is impossible, the resulting nodes are known as leaf nodes.

VI. Parent and Child Nodes: A parent node is the node that serves as the starting point for the creation of child nodes. Child nodes are the sub-nodes that branch out from the main node.

- Flowchart of Decision Tree J48:

```
                                   Start
                                     |
                                [Root Node]
                                     |
                       Is Glucose Level ≤ Threshold?
                                 /         \
                               Yes          No
                                |           |
                 [Leaf: Class 0] [Node: Split on Temperature]
                                  /        \
          Is Temperature ≤ Threshold?  Is Temperature ≤ Threshold?
                      /          \          /          \
                    Yes          No       Yes           No
                     |           |         |            |
           [Leaf: Class 1] [Leaf: Class 0] [Leaf: Class 1] [Leaf: Class 0]
```

Explanation:

Root Node:

The first decision is made based on whether the glucose level is below a certain threshold.

Node: Split on Temperature:

If the glucose level is below the threshold, the next decision is based on the temperature.

If the glucose level is not below the threshold, the process moves to a different set of decisions.

Leaf Nodes:

Leaf nodes represent the final outcomes or classes. In this simplified example, Class 0 and Class 1 are the predicted outcomes.

Decision Criteria:

Each decision node tests a specific feature (e.g., glucose level, temperature) and determines the path to follow based on the feature's value.

Thresholds:

Thresholds are values that determine which branch to follow at decision nodes. These values are determined during the training phase based on the dataset.

Leaf Predictions:

The leaf nodes contain the predicted outcomes. In a binary classification scenario (Class 0 or Class 1), the leaf nodes represent the final prediction.

- Example on sample data set:

Sr. No	Glucose Level	Body Temperature	Diabetes Class
1	120	98.6	0
2	180	98.2	1
3	150	99.0	0
4	130	97.5	0
5	160	98.8	1
6	140	98.0	0
7	190	98.7	1
8	145	98.5	0
9	170	99.2	1
10	155	98.3	0

- Flowchart Based on Above Example:

```
                 [Root Node]
                      |
         Is Glucose Level ≤ 150?
         /                     \
      Yes                        No
       |                        |
[Leaf: No Diabetes]   [Node: Split on Temperature]
                                /                \
           Is Temperature ≤ 98.5?            Is Temperature ≤ 98.5?
           /                   \            /                  \
         Yes                    No         Yes                  No
          |                    |           |                   |
    [Leaf: Diabetes]   [Leaf: No Diabetes] [Leaf: Diabetes]   [Leaf: Diabetes]
```

Explanation:

1. Root Node:

The first decision is made based on whether the glucose level is less than or equal to 150.

2. Node: Split on Temperature:

If the glucose level is below or equal to 150, the next decision is based on body temperature.

If the glucose level is above 150, the decision tree predicts "Diabetes" without considering temperature.

3. Leaf Nodes:

The leaf nodes represent the final predictions.

For example, if glucose level is below 150 and temperature is below 98.5, the prediction is "Diabetes."

4. Thresholds:

Thresholds (150 for glucose, 98.5 for temperature) are determined during the training phase based on the dataset.

3.6 PROPOSED BLOCK DIAGRAM

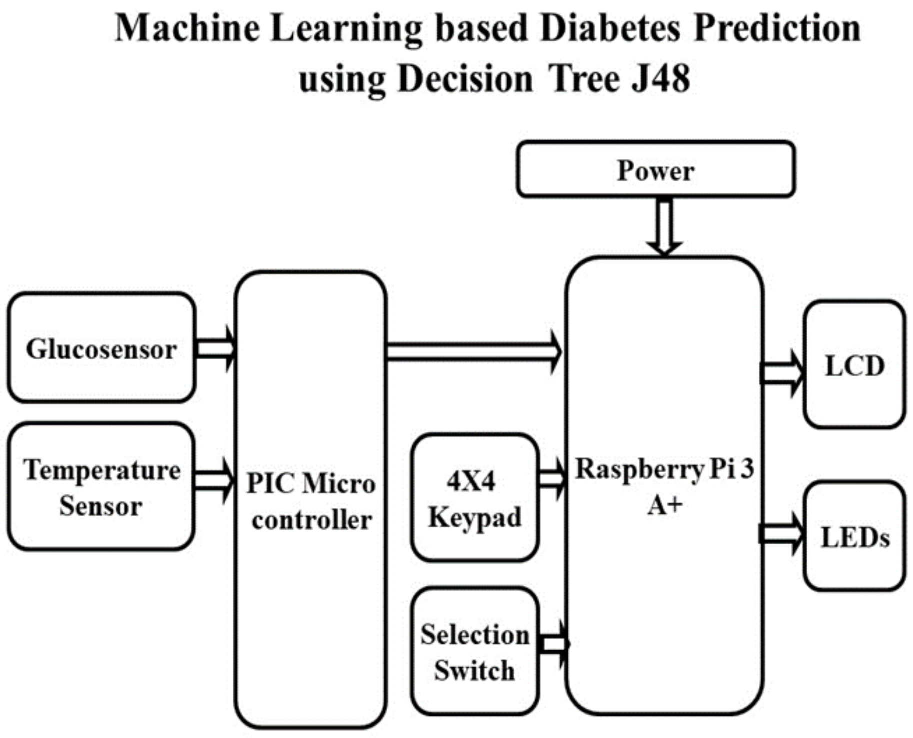


Fig 3.5.1 Proposed block diagram of the kit

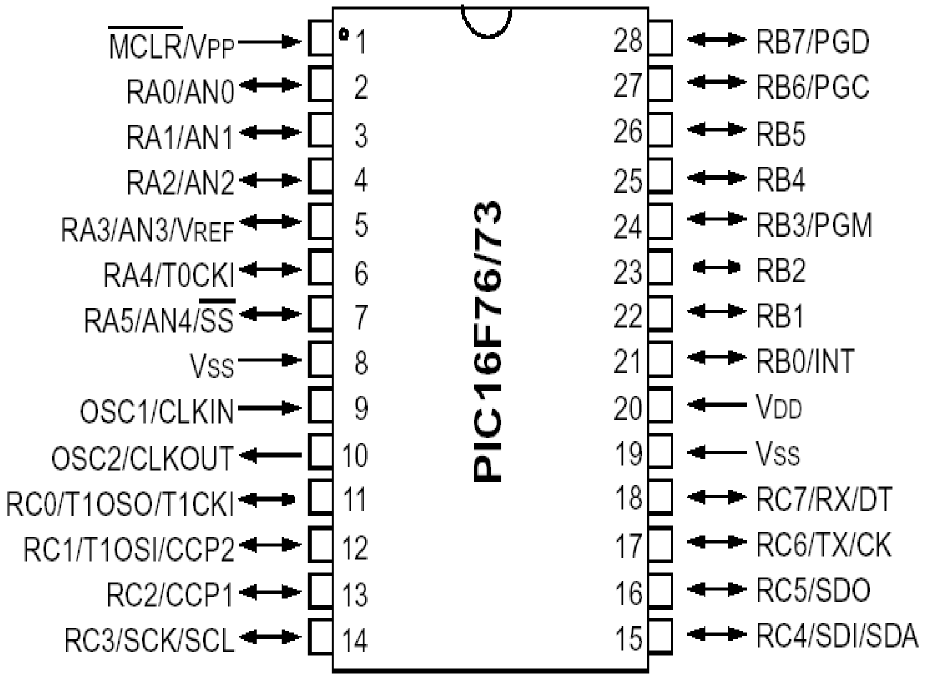
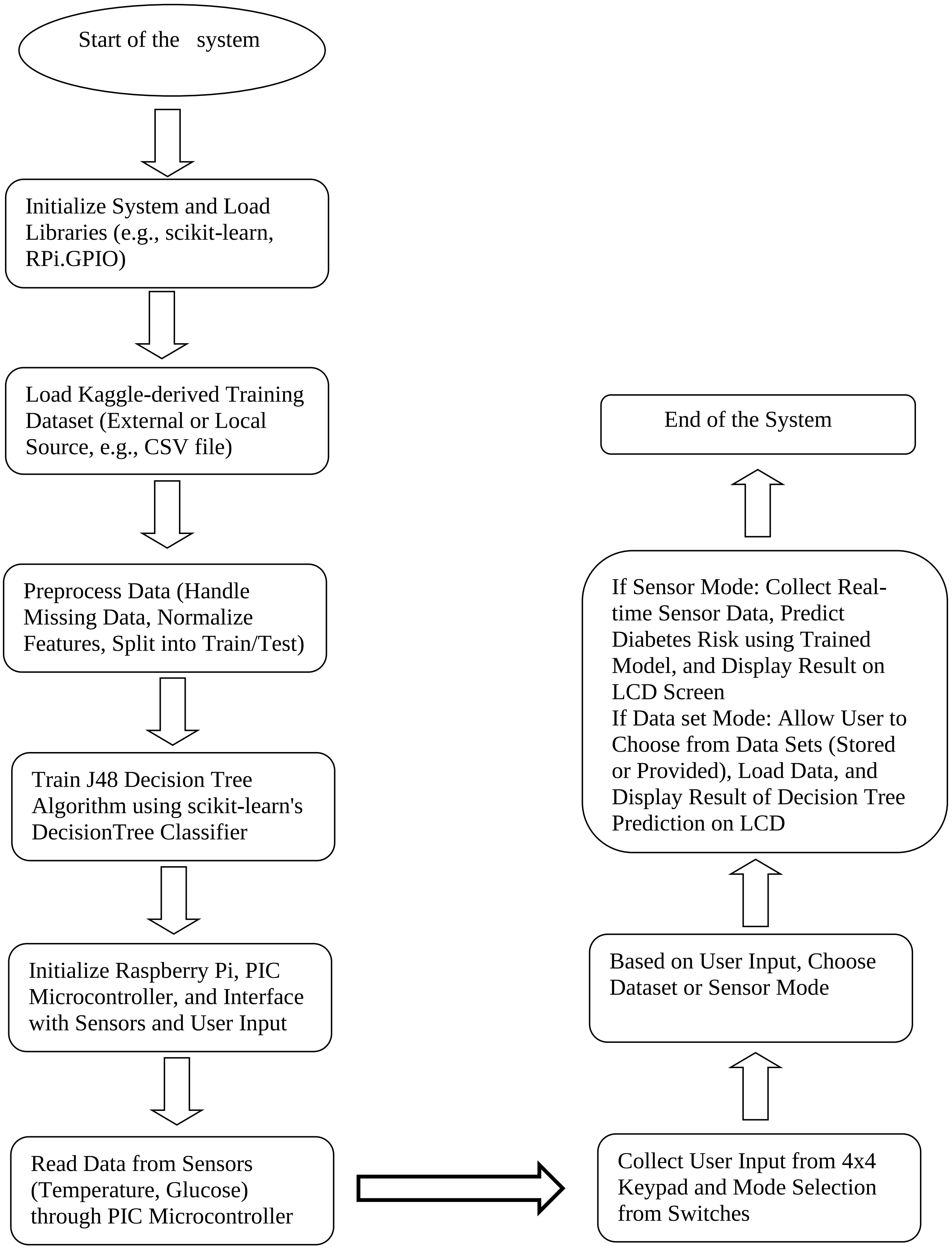


Fig 3.5.2 Pin Description of PIC Microcontroller

3.7 Flowchart of worflow:


Start of the system
Initialize System and Load Libraries (e.g., scikit-learn, RPi.GPIO)
Load Kaggle-derived Training Dataset (External or Local Source, e.g., CSV file)
Preprocess Data (Handle Missing Data, Normalize Features, Split into Train/Test)
Train J48 Decision Tree Algorithm using scikit-learn's DecisionTree Classifier
Initialize Raspberry Pi, PIC Microcontroller, and Interface with Sensors and User Input
Read Data from Sensors (Temperature, Glucose) through PIC Microcontroller
Collect User Input from 4x4 Keypad and Mode Selection from Switches
Based on User Input, Choose Dataset or Sensor Mode
If Sensor Mode: Collect Real-time Sensor Data, Predict Diabetes Risk using Trained Model, and Display Result on LCD Screen
If Data set Mode: Allow User to Choose from Data Sets (Stored or Provided), Load Data, and Display Result of Decision Tree Prediction on LCD
End of the System

3.8 Methodology for Prediction

1. **Data Collection:**
    - Gather data from patients, including features such as glucose levels and body temperature.

2. **Data Preprocessing:**
    - Handle missing data: Impute or remove missing values.
    - Encode categorical variables if any.
    - Normalize or scale numerical features if needed.

3. **Dataset Splitting:**
    - Split the dataset into training and testing sets. The training set is used to train the Decision Tree J48 model, and the testing set is used to evaluate its performance.

4. **Feature Selection:**
    - The Decision Tree J48 algorithm automatically selects the most informative features during the training phase based on information gain.

5. **Model Training:**
    - Feed the training dataset into the Decision Tree J48 algorithm.
    - The algorithm builds a decision tree by recursively selecting features and thresholds that maximize information gain.

6. **Model Evaluation:**
    - Evaluate the model's performance on the testing dataset to assess its predictive accuracy.
    - Common evaluation metrics include accuracy, precision, recall, and F1 score.

7. **Fine-Tuning (Optional):**
    - Adjust hyperparameters or perform pruning to optimize the model's performance.
    - Pruning helps prevent overfitting by removing unnecessary branches from the decision tree.

8. **Prediction on New Data:**
   - Once the model is trained and evaluated, it can be used to make predictions on new, unseen data.

9. **Integration with Raspberry Pi:**
   - Implement the Decision Tree J48 model on the Raspberry Pi, enabling it to process real-time data from the glucose and temperature sensors.

10. **User Interaction:**
    - Allow users to input data through the 4x4 keypad or use the automated mode with dynamic data collection.

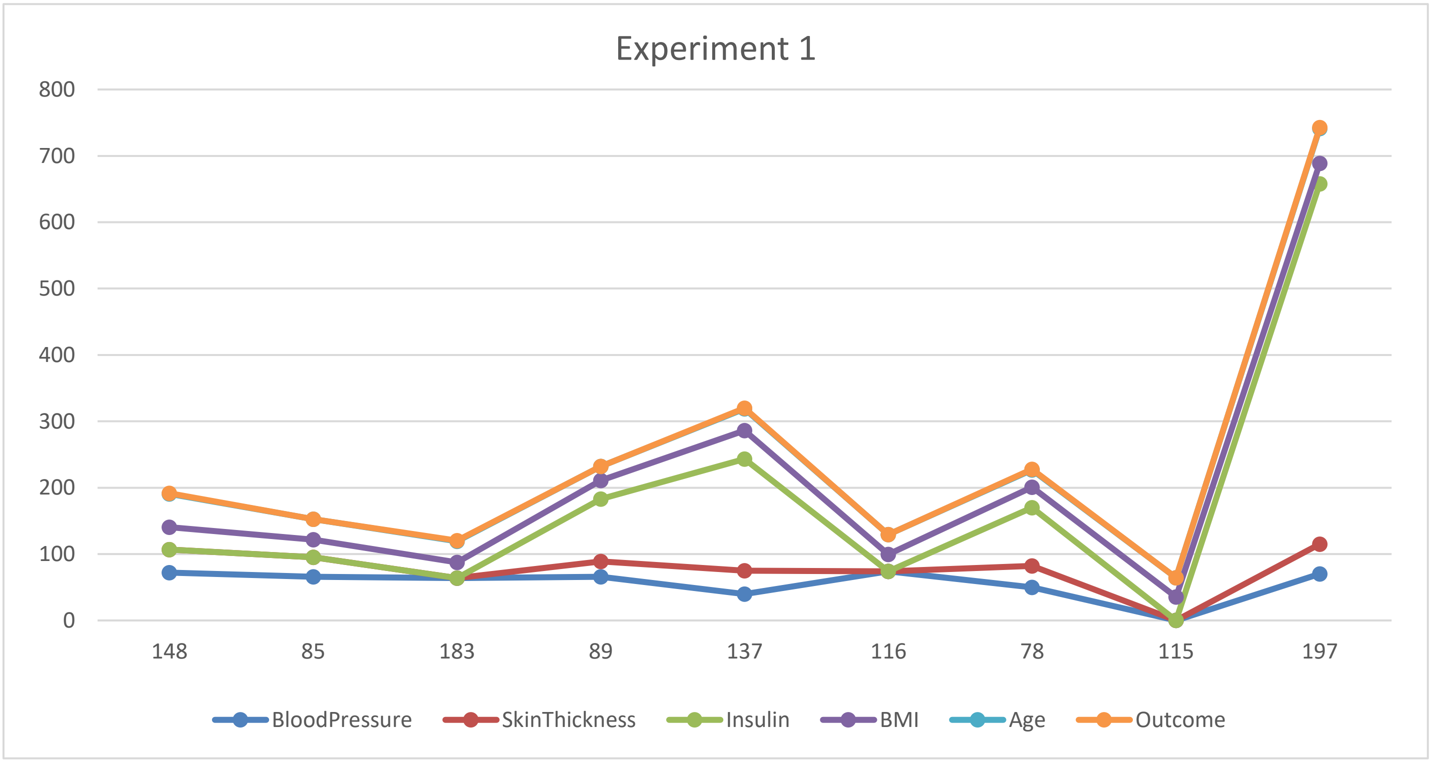
11. **Prediction Display:**
    - Display the prediction results on the LCD screen connected to the Raspberry Pi.

12. **Toggle Between Modes:**
    - Implement the functionality to toggle between Data set and Sensor modes using the selection switch.

- For Example, Experiment 1

Glucose	BloodPressure	SkinThickness	Insulin	BMI	Age	Outcome
148	72	35	0	33.6	50	1
85	66	29	0	26.6	31	0
183	64	0	0	23.3	32	1
89	66	23	94	28.1	21	0
137	40	35	168	43.1	33	1
116	74	0	0	25.6	30	0
78	50	32	88	31	26	1
115	0	0	0	35.3	29	0
197	70	45	543	30.5	53	1

- Plotted Graph:



experiment 1 were created.

3.9 Python Code:

Python code is essential for implementing the machine learning aspect of this project, particularly for training the Decision Tree J48 model and integrating it with the Raspberry Pi. Here's a high-level overview of the potential components that might involve Python code:

1. Data Preprocessing:

Python code may be required to handle missing data, encode categorical variables, and normalize or scale numerical features.

2. Machine Learning Model Training:

The scikit-learn library in Python is commonly used for machine learning tasks. You would use it to instantiate and train the Decision Tree J48 model on the preprocessed dataset.

3. Integration with Raspberry Pi:

Python scripts on the Raspberry Pi can be used to establish communication with the PIC microcontroller and control the sensors.

4. Real-time Prediction:

Python code on the Raspberry Pi can process real-time data from sensors, invoke the trained model for predictions, and display the results on the LCD screen.

5. User Interaction (4x4 Keypad):

Python code can handle the input from the 4x4 keypad, allowing users to choose between automated and manual modes or input data.

6. Python Code:

```
from pad4pi import rpi_gpio
import keras

import RPi.GPIO as GPIO
import time, sys, os

import serial

import datetime
from time import sleep

from datetime import datetime

ser = serial. Serial("/dev/ttyS0",9600, timeout = 1)
```

```
KEYPAD = [
        ["1","2","3","A"],
        ["4","5","6","B"],
        ["7","8","9","C"],
        ["*","0","#","D"]
]

COL_PINS = [19,13,6,5] # BCM numbering
ROW_PINS = [16,20,21,26] # BCM numbering


# Define GPIO to LCD mapping
LCD_RS = 11
LCD_E  = 9
LCD_D4 = 10
#LCD_D5 = 24
#LCD_D6 = 23
#LCD_D7 = 18
LCD_D5 = 22
LCD_D6 = 27
LCD_D7 = 17

led = 12

sw = 18

ap = argparse.ArgumentParser()
ap.add_argument("-p", "--disease-predictor", required=True,
            help="path to predictor")

args = vars(ap.parse_args())

factory = rpi_gpio. KeypadFactory()

keypad = factory.create_keypad(keypad=KEYPAD, row_pins=ROW_PINS, col_pins=COL_PINS)

print("Init DONE...")

GPIO.setmode(GPIO.BCM)       # Use BCM GPIO numbers
GPIO.setup(LCD_E, GPIO.OUT)  # E
GPIO.setup(LCD_RS, GPIO.OUT) # RS
GPIO.setup(LCD_D4, GPIO.OUT) # DB4
GPIO.setup(LCD_D5, GPIO.OUT) # DB5
GPIO.setup(LCD_D6, GPIO.OUT) # DB6
GPIO.setup(LCD_D7, GPIO.OUT) # DB7

GPIO.setup(led, GPIO.OUT) # DB7
GPIO.setup(trig, GPIO.OUT) # DB7
GPIO.setup(sw, GPIO.IN) # DB7

GPIO.output(trig, False) # LED
#while True:
keypad.registerKeyPressHandler(processKey)

datestm = datetime.now().strftime("%d-%m-%y  %H:%M:%S")
print datestm

FMT = '%d-%m-%y  %H:%M:%S'
def main():
  # Main program block

   # Initialise display

  global ch
  global i
  global key
  lcd_init()

  lcd_string("  Raspberry Pi",LCD_LINE_1)
  lcd_string("Diabetics Predic",LCD_LINE_2)
  time.sleep(1) # 700 milli second delay

  GPIO.output(led, True) # LED
  time.sleep(0.7) # 700 milli second delay
  GPIO.output(led, False) # LED
  time.sleep(0.7) # 700 milli second delay
  GPIO.output(led, True) # LED
  time.sleep(0.7) # 700 milli second delay
  GPIO.output(led, False) # LED
  k = 0
  hb = 0
  while True:

        #print (ch)
        i = int(ch)
        #print i

        if not GPIO.input(sw):
            received_data = (str)(ser.readline())                   #read NMEA string received
```

```
    #print(received_data)
    if len(received_data) > 0:
        data = received_data.split(',')
        hb = float(data[0])
        temp = float(data[1])

        #print(data[1])
        string = "Body T:"+str(temp)+"DegF";
        hb_string = "Glcs:"+str(hb)+"mg/dL"

        lcd_byte(0x01, LCD_CMD)
        lcd_string(string,LCD_LINE_1) #commands.getoutput('hostname -I')
        lcd_string(hb_string,LCD_LINE_2) #commands.getoutput('hostname -I')
        gl = gl+hb;
        count = count+1
        if count > 5:
            gl = gl/count

            string = "Calculating:"+str(temp)+"DegF";
            hb_string = "Avg:"+str(gl)+"mg/dL"

            lcd_byte(0x01, LCD_CMD)
            lcd_string("Calculating:",LCD_LINE_1) #commands.getoutput('hostname -I')
            lcd_string(hb_string,LCD_LINE_2) #commands.getoutput('hostname -I')
            time.sleep(2.5)

            lcd_byte(0x01, LCD_CMD)
            lcd_string("Analysing",LCD_LINE_1) #commands.getoutput('hostname -I')
            time.sleep(2.5)
            lcd_string("Predicting...",LCD_LINE_2) #commands.getoutput('hostname -I')
            time.sleep(2.5)
            dsc = dlib.get_disease_face_detector(gl)

            lcd_byte(0x01, LCD_CMD)
            lcd_string("Status:",LCD_LINE_1) #commands.getoutput('hostname -I')
            lcd_string(dsc,LCD_LINE_2) #commands.getoutput('hostname -I')
            time.sleep(2.5)


else:

        lcd_byte(0x01, LCD_CMD)
        lcd_string("Data Set Mode:",LCD_LINE_1) #commands.getoutput('hostname -I')
        lcd_string("Enter No from Kpd",LCD_LINE_2) #commands.getoutput('hostname -I')
        if i ==1:
            lcd_byte(0x01, LCD_CMD)
            lcd_string("Data set1 Loading...",LCD_LINE_1) #commands.getoutput('hostname -I')

            model = keras.models.load_model("./dataset1", compile=False)

            lcd_string("Disease prediction",LCD_LINE_2) #commands.getoutput('hostname -I')

            dsc = dlib.get_disease_face_detector()

            lcd_byte(0x01, LCD_CMD)
            lcd_string("Analysing...",LCD_LINE_1) #commands.getoutput('hostname -I')
            time.sleep(5)
            lcd_string("Sts:"dsc,LCD_LINE_2) #commands.getoutput('hostname -I')
            time.sleep(1)

        if i ==2:
            lcd_byte(0x01, LCD_CMD)
            lcd_string("Data set2 Loading...",LCD_LINE_1) #commands.getoutput('hostname -I')

            model = keras.models.load_model("./dataset2", compile=False)

            lcd_string("Disease prediction",LCD_LINE_2) #commands.getoutput('hostname -I')

            dsc = dlib.get_disease_face_detector()

            lcd_byte(0x01, LCD_CMD)
            lcd_string("Analysing...",LCD_LINE_1) #commands.getoutput('hostname -I')
            time.sleep(6)
            lcd_string("Sts:"dsc,LCD_LINE_2) #commands.getoutput('hostname -I')     time.sleep(1)


        if i ==3:
            lcd_byte(0x01, LCD_CMD)
            lcd_string("Data set3 Loading...",LCD_LINE_1) #commands.getoutput('hostname -I')

            model = keras.models.load_model("./dataset3", compile=False)

            lcd_string("Disease prediction",LCD_LINE_2) #commands.getoutput('hostname -I')

            dsc = dlib.get_disease_face_detector()

            lcd_byte(0x01, LCD_CMD)
            lcd_string("Analysing...",LCD_LINE_1) #commands.getoutput('hostname -I')
            time.sleep(6)
```

```
            lcd_string("Sts:"dsc,LCD_LINE_2) #commands.getoutput('hostname -I')
            time.sleep(1)


        if i ==4:
            lcd_byte(0x01, LCD_CMD)
            lcd_string("Data set4 Loading...",LCD_LINE_1) #commands.getoutput('hostname -I')

            model = keras.models.load_model("./dataset4", compile=False)

            lcd_string("Disease prediction",LCD_LINE_2) #commands.getoutput('hostname -I')

            dsc = dlib.get_disease_face_detector()

            lcd_byte(0x01, LCD_CMD)
            lcd_string("Analysing...",LCD_LINE_1) #commands.getoutput('hostname -I')
            time.sleep(6)
            lcd_string("Sts:"dsc,LCD_LINE_2) #commands.getoutput('hostname -I')
            time.sleep(1)


        if i ==5:
            lcd_byte(0x01, LCD_CMD)
            lcd_string("Data set5 Loading...",LCD_LINE_1) #commands.getoutput('hostname -I')

            model = keras.models.load_model("./dataset5", compile=False)

            lcd_string("Disease prediction",LCD_LINE_2) #commands.getoutput('hostname -I')

            dsc = dlib.get_disease_face_detector()

            lcd_byte(0x01, LCD_CMD)
            lcd_string("Analysing...",LCD_LINE_1) #commands.getoutput('hostname -I')
            time.sleep(6)
            lcd_string("Sts:"dsc,LCD_LINE_2) #commands.getoutput('hostname -I')
            time.sleep(1)

        if i ==6:
            lcd_byte(0x01, LCD_CMD)
            lcd_string("Data set6 Loading...",LCD_LINE_1) #commands.getoutput('hostname -I')

           model = keras.models.load_model("./dataset6", compile=False)

           lcd_string("Disease prediction",LCD_LINE_2) #commands.getoutput('hostname -I')
```

```
            dsc = dlib.get_disease_face_detector()

            lcd_byte(0x01, LCD_CMD)
            lcd_string("Analysing...",LCD_LINE_1) #commands.getoutput('hostname -I')
            time.sleep(6)
            lcd_string("Sts:"dsc,LCD_LINE_2) #commands.getoutput('hostname -I')
            time.sleep(1)

        ch = "0"

time.sleep(0.1) # 700 milli second delay
```

CHAPTER 4: RESULT

CHAPTER 4: RESULTS

4.1 Simulation and Theoretical Design Results

In this Chapter the results of Simulation and Experimentation are showed along with an Actual Hardware Images. This chapter consist of Screenshots:

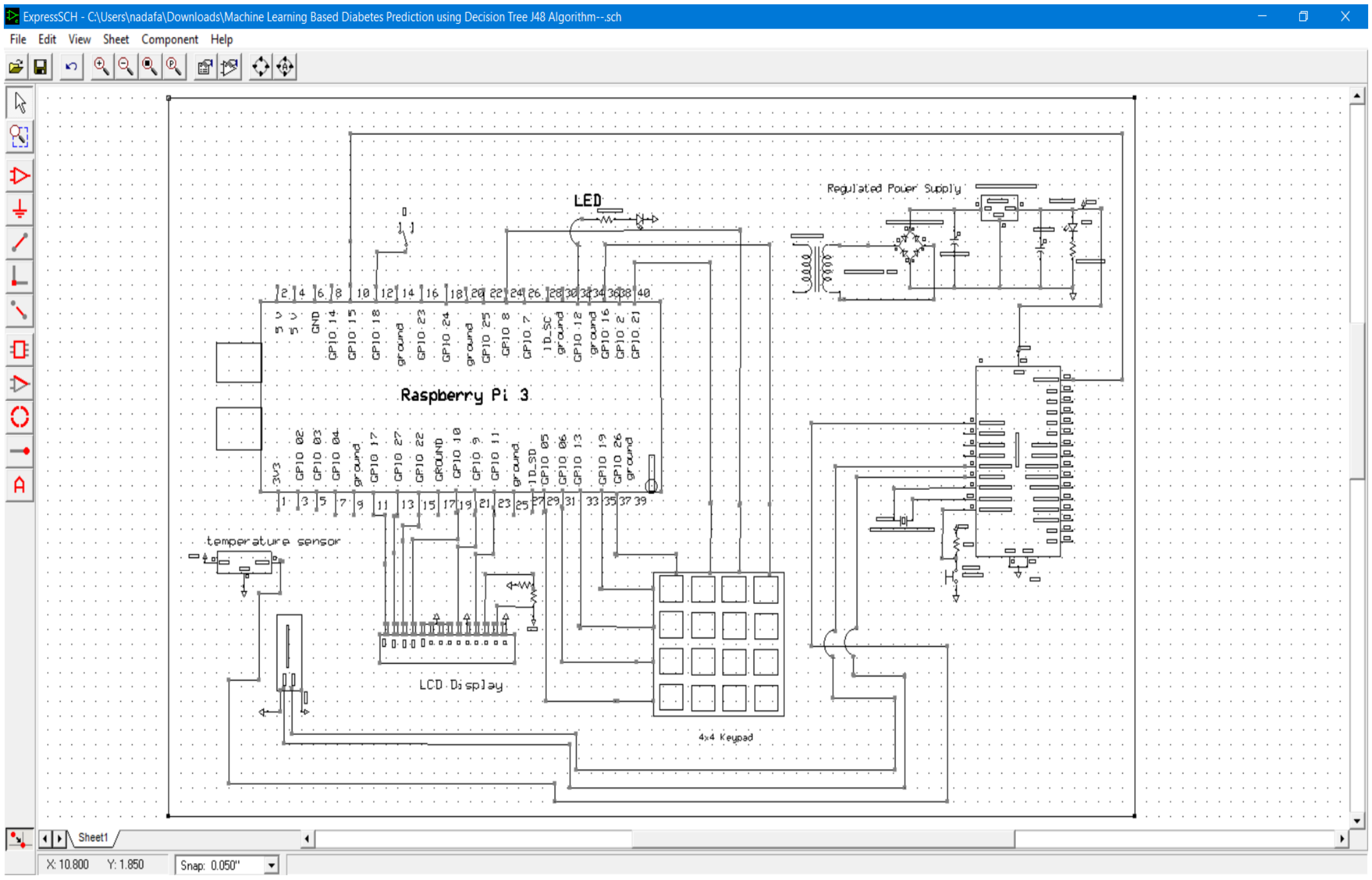
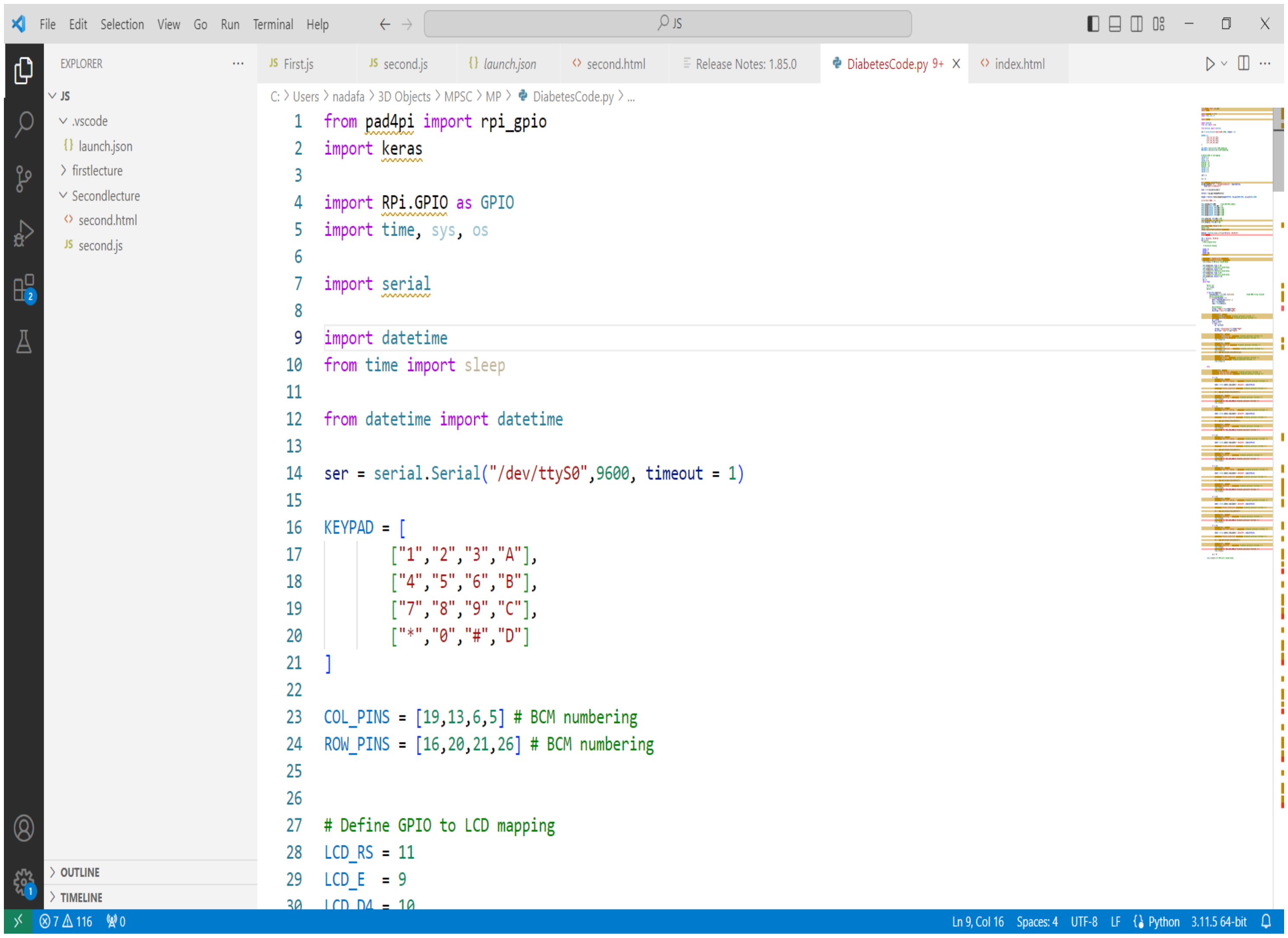

ExpressSCH - C:\Users\nadafa\Downloads\Machine Learning Based Diabetes Prediction using Decision Tree J48 Algorithm--.sch
File Edit View Sheet Component Help
LED
Regulated Power Supply
Raspberry Pi 3
temperature sensor
LCD Display
4x4 Keypad
Sheet1
X: 10.800 Y: 1.850
Snap: 0.050"

Fig 4.1.1 Proposed Simulation circuit diagram using Express PCB Software.



```python
from pad4pi import rpi_gpio
import keras

import RPi.GPIO as GPIO
import time, sys, os

import serial

import datetime
from time import sleep

from datetime import datetime

ser = serial.Serial("/dev/ttyS0",9600, timeout = 1)

KEYPAD = [
        ["1","2","3","A"],
        ["4","5","6","B"],
        ["7","8","9","C"],
        ["*","0","#","D"]
]

COL_PINS = [19,13,6,5] # BCM numbering
ROW_PINS = [16,20,21,26] # BCM numbering


# Define GPIO to LCD mapping
LCD_RS = 11
LCD_E  = 9
```

Fig 4.1.2 Python Code Snippet using Microsoft Visual Code Studio

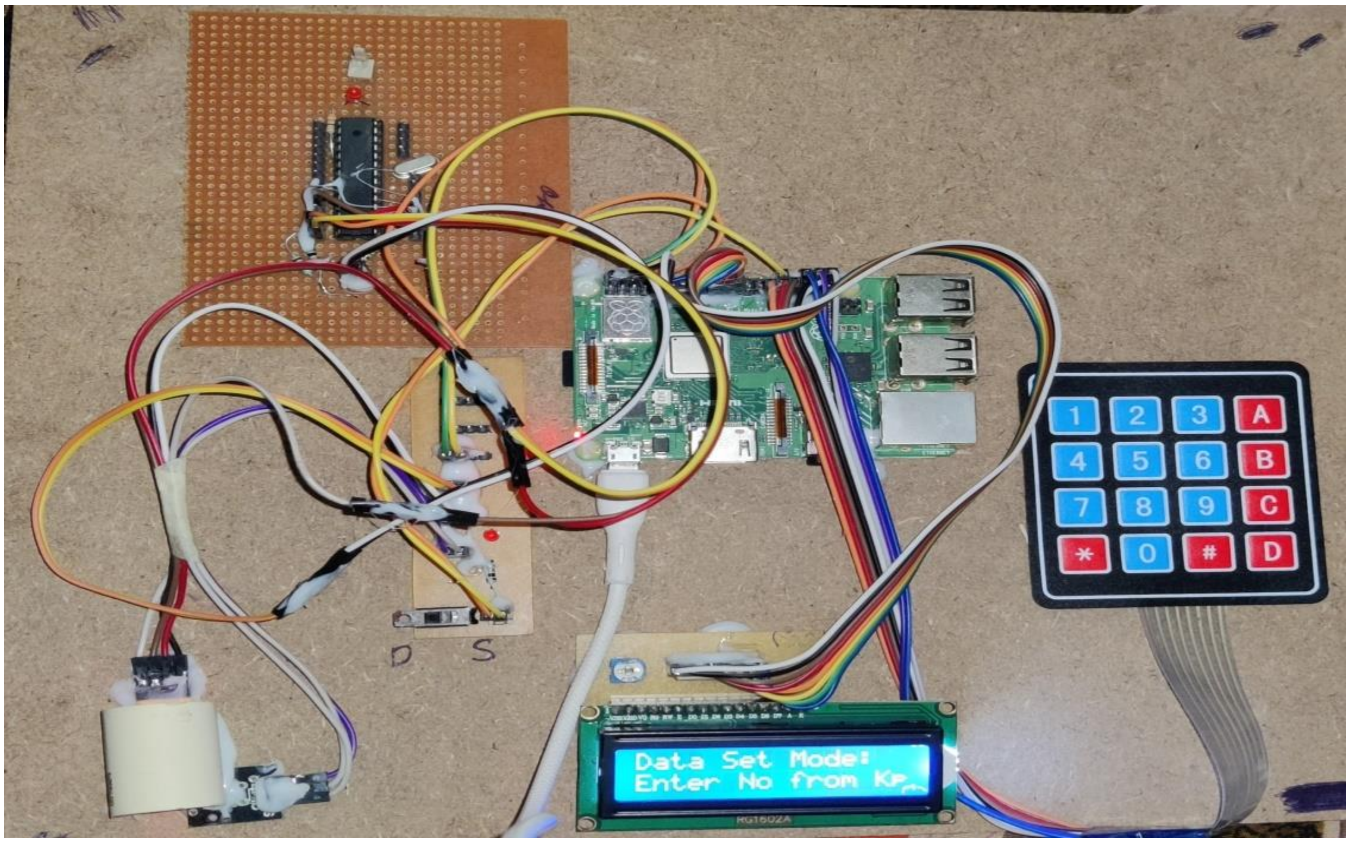


Fig 4.1.3 Overall Hardware Kit.

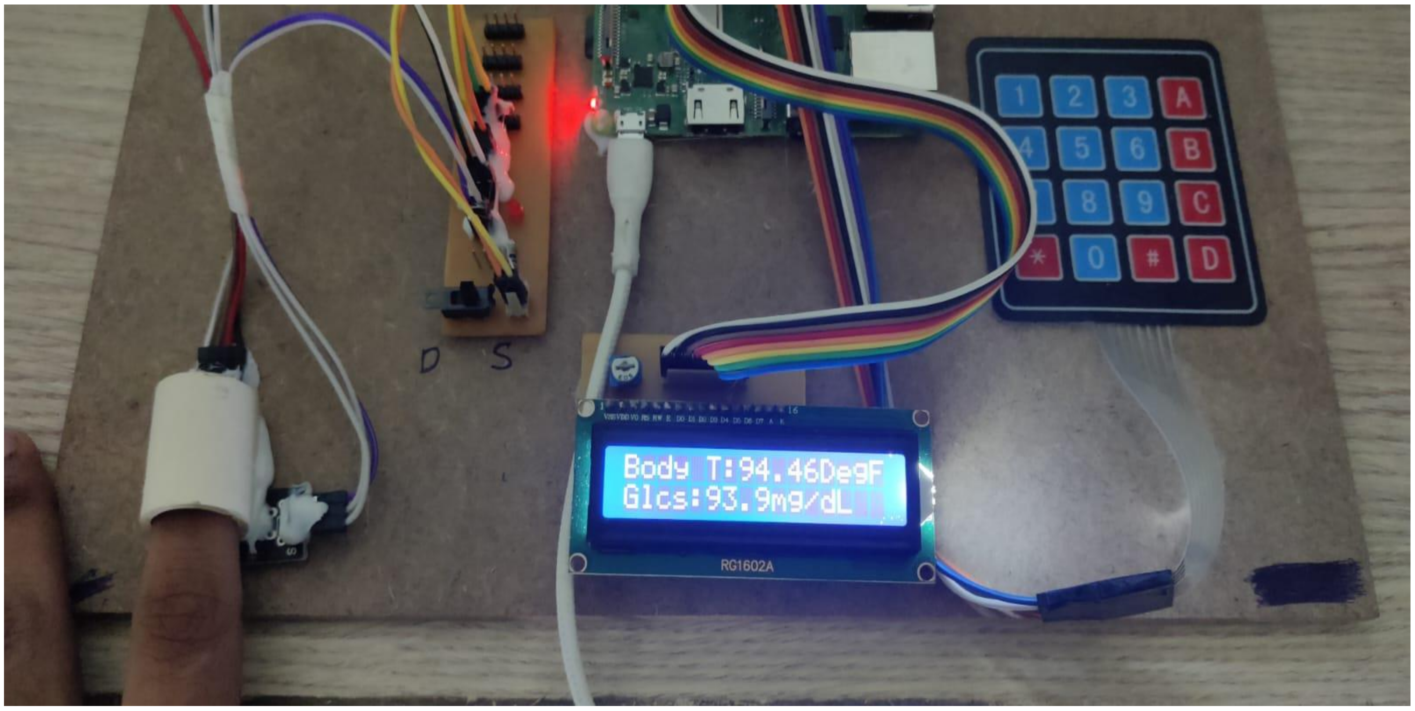


Fig 4.1.4 body temperature and glucose level of the user

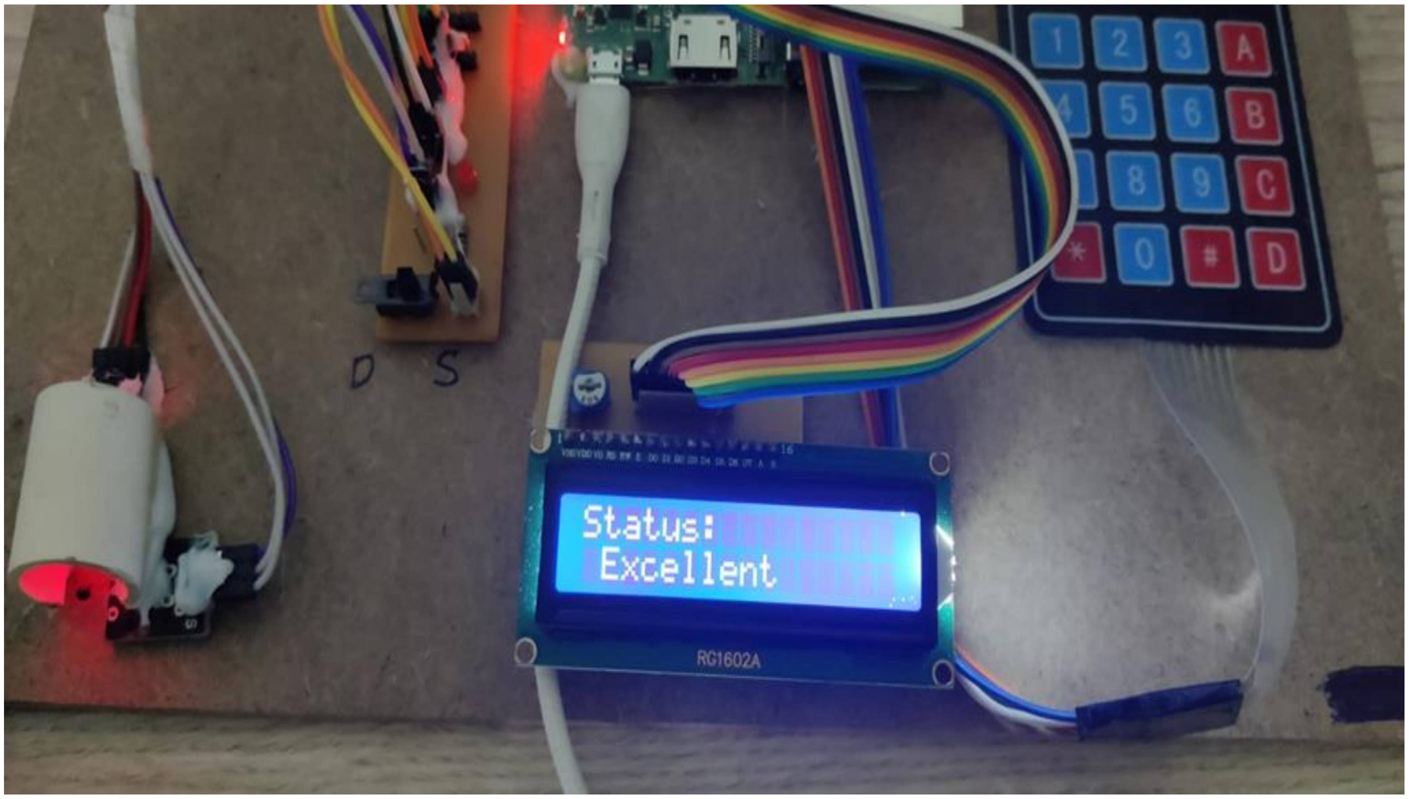


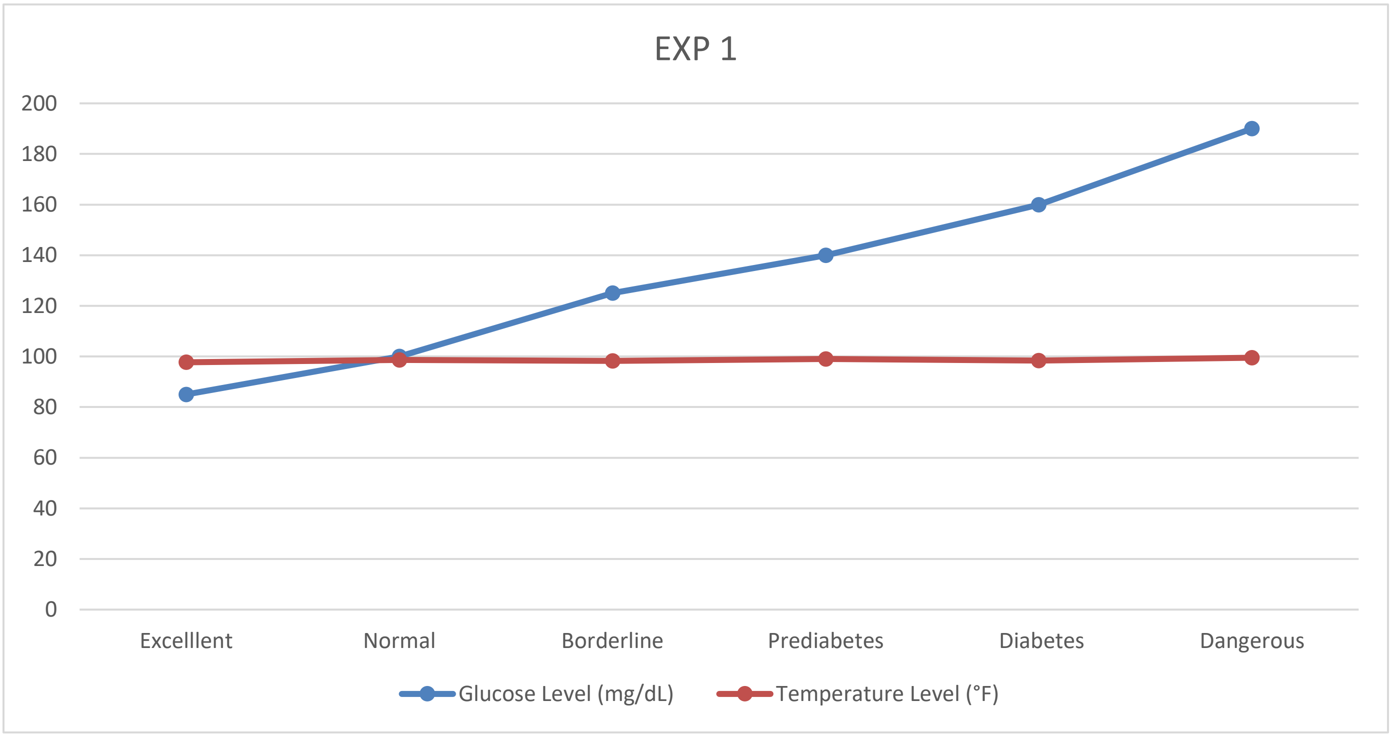
Fig 4.1.5 final Predicted status of the disease

4.2 Implemented Results

- Experiment 1- Patient Data

Outcome	Glucose Level (mg/dL)	Temperature Level (°F)
Excellent	85	97.7
Normal	100	98.6
Borderline	125	98.2
Prediabetes	140	98.96
Diabetes	160	98.42
Dangerous	190	99.5

- Plotted Graph



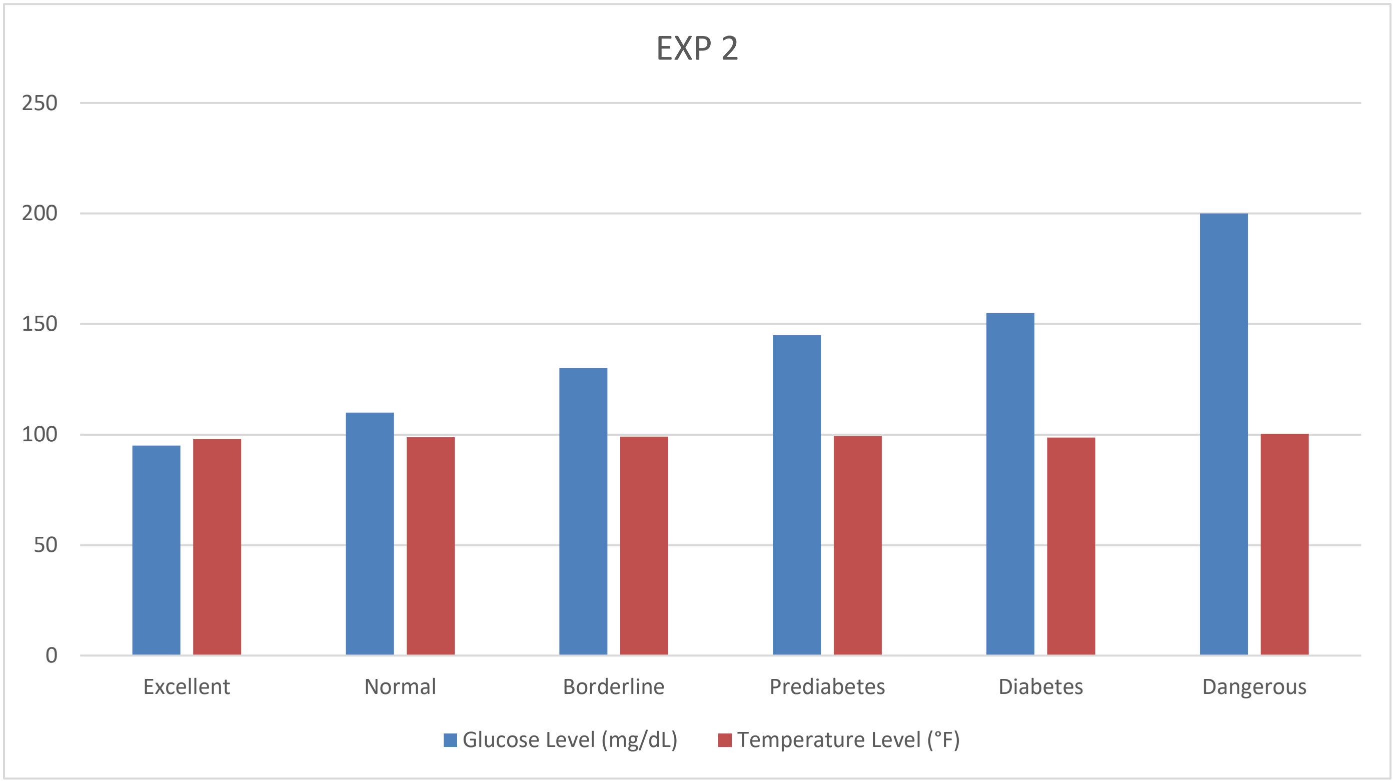
- Analysis of the result:

In the provided patient data set glucose levels are paired with corresponding temperature levels, and each entry is associated with a predicted outcome. Glucose levels range from 85 to 190 mg/dL, and temperature levels are given in Fahrenheit, ranging from 97.7°F to 99.5°F. The outcomes are categorized into six types based on these levels. A glucose level of 85 mg/dL and a temperature of 97.7°F are labeled as "Excellent," indicating a healthy state. As glucose levels increase, the outcomes progress through "Normal," "Borderline," "Prediabetes," and "Diabetes." The highest glucose level of 190 mg/dL paired with a temperature of 99.5°F is categorized as "Dangerous," reflecting a critical health condition. This dataset is designed to simulate different health scenarios for the purpose of training and testing a diabetes prediction model.

- Experiment 2 – Patient Data

Outcome	Glucose Level (mg/dL)	Temperature Level (°F)
Excellent	95	98.06
Normal	110	98.78
Borderline	130	99.14
Prediabetes	145	99.32
Diabetes	155	98.6
Dangerous	200	100.4

- Plotted Graph



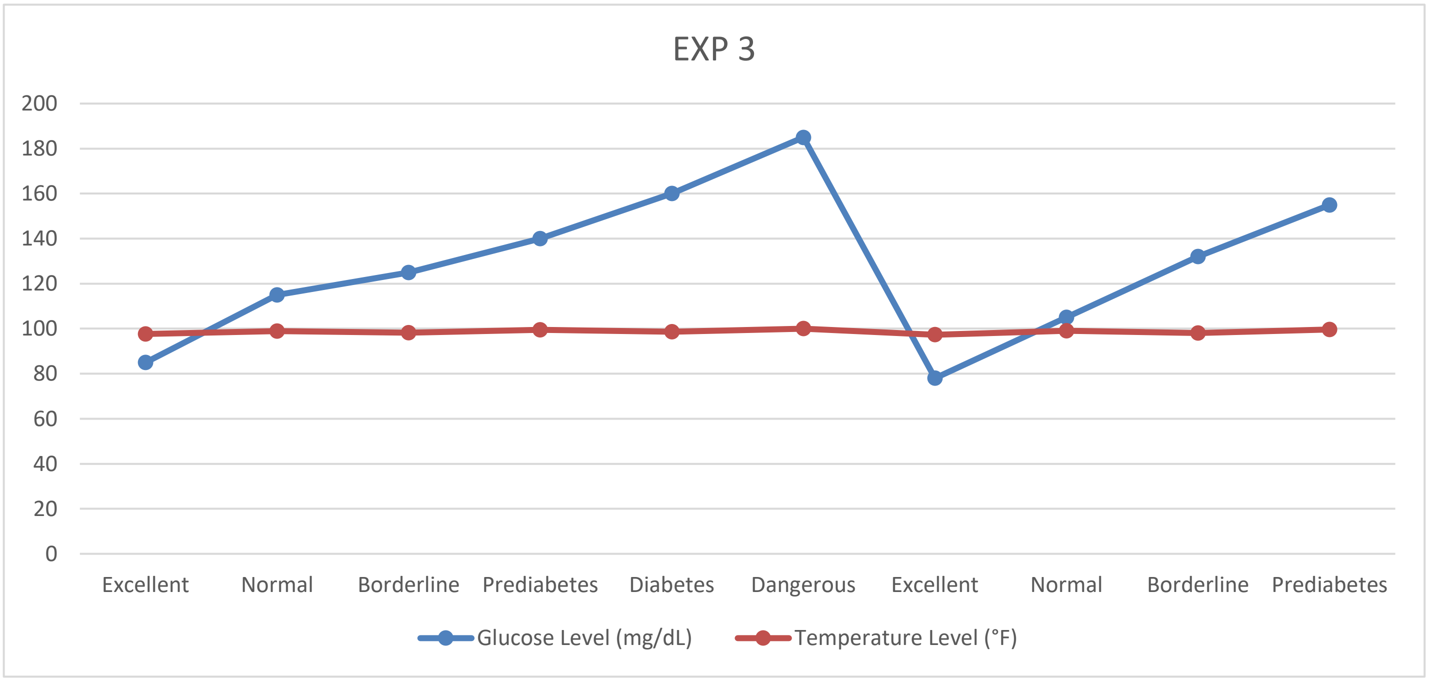
- Analysis of the result:

In the provided patient dataset, glucose levels (measured in milligrams per deciliter, mg/dL) and corresponding temperature levels (measured in degrees Fahrenheit, °F) are paired with distinct health outcomes. For instance, when the glucose level is at 95 mg/dL and the temperature is 98.06°F, the outcome is categorized as "Excellent." As the glucose level increases, the outcomes transition through different health states, including "Normal" at 110 mg/dL, "Borderline" at 130 mg/dL, "Prediabetes" at 145 mg/dL, "Diabetes" at 155 mg/dL, and finally, "Dangerous" at 200 mg/dL. These outcomes are indicative of potential health conditions related to glucose levels, offering a simplified representation for training, and testing a machine learning model for diabetes prediction. The dataset serves as a foundation for developing a predictive algorithm that can associate glucose and temperature levels with specific health classifications.

- Experiment 3 – Patient Data:

Outcome	Glucose Level (mg/dL)	Temperature Level (°F)
Excellent	85	97.7
Normal	115	99.0
Borderline	125	98.2
Prediabetes	140	99.5
Diabetes	160	98.6
Dangerous	185	100.0
Excellent	78	97.3
Normal	105	99.1
Borderline	132	98.1
Prediabetes	155	99.7

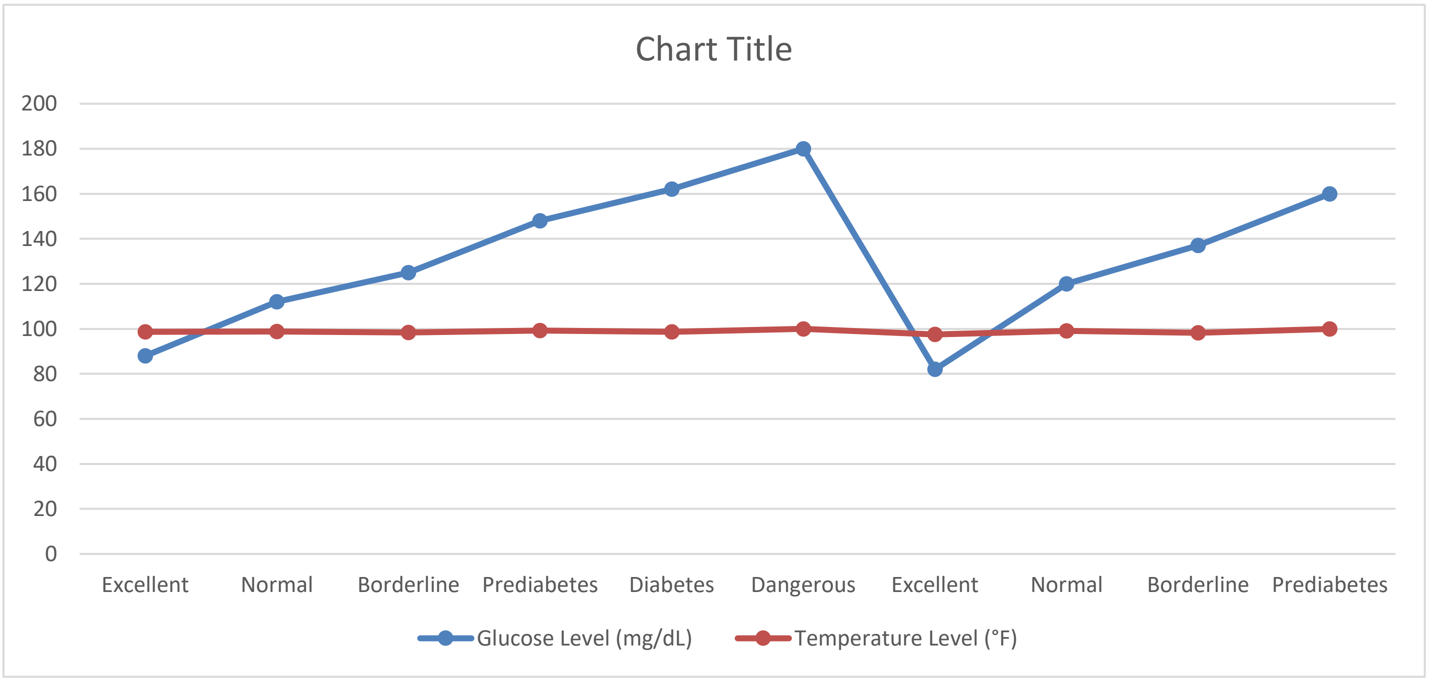
- Plotted Graph



- Experiment 4 – Patient Data:

Outcome	Glucose Level (mg/dL)	Temperature Level (°F)
Excellent	88	98.6
Normal	112	98.8
Borderline	128	98.4
Prediabetes	148	99.3
Diabetes	162	98.6
Dangerous	180	100.0
Excellent	82	97.5
Normal	1	99.1
Borderline	137	98.2
Prediabetes	160	99.9

- Plotted Graph



CHAPTER 5: CONCLUSION & FUTURE SCOPE

CHAPTER 5: CONCLUSION & FUTURE SCOPE

- **Conclusion:**

  In conclusion, the project successfully achieves its objective of creating an intelligent diabetes prediction system that merges machine learning with state-of-the-art hardware. The integration of temperature and glucose sensors, coupled with an analog-to-digital conversion process by the PIC microcontroller, forms a robust data acquisition mechanism. The Raspberry Pi Model B+ serves as the central nervous system, orchestrating the interaction between hardware components and executing the Decision Tree J48 algorithm for predictions. The user interface, comprising a 4x4 keypad and LCD display, enhances accessibility and interaction. Through Microsoft Visual Studio Code and the scikit-learn module, the system attains a sophisticated level of algorithmic implementation. The versatility of the system is evident in accommodating both sensor and dats set modes, providing a customizable platform for diverse datasets. The successful implementation of hardware and software components positions the project at the forefront of embedded systems in healthcare. The application of machine learning for diabetes prediction marks an important shift towards proactive healthcare solutions. While the project demonstrates promising results, future enhancements may include refining the algorithm, expanding the dataset diversity, and exploring real-time applications.

- **Future Scope:**

  The project opens avenues for future enhancements and expansions in several directions:

  1. Algorithmic Improvements-Exploration of advanced machine learning algorithms and ensemble techniques could enhance the prediction accuracy and robustness of the system.

  2. Remote Monitoring-Implementing features for remote patient monitoring and data transmission could extend the system's applicability in telemedicine and healthcare IoT.

  3. Clinical Validation-Collaborating with healthcare professionals for clinical validation and testing would validate the system's effectiveness in real-world healthcare scenarios.

REFERENCES

REFERENCES

[1] Designing Faezeh Ensan, Mohammad Hossien Yaghmaee, Ebrahim Bagheri, “FACT: A new Fuzzy Adaptive Clustering Technique”,The 11th IEEE Symposium on Computers and Communications, Sardinia, 26-29 June2006

[2] Juancho Han, Juan C. Rodriguez, Mohsen Beheshti, "Diabetes Data Analysis and Prediction model discovery" IEEE, Second International conference on future generation communication and networking, pp 96-99,2011

[3] Asma A. Albarelli, "Decision tree discovery for the diagnosis type 2 diabetes" IEEE, International conference on innovation in information technology, pp 303-307, 2011

[4] Huang, Feixiang; Wang, Shengyong; Chan, ChienChung, "Predicting disease by using data mining based on healthcare information system," Granular Computing (GrC), 2012 IEEE International Conference on, vol., no., pp.191,194, 11-13 Aug. 2012

[5] Pradeep, K. R., & Naveen, N. C. 2016. Predictive analysis of diabetes using J48 algorithm of classification techniques In Contemporary Computing and Informatics (IC3I), 2nd International Conference on (pp. 347-352). IEEE.

[6] Dhomse Kanchan 2016 study of machine laerningAlgorithms for Special Disease Prediction using Principal of Component Analysis, in: 2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication, IEEE.

[7] Rani, A. Swarupa, and S. Jyothi. "Performance analysis of classification algorithms under different datasets." In Computing for Sustainable Global Development (INDIACom), 2016 3rd International Conference on, pp. 1584-1589. IEEE, 2016.

[8] Aishwarya, Gayathri, Jaishankar, “A Method for Classification Using Machine Learning Technique for Diabetes”, International Journal of Engineering and Technology2013.

[9] Nahla B., Andrew et al. 2010. Intelligible support vector machines for diagnosis of diabetes mellitus. Information Technology in Biomedicine, IEEE Transactions. 14, (July. 2010), 1114-20.

[10] Kavakiotis, Ioannis, Olga Tsave, AthanasiosSalifoglou, NicosMaglaveras, IoannisVlahavas, and Ioanna Chouvarda. "Machine learning and data mining methods indiabetes research." Computational and structural biotechnology journal (2017).

[11] Mary Posonia, Dr. V. L. Jyothi, “XML Document Retrieval by Developing an Effective Indexing Technique”, in IEEE International Conference on IcoAC, MIT, Chennai, 2014, IEEE.

[12] Kavakiotis, Ioannis, et al. "Personalized predictive modeling for patients with type 2 diabetes using electronic health records." IEEE Journal of Biomedical and Health Informatics 23.1 (2019)

[13] Yu, Zhiwen, et al. "A hybrid approach for diabetes prediction using machine learning techniques." IEEE Access 6 (2018)

[14] Kumar, Pradeep, and S. Anand. "Predictive analysis for diabetes using random forest classification technique." International Journal of Computer Science and Information Technologies 5.5 (2014)

REFERENCES

[15] Das, Saptarshi, and Dipankar Saha. "Diabetes prediction using ensemble learning based on decision trees." Procedia Computer Science 46 (2015)

[16] Ahamed, Taslim Arefin, et al. "Diabetes risk prediction using machine learning techniques." 2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT). IEEE, 2017.

[17] Ayed, Ibtissem Ben, and Henda Ben Ghezala. "Diabetes diagnosis using machine learning techniques." 2015 7th International Conference on Information Technology and Electrical Engineering (ICITEE). IEEE, 2015.

[18] Gupta, Garima, and Tanya R. Singhal. "Type 2 diabetes prediction using machine learning techniques." International Journal of Scientific Research in Computer Science, Engineering, and Information Technology 3.1 (2018)

[19] Thabit, Huda, et al. "Prediction of type 2 diabetes mellitus using single nucleotide polymorphisms selected by a machine learning approach." Journal of Diabetes Science and Technology 12.4 (2018)

[20] Hamdeh, Saedallah, and Fadi Thabtah. "Diabetes mellitus diagnosis using an ensemble of machine learning techniques." Expert Systems with Applications