

# **MULTIPLE DISEASE PREDICTION SYSTEM USING MACHINE LEARNING TECHNIQUES**

**18CSP107L  
A PROJECT REPORT**

*Submitted by*  
**SRITEJ SHINE (RA2011026010212)  
KOLIPAKULA DWARAKADEESH (RA2011026010229)**

*Under the Guidance of*  
**Dr. P. KANAGARAJU**  
(Assistant Professor, Department of Computational Intelligence)

*In partial fulfillment of the Requirements for the Degree  
of*

**BACHELOR OF TECHNOLOGY  
in  
COMPUTER SCIENCE AND ENGINEERING WITH SPECIALIZATION  
IN ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING  
of  
FACULTY OF ENGINEERING AND TECHNOLOGY**



**DEPARTMENT OF COMPUTATIONAL INTELLIGENCE  
FACULTY OF ENGINEERING AND TECHNOLOGY  
SRM INSTITUTE OF SCIENCE AND TECHNOLOGY  
KATTANKULATHUR-603203**

**NOVEMBER 2023**

# **SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

(Under Section 3 of UGC Act, 1956)

## **BONAFIDE CERTIFICATE**

Certified that 18CSP107L Minor project report titled “**Multiple Diseases Prediction System using Machine Learning Techniques**” is the bonafide work “**SRITEJ SHINDE(RA2011026010212), KOLIPAKULA DWARAKADEESH(RA2011026010229)**” of who carried out the project work under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion for this or any other candidate.

SIGNATURE

**Dr. P. Kanagaraju**

Guide

Assistant Professor

Department of Computational Intelligence

SIGNATURE

**Dr. R. Annie Uthra**

Professor and Head

Department of Computational Intelligence

SIGNATURE

**Dr. D. Anitha**

Panel Head

Assistant Professor

Department of Computational Intelligence

**Department of Computational Intelligence**  
**SRM Institute of Science & Technology**

**Own Work Declaration Form**

**Degree/ Course : Bachelor of Technology, Computer Science Engineering**

**Student Name : SRITEJ SHINDE, KOLIPAKULA DWARAKADEESH**

**Registration Number : RA2011026010212, RA2011026010229**

**Title of Work : Multiple diseases prediction using Machine Learning Techniques**

We hereby certify that this assessment compiles with the University's Rules and Regulations relating to Academic misconduct and plagiarism, as listed in the University Website, Regulations, and the Education Committee guidelines. We confirm that all the work contained in this assessment is my / our own except where indicated, and that we have met the following conditions:

- Clearly references / lists all sources as appropriate
- Referenced and put in inverted commas all quoted text (from books, web, etc)
- Given the sources of all pictures, data etc. that are not my own
- Not made any use of the report(s) or essay(s) of any other student(s) either past or present
- Acknowledged in appropriate places any help that I have received from others (e.g. fellow students, technicians, statisticians, external sources)
- Compiled with any other plagiarism criteria specified in the Course handbook / University website

I understand that any false claim for this work will be penalized in accordance with the University policies and regulations

**DECLARATION:**

I am aware of and understand the University's policy on Academic misconduct and plagiarism and I certify that this assessment is my / our own work, except where indicated by referring, and that I have followed the good academic practices noted above.

**STUDENT 1 SIGNATURE:**

**STUDENT 2 SIGNATURE:**

**DATE:**

## ACKNOWLEDGEMENT

We express our humble gratitude to **Dr. C. Muthamizhchelvan**, Vice-Chancellor, SRM Institute of Science and Technology, for the facilities extended for the project work and his continued support.

We extend our sincere thanks to Dean-CET, SRM Institute of Science and Technology, **Dr. T.V. Gopal**, for his invaluable support.

We wish to thank **Dr. Revathi Venkataraman, Professor & Chairperson**, School of Computing, SRM Institute of Science and Technology, for her support throughout the project work.

We are incredibly grateful to our Head of the Department, **Dr. R. Annie Uthra, Professor**, Department of Computational Intelligence, SRM Institute of Science and Technology, for her suggestions and encouragement at all the stages of the project work.

We want to convey our thanks to our Panel head, **Dr. D. Anitha, Assistant Professor**, Department of Computational Intelligence, SRM Institute of Science and Technology, for her input during the project reviews and support.

We register our immeasurable thanks to our Faculty Advisor, **Dr. S. Velliangiri, Assistant Professor**, Department of Computational Intelligence, SRM Institute of Science and Technology, for leading and helping us to complete our course.

Our inexpressible respect and thanks to my guide, **Dr. P. Kanagaraju, Assistant Professor**, Department of Computational Intelligence, SRM Institute of Science and Technology, for providing me with an opportunity to pursue my project under his mentor-ship. He provided me with the freedom and support to explore the research topics of my interest. His passion for solving problems and making a difference in the world has always been inspiring.

We sincerely thank the Computational Intelligence staff and students, SRM Institute of Science and Technology, for their help during our project. Finally, we would like to thank parents, family members, and friends for their unconditional love, constant support, and encouragement.

**SRITEJ SHINDE(RA2011026010212)**

**KOLIPAKULADWARAKADEESH(RA2011026010229)**

## ABSTRACT

The "Multiple Disease Prediction System" is a web application designed to predict the likelihood of two common health conditions, namely diabetes and heart disease, using machine learning models. This system leverages the power of Streamlit, a user-friendly Python library for creating interactive web applications, to provide an intuitive and accessible interface for users to input their health-related data and obtain predictions regarding these diseases. The application begins by importing essential libraries, including `pickle` for loading pre-trained machine learning models and `Streamlit` for web app development. Two machine learning models are loaded into memory, one for diabetes prediction and another for heart disease prediction. The user interface is thoughtfully designed with a sidebar that allows users to choose between "Diabetes Prediction" and "Heart Disease Prediction." This approach enables users to access the specific prediction they require, making the application versatile and user-centric.

For the "Diabetes Prediction" feature, the user is presented with input fields to specify various health parameters, such as the number of pregnancies, glucose levels, blood pressure, skin thickness, insulin levels, BMI, diabetes pedigree function, and age. Once these details are entered, the user can trigger the prediction process by clicking the "Diabetes Test Result" button. The machine learning model processes the input data and generates a prediction, subsequently informing the user whether they are at risk of diabetes. Similarly, for "Heart Disease Prediction," the user is prompted to input health-related information such as age, sex, chest pain types, resting blood pressure, cholesterol levels, fasting blood sugar levels, and other critical variables. Upon clicking the "Heart Disease Test Result" button, the system utilizes the pre-trained heart disease prediction model to assess the user's risk of heart disease. The results are then presented to the user in a clear and comprehensible manner. In both scenarios, the application provides instant feedback to the user, indicating whether they are at risk of the respective diseases or not. This empowers users to make informed decisions about their health and seek professional medical advice if necessary.

# TABLE OF CONTENTS

<b>ABSTRACT</b>	<b>v</b>
<b>LIST OF FIGURES</b>	<b>viii</b>
<b>ABBREVIATIONS</b>	<b>ix</b>
<b>1 INTRODUCTION</b>	<b>1</b>
1.1 Disease Detection System	3
1.2 Data Collection and Management	3
1.3 Feature Engineering	3
1.4 Machine Learning Models	3
1.5 Model Validation and Evaluation	4
1.6 Real-Time Data Integration	4
1.7 User-Friendly Interface	4
1.8 Integration with Healthcare Ecosystem	4
1.9 Security and Privacy	5
1.10 Continuous Improvement and Maintenance	5
<b>2 LITERATURE SURVEY</b>	<b>6</b>
<b>3 PROBLEM STATEMENT AND PROPOSED SOLUTION</b>	<b>10</b>
3.1 Problem Statement	11
3.2 Problem Module	13
<b>4 METHODOLOGY</b>	<b>16</b>
4.1 Data Collection	16
4.2 Data Preprocessing	17
4.3 Model Loading	17
4.4 User Interface Creation using Streamlit	17
4.5 Making Disease Prediction	17
4.6 Modeling	18
4.7 Cross-Validation	18

4.8	Model Generation	20
4.9	Hyper-parameter Tuning	23
4.10	Training Steps of Support Vector Machine Model	25
4.11	Output Prediction	27
<b>5</b>	<b>TECHNICAL REQUIREMENTS</b>	<b>29</b>
<b>6</b>	<b>SYSTEM DESIGNS</b>	<b>30</b>
6.1	System Components	30
6.2	System Flow	31
6.3	System Design Considerations	32
<b>7</b>	<b>CODING AND TESTING</b>	<b>34</b>
<b>8</b>	<b>RESULTS AND ANALYSIS</b>	<b>44</b>
8.1	Result	44
8.2	Analysis	47
<b>9</b>	<b>CONCLUSION AND FUTURE DEVELOPMENTS</b>	<b>48</b>
	<b>REFERENCES</b>	<b>51</b>
	<b>APPENDIX</b>	
<b>A</b>	<b>CONFERENCE PUBLICATION</b>	<b>54</b>
<b>B</b>	<b>PLAGIARISM REPORT</b>	<b>55</b>

## **LIST OF FIGURES**

<b>Figure No.</b>	<b>Figure Name</b>	<b>Page No.</b>
1.1	System Architecture	2
3.1	Design of Prediction system	15
7.1	Shows the python libraries	34
7.2	Shows the head values of the data	34
7.3	Shows the X values of the data	35
7.4	Shows the Y values of the data	36
8.1	Shows the diabetes prediction Interface	45
8.2	Shows the heart disease prediction Interface	46



## **LIST OF ABBREVIATIONS**

<b>AI</b>	Artificial Intelligence
<b>BMI</b>	Body Metabolism Index
<b>SVM</b>	Support Vector Machine
<b>FED</b>	Feature Engineering and Designing
<b>ML</b>	Machine Learning
<b>R&amp;D</b>	Research and Development
<b>UX</b>	User Experience
<b>GUI</b>	Graphical User Interface
<b>API</b>	Application Programming Interface

# CHAPTER 1

## INTRODUCTION

In an era where data-driven decision-making is paramount, the integration of advanced technologies, particularly machine learning, is transforming the landscape of healthcare and diagnostics. Recognizing the significance of this transformation, the “Multiple Disease Prediction System” emerges as a groundbreaking web application designed to empower individuals with the ability to predict two common but potentially life-altering health conditions: diabetes and heart disease.

Chronic illnesses, such as diabetes and heart disease, have reached epidemic proportions globally, significantly impacting the quality of life and longevity of individuals. Timely diagnosis and risk assessment play a pivotal role in the prevention and management of these conditions. This is where the “Multiple Disease Prediction System” steps in, bridging the gap between cutting-edge technology and accessible healthcare information.

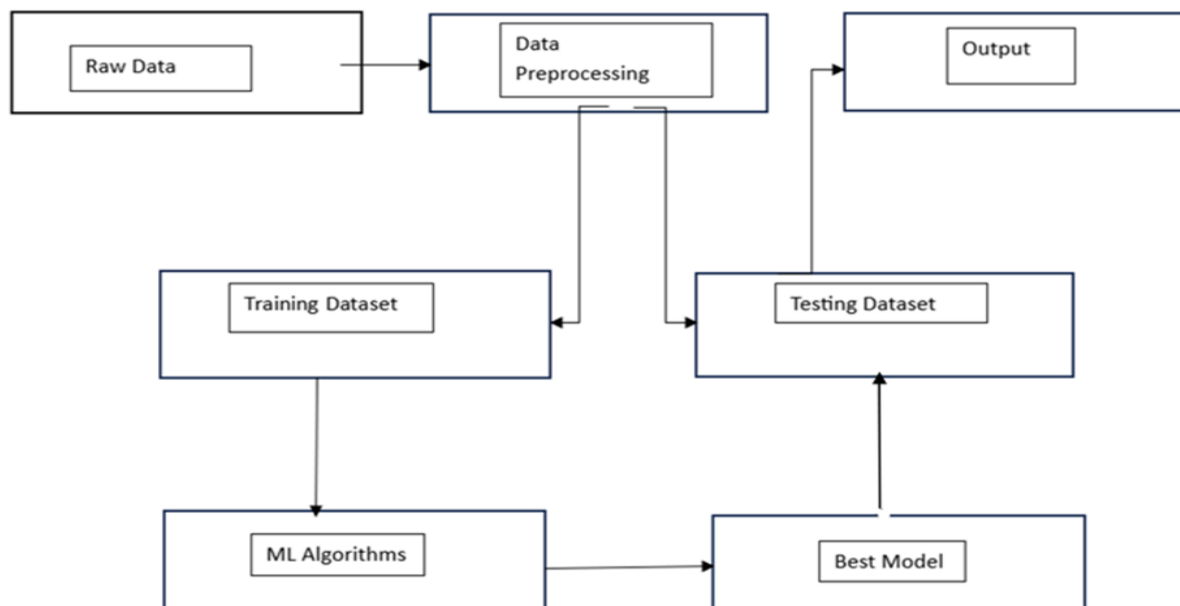
The heart of this innovative system lies in its Integration of machine learning models, enabling accurate predictions based on a user’s health-related data. With a user-friendly interface powered by Streamlit, a Python library tailored for creating interactive web applications, this platform provides individuals with a simple yet robust tool for disease risk assessment.

This application is a testament to the fusion of healthcare expertise and technological prowess. It encapsulates the knowledge of medical professionals, encoded into machine learning algorithms, and puts it into the hands of individuals, enabling them to take proactive steps toward healthier living. Offering instant predictions, it enhances personal health awareness and encourages early intervention, potentially saving lives

In the subsequent sections, searching into the workings and functionalities of the “Multiple Disease Prediction System.” Here explore its intuitive design, seamless navigation, and precise prediction capabilities that make it a valuable asset for individuals seeking to understand and mitigate their

risk of diabetes and heart disease. With the power of data-driven insights at their fingertips, users can make informed decisions and embark on a journey toward better health and well-being.

The "Multiple Disease Prediction System" is not just a web application; it represents a fusion of technology and healthcare, a gateway to preventive healthcare, and a tool for personalized well-being. Here searching into its inner workings, and discover a powerful instrument that leverages technology for the betterment of personal health, revolutionizing of the approach and address health concerns.



**Fig. 1.1 Overview of Multiple disease Prediction**

The figure 1.1 shows the system architecture for the multiple disease prediction system typically involves several components of data working together to provide the information for multiple disease prediction. Using Machine Learning Algorithms below provides the explanation of the components of the architecture diagram.

## **1.1 Disease Detection System**

Creating a robust and elaborate disease detection system requires careful planning, the integration of cutting-edge technology, and a thorough understanding of the healthcare domain. Such a system can have a profound impact on healthcare by aiding in early disease detection, providing timely interventions, and ultimately improving patient outcomes. In this comprehensive guide, let's outline the key components and considerations involved in building an advanced disease detection system.

## **1.2 Data Collection and Management**

The foundation of any disease detection system is data. Comprehensive and diverse datasets containing health-related information are essential for training and validating machine learning models. Data can be collected from various sources, including electronic health records, wearable devices, medical imaging, and patient-reported data. Ensuring the quality, privacy, and security of these datasets is of utmost importance.

## **1.3 Feature Engineering**

Feature engineering involves selecting relevant variables or attributes from the collected data. This step is crucial for building effective disease prediction models. Domain expertise is essential here, as healthcare data can be highly nuanced. Feature engineering also includes data preprocessing, such as handling missing values and normalizing data.

## **1.4 Machine Learning Models**

The heart of the disease detection system lies in the selection and development of machine learning models. Depending on the specific disease, various algorithms such as logistic regression, decision trees, random forests, support vector machines, or deep learning techniques can be employed. These models are trained on historical data to learn patterns and make predictions about the presence or risk of disease.

### **1.5 Model Validation and Evaluation**

It is vital to validate the performance of disease detection models. This involves splitting the dataset into training and testing sets, employing cross-validation techniques, and assessing metrics like accuracy, precision, recall, F1-score, and ROC curves. Model performance should be thoroughly evaluated to ensure its reliability and generalization.

### **1.6 Real-time Data Integration**

For practical applications, real-time data integration is crucial. Disease detection systems may need to ingest data continuously, enabling immediate risk assessment and decision-making. This requires seamless integration with data sources, data processing pipelines, and real-time model deployment.

### **1.7 User-Friendly Interface**

A user-friendly interface is essential for the adoption of the disease detection system. Web applications, mobile apps, or dashboard interfaces can be developed to present the results to users in an understandable and accessible manner. Design considerations should prioritize simplicity and clarity.

### **1.8 Integration with Healthcare Ecosystem**

For healthcare professionals, seamless integration with existing healthcare systems and electronic health records (EHRs) is important. This allows for streamlined access to patient data and ensures that predictions are available to medical staff during clinical decision-making.

## **1.9 Security and Privacy**

Given the sensitivity of healthcare data, robust security and privacy measures must be in place. Compliance with data protection regulations like HIPAA (Health Insurance Portability and Accountability Act) is crucial.

## **1.10 Continuous Improvement and Maintenance**

Healthcare is an evolving field, and disease detection models should be regularly updated and improved. Continuous monitoring of model performance, data quality, and emerging medical knowledge is essential. The disease detection system represents a dynamic fusion of data, technology, healthcare expertise, and a commitment to improving healthcare outcomes. When designed and implemented with care, such systems have the potential to revolutionize disease prevention, early intervention, and patient care. The journey to building such a system is a multidisciplinary effort that can significantly impact the future of healthcare and the well-being of individuals.

## CHAPTER 2

### LITERATURE SURVEY

Cellular network providers are becoming increasingly competitive in the telecommunications business, and churn control has emerged as a critical issue in the telecommunications industry. Sharma (2023) used a neural network to forecast customer turnover in cellular network service. The dataset comprises 20 variables with data of about 2,427 customers gathered from the University of California, Irvine's UCI Machine Learning Database. Clementine data mining software program from SPSS Inc. was used to create the neural network. Clementine provides two distinct types of supervised neural networks, namely Multilayer Perceptron (MLP) and Radial Basis Function Network (RBFN). The algorithm forecasts customer attrition with an accuracy rate of more than 92%, according to the results.

Taiwo and Adeyemo (2022) employed observational and predictive data mining approaches to analyze neural feature behavior and identify potential diseases with a high likelihood of churn in a medical imaging system. During the descriptive stage, patients were grouped based on their usage behavioral characteristics, and the clustering techniques employed were K-Means and Expected Maximization (EM). Weka was used to implement the Decision Stump, M5P, and Rep Tree classifier algorithms during the prediction stage. The results reveal that EM outperforms K-mean in the descriptive stage while M5P outperforms both Decision Stump and Rep Tree in the predictive stage.

He *et al.* (2022) conducted a study on Multiple Diseases affected patients via Support Vector Machine. The dataset comprises 50000 patient records that have been retrieved from a database system of a Chinese medical dataset. 46,406 records were used to model after missing values and anomalies were eliminated. Even though the SVM technique was used, due to the unbalanced qualities of the dataset, the Random sampling approach was used to improve SVM since it has a greater degree of recognition. The findings showed that incorporating random sampling and the support vector machine method boosted predictive power and properly forecasted churning rate.

Adeyemo and Oyeniya (2022) Multiple Diseases region specification has become a key issue in a disease-focused healthcare business, and banks have attempted to measure patient interaction in order to discover early warning indications in patient behavior, such as a decrease in the rate of affected identity and response activity. They also worked on disease research in the banking industry, developing a model that employed K-Means and Repeated Incremental Pruning to Produce Error Reduction (JRip algorithm) and was deployed on CNN. The dataset was collected from a large image based medical ideation database and image warehouse. The results show patterns in consumer behavior and assist banks in identifying clients who are likely to have classified into Glioma, Meningioma and Pituitary Diseases in the various regions of the .

Wang *et al.* (2021) developed a huge ensemble model for predicting client attrition in search ads. A prominent theme in Multiple Diseases detection research involves the utilization of advanced imaging modalities. Magnetic Resonance Imaging (MRI) remains a cornerstone, offering high-resolution anatomical images. However, studies such as this have explored the integration of functional MRI (fMRI) and diffusion tensor imaging (DTI) to not only locate diseases but also assess their impact on surrounding tissue, providing valuable insights for surgical planning. characteristics are complimentary, with an AUC (area under the curve of ROC) value of 0.8410.

Rosa (2021) Machine learning and AI have emerged as pivotal tools in automating the detection and classification of Multiple Disease. Research highlights the effectiveness of convolutional neural networks (CNNs) in identifying disease regions within MRI scans, achieving high accuracy rates. Similarly, demonstrates the potential of deep learning models in distinguishing between benign and malignant diseases, offering rapid and precise diagnoses. Feature extraction techniques have gained attention for their role in improving disease detection accuracy. Research papers such as emphasize the importance of extracting texture and shape features from MRI images, showing how these features can aid in distinguishing between disease types and predicting patient outcomes. Multi-Modal Approaches: Several studies have embraced multi-modal approaches that combine different imaging techniques to enhance diagnostic accuracy.



R.komar, *et al.* (2021) - "Scalable and accurate deep learning with electronic health records": This study demonstrated the potential of deep learning models in utilizing electronic health records for precise diagnosis and treatment recommendations, even though it was not exclusively dermatology-focused.

Aditi Gavhane (2020), Gouthami Kokkula, Isha Panday, Prof. Kailash Devadkar, "Prediction of Heart Disease using Machine Learning" Gavhane et al. Have worked on the multi-layer perceptron model for the prediction of heart diseases in human being and the accuracy of the algorithm using CAD technology. If the number of person using the prediction system for their diseases prediction, then the awareness about the diseases is also going to increases and it make reduction in the death rate of heart patient.

Pahulpreet Singh Kohli and Shriya Arora (2021), "Application of Machine Learning in Diseases Prediction Machine learning algorithms are used for various type of diseases predication and many of the researchers have work on this like Kohali et al. work on heart diseases prediction using logistic regression, diabetes prediction using support vector machine, breast cancer prediction using Adaboost classifier and concluded that the logistic regression give the accuracy of 87.1%, support vector machine give the accuracy of 85.71%, Adaboost classifier give the accuracy up to 98.57% which good for predication point of view

Sarwar, *et al.* (2021), discuss predictive analytics in healthcare, a number of machine learning algorithms are used in this study. For experiment purposes, a dataset of patient's medical is obtained. The performance and accuracy of the applied algorithms are discussed and compared.

Afzal Hussain Shahid and Maheshwari Prasad Singh (2020), proposed the paper titled "A deep learning approach for prediction of Parkinson's disease progression". This paper proposed a deep neural network (DNN) model using the reduced input feature space of Parkinson's telemonitoring dataset to predict Parkinson's disease (PD) progression and also proposed a PCA based DNN model for the prediction of Motor-UPDRS and Total-UPDRS in Parkinson's Disease progression. The DNN model was evaluated on a real-world PD dataset taken from UCI.

Mohan *et al.* (2021), have introduced a novel technique that has aimed at discovering important features by utilizing machine-learning model for ensuring in enhancing the accuracy in cardiovascular disease prediction. The prediction approach was initiated by diverse integrations of features and classification methods. The experimental analysis has revealed that the deep learning approaches have achieved maximum accuracy while evaluating with other existing methods

Adhi *et al.* (2020), have implemented a Coronary Heart Disease (CHD) using deep structured learning models. The base deep structured learning models of another ensemble were utilized for generated a two-tier ensemble. The forecasting model has estimated on standardized heart disease datasets to generalize the features of the suggested model. However, the optimal features were obtained using particle swarm optimization-based feature selection. Finally, the suggested method was evaluated using the twofold and tenfold statistical test, and it was revealed that the recommended method has provided enriched performance than the other optimization algorithms.

Fitriyani *et al.* (2020), have estimated a successful method to predict heart disease that has consisted of clustering-based application for detecting and eliminating the outliers, over-sampling approach for corresponding the distribution of learning data and XGBoost for predicting the heart disease. Two datasets were utilized for constructing the model and the experimental outcomes were evaluated with existing models. The outcomes have demonstrated that the suggested model has performed well, while evaluating with diferent traditional approaches. Moreover, the proposed model has designed the prototype to assist doctors detect the patient's heart disease level in terms of their present situation.

Men *et al.* (2021), have proposed a novel model for performing multi-disease prediction. The suggested approach has used a deep learning approach and extended it with two procedures. The time-aware procedure was utilized for handling the sequential indiscretion across clinical visits. For prediction task, the attention-based procedure has helped in resolving the signficance of every visit. Thus, the suggested model has obtained good performance while predicting future disease diagnoses.

## **CHAPTER 3**

### **PROBLEM STATEMENT AND PROPOSED SOLUTION**

The use of Multiple disease prediction systems has increased over the past decade as the Medical science industry moves towards digitization and automation of services. The development of the technical system has been driven by the need to improve doctor care, reduce Medical errors, and increase efficiency in Medical science delivery. IMS has also been designed to address the challenges of managing large volumes of doctor data and records, which can be time-consuming and costly when done manually. This paper provides an overview of Medical science application management systems, their features, and the benefits they offer to Medical science organizations. The paper discusses the various types of IMS, the key features of IMS, and their impact on Medical science delivery.

Multiple Diseases detection is a critical area of medical diagnosis, where early and accurate identification plays a pivotal role in improving patient outcomes. However, the existing methods often face challenges in terms of detection accuracy, efficiency, and patient-centered care. Additionally, the psychological and emotional well-being of patients undergoing Multiple disease diagnoses and treatment is often overlooked. To address these challenges comprehensively, proposing a Multiple Diseases detection system with Machine Learning techniques.

The proposed solution aims to provide a holistic approach to Multiple Diseases detection and patient care, leveraging the power of machine learning algorithms while incorporating psychological well-being considerations. The solution can be divided into different components: Multiple Disease detection, Front-End User Interface, Machine Learning Models, Navigation Sidebar, Result Display.

### **3.1 Problem Statement**

#### **1 Health Condition Prediction:**

The primary goal is to predict the likelihood of two major health conditions: diabetes and heart disease. These conditions have a significant impact on public health, and early detection and intervention can be vital for affected individuals.

#### **2 Accessibility:**

The problem statement recognizes the need for accessibility to health risk prediction. Users should be able to access this service conveniently from the comfort of their homes or any location with an internet connection.

#### **3 User-Friendly Interface:**

The problem involves creating a user-friendly and intuitive interface for users, allowing them to enter their health-related data with ease.

#### **4 Data-Driven Predictions:**

The problem statement focuses on leveraging historical health-related data to make informed predictions about whether an individual is likely to have diabetes or heart disease.

Machine learning models are used to process this data and provide predictions based on patterns learned from past cases.

#### **5 Customized Predictions:**

The application aims to provide personalized predictions by considering individual health-related features and information provided by the user.

## **6 Health Awareness:**

The problem addresses the need to raise health awareness by giving users insight into their risk of diabetes or heart disease, potentially motivating them to seek medical advice or make lifestyle changes.

## **7 Binary Classification:**

The problem involves binary classification tasks, where individuals are categorized into one of two classes: either at risk for the health condition or not at risk.

## **8 Model Deployment:**

The solution involves deploying pre-trained machine learning models to ensure that users can access the prediction service without requiring in-depth knowledge of machine learning or data science.

## **9 Data Privacy and Security:**

Ensuring data privacy and security is important. The problem statement should consider how user-provided health data is handled and stored to protect sensitive information.

## **10 Performance and Reliability:**

The system needs to be designed for performance and reliability to handle concurrent user requests and provide predictions in a timely and accurate manner.

## **11 Iterative Improvement:**

Continuous improvement is part of the problem statement, where models, features, and the user interface can be enhanced based on feedback and emerging data.

## **12 Documentation and Transparency:**

The problem statement may involve documenting how predictions are made, the machine learning models used, and providing transparency to users about the process. The problem statement

revolves around creating a web application that enables users to assess their risk of diabetes and heart disease through user-friendly, data-driven, and personalized predictions.

### **3.2 Proposed Module**

#### **1 Front-End User Interface Module:**

This module is responsible for creating the front-end interface of the web application using Streamlit. It provides the user with a visually appealing and user-friendly interface for entering health-related data and receiving prediction results. Components within this module include input fields for data entry, buttons for submitting data, and result display areas.

#### **2 Machine Learning Prediction Module:**

This module encompasses the machine learning components responsible for making predictions of diabetes and heart disease based on user-provided data. It involves loading pre-trained machine learning models (not visible in the code) from saved pickle files to perform the predictions. Specific functions are implemented to process user data, pass it to the appropriate model, and obtain the prediction results.

#### **3 Data Handling and Preprocessing Module:**

Data preprocessing is a critical step to ensure that the user-provided data is in the correct format for machine learning model input. The module handles data cleaning, missing value imputation, scaling, and any necessary transformations to make the data compatible with the models.

#### **4 Navigation and User Interaction Module:**

This module handles user interactions and navigation within the web application. It includes the navigation sidebar that allows users to select between "Diabetes Prediction" and "Heart Disease Prediction" and switch between the prediction tasks.

## **5 Result Display Module:**

After predictions are made by the machine learning models, this module is responsible for displaying the results to the user. It uses Streamlit's `st.success`` function to present messages to the user, indicating whether they are likely to have diabetes or heart disease based on the model's predictions.

## **6 Model Loading and Management Module:**

While not explicitly visible in the code, this module deals with loading the pre-trained machine learning models (SVM for heart disease and an unspecified model for diabetes) from saved pickle files. It may also involve managing the models, keeping them up to date, and ensuring they are available for prediction.

## **7 Security and Data Privacy Module:**

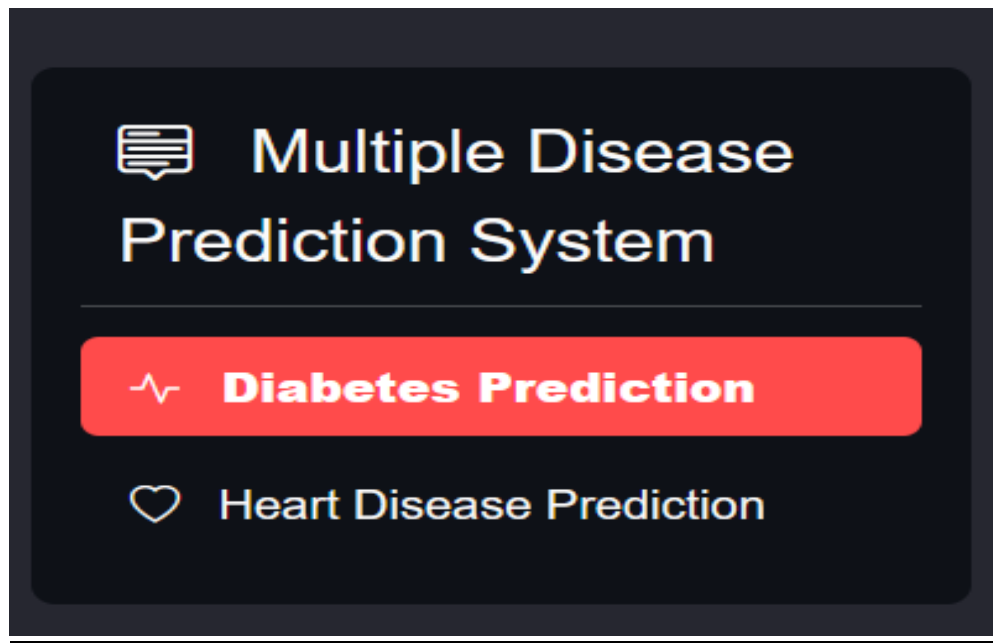
The module would typically address data privacy and security concerns, ensuring that user-provided health data is handled securely and protecting sensitive information.

## **8 Continuous Improvement and Feedback Module:**

The proposed module focuses on collecting user feedback and integrating iterative improvements into the application. It allows for enhancements in the form of feature updates, better model performance, and additional features based on user feedback and emerging data.

## **9 Documentation and Transparency Module:**

While not explicitly visible in the code, this module could involve creating user documentation and explanations about the prediction process, the machine learning models used, and the data sources to ensure transparency and user understanding. The proposed modules in the web application code are responsible for creating an accessible and user-friendly interface for predicting diabetes and heart disease, processing user data, making predictions, handling user interactions, and continuously improving the application.



**Fig. 3.1 Design of the Prediction System**

This figure 3.1 shows user interface the of the multiple disease prediction system which predicts diabetes prediction and heart disease prediction and the design of the interface



## **CHAPTER 4**

### **METHODOLOGY**

An integrated Medical science system is a complex and multifaceted endeavor that requires a comprehensive methodology for successful implementation. This project outlines a methodology for implementing an integrated Medical science system, covering key areas such as planning, stakeholder engagement, technology infrastructure, data management, and evaluation. The methodology for this application involves several key steps, including data collection, data preprocessing, model loading, user interface creation using Streamlit, and making disease predictions. Here's an elaboration of the methodology for the "Diabetes Prediction" and "Heart Disease Prediction" sections of the code

#### **4.1 Data Collection:**

The data collection process for both diabetes and heart disease prediction involves gathering specific health-related features that are known to be relevant for predicting these diseases. These features are entered by the user through a web interface.

For the "Diabetes Prediction" section, users provide information such as the number of pregnancies, glucose level, blood pressure, skin thickness, insulin level, BMI, diabetes pedigree function, and age. For the "Heart Disease Prediction" section, users input features including age, sex, chest pain type, resting blood pressure, serum cholesterol level, fasting blood sugar level, resting electrocardiographic results, maximum heart rate achieved, exercise-induced angina, ST depression induced by exercise, slope of the peak exercise ST segment, major vessels colored by fluoroscopy, and thalassemia status.

## **4.2 Data Preprocessing:**

The data entered by the user is typically in the form of text inputs, and it may need to be preprocessed to ensure that it matches the format expected by the machine learning models. This preprocessing step may involve data type conversion, handling missing values, and scaling/normalizing the input features.

## **4.3 Model Loading:**

The application loads pre-trained machine learning models using the `pickle` library. These models are already trained on historical data to make predictions.

## **4.4 User Interface Creation using Streamlit:**

The application uses Streamlit to create a user-friendly interface for users to interact with. It includes the following components:

A sidebar with navigation options allows users to choose between "Diabetes Prediction" and "Heart Disease Prediction." Input fields allow users to enter their health-related data for prediction. The input fields are organized in columns for easy data entry. Buttons labeled "Diabetes Test Result" and "Heart Disease Test Result" that users click to trigger the prediction process. Result display components (using `st.success`) for showing the prediction outcomes.

## **4.5 Making Disease Predictions:**

After the user enters their health-related data and clicks the respective prediction button, the application passes the user's input data to the loaded machine learning models (i.e., `diabetes\_model` for diabetes prediction and `heart\_model` for heart disease prediction). The machine learning models make predictions based on the user's input data. For diabetes, it predicts whether the person is diabetic or not, and for heart disease, it predicts whether the person has heart disease or not.

The prediction results are then displayed to the user as a message indicating the disease status, such as "The person is diabetic" or "The person is not diabetic" for diabetes prediction, and "The person has heart disease" or "The person does not have any heart disease" for heart disease prediction. The methodology involves user-provided data, model loading, a Streamlit-based user interface, and making predictions based on the input data using pre-trained machine learning models. The results are then presented to the user through the web application.

#### **4.6 Modeling:**

Modeling involves selecting an appropriate algorithm and training the model on the data. The model is evaluated using metrics such as accuracy, precision, and recall.

Overall, data analysis is a critical step in machine learning, as it involves preparing the data for modeling and gaining insights into the problem domain. By following these steps, let's ensure that the machine learning model is effective in solving the problem at hand.

#### **4.7 Cross Validation:**

Cross-validation is a critical step in evaluating the performance of machine learning models, ensuring that the model's performance is reliable and robust. In the context of the provided code for diabetes and heart disease prediction, cross-validation could be used to assess the models' performance. Here's an elaboration of how cross-validation could be applied to this scenario:

##### **1 Cross-Validation Overview:**

Cross-validation is a resampling procedure that involves splitting the available data into multiple subsets. It helps in estimating the model's performance on unseen data, detecting issues like overfitting or underfitting, and providing a more reliable assessment of model performance. Common techniques include k-fold cross-validation and stratified cross-validation.

## **2 Data Collection and Preparation:**

Before applying cross-validation, you would typically start with a dataset that includes historical health-related data and corresponding labels for diabetes and heart disease. This dataset is used to train and validate the machine learning models.

## **3 Model Training:**

The machine learning models for diabetes and heart disease prediction are trained using the available data. This is typically done with a training set, and the model learns to make predictions based on the input features.

## **4 Cross-Validation Implementation:**

K-Fold Cross-Validation:

Divide the dataset into 'k' roughly equal-sized folds or partitions (usually 5 or 10). Train and evaluate the model 'k' times, each time using a different fold as the validation set and the remaining folds as the training set. Calculate the performance metrics (e.g., accuracy, F1 score) for each fold. Compute the average and standard deviation of the performance metrics across the 'k' iterations.

Stratified Cross-Validation:

For imbalanced datasets (e.g., if there are significantly fewer positive cases for heart disease or diabetes), it's important to ensure that each fold has a representative distribution of classes. Stratified cross-validation maintains the class distribution in each fold.

## **5 Performance Metrics:**

Choose appropriate performance metrics to evaluate the models. Common metrics for binary classification tasks (like diabetes and heart disease prediction) include accuracy, precision, recall, F1-score, and the area under the receiver operating characteristic curve (AUC-ROC).

## **6 Model Selection and Hyperparameter Tuning:**

Cross-validation can help in comparing different models and selecting the best one. It can also be used for hyperparameter tuning by assessing model performance under various hyperparameter configurations.

## **7 Implementation in the Provided Code:**

To implement cross-validation in the provided code, you would need to make some modifications:

Load the dataset used for training and testing the models. Replace the direct use of ``diabetes_model`` and ``heart_model`` with a cross-validation loop. Split the data into training and testing sets in each fold. Train and evaluate the model in each fold, collecting performance metrics. Compute the average and standard deviation of the performance metrics across the folds. Display the cross-validated results, which provide a more reliable estimate of the models' performance.

By implementing cross-validation, you can obtain a better understanding of how well the machine learning models generalize to unseen data, and you can be more confident in the accuracy of the predictions made by the models.

## **4.8 Model Generation:**

In the provided code, the models for diabetes and heart disease prediction have been loaded from saved pickle files, which means they are pre-trained models. Model generation typically involves the following steps, but in this case, these steps would have been performed before saving the models to the pickle files. Here's an elaboration of how model generation would typically be done:

## **Model Generation for Diabetes and Heart Disease Prediction:**

### **1 Data Collection and Preparation:**

Collect a dataset that contains historical health-related data and labels for diabetes and heart disease. The dataset should be divided into features (input variables) and target labels (indicating whether the person has the disease or not).

### **2 Data Preprocessing:**

Preprocess the dataset by handling missing values, scaling or normalizing features, encoding categorical variables, and splitting the data into training and testing sets. Data preprocessing ensures that the input data is in a suitable format for machine learning.

### **3 Feature Selection or Engineering:**

Analyze the dataset and select relevant features that are likely to influence the prediction of diabetes or heart disease. Feature engineering may involve creating new features or transforming existing ones to improve model performance.

### **4 Model Selection:**

Choose the appropriate machine learning algorithms for each prediction task (diabetes and heart disease). Common algorithms for binary classification tasks include logistic regression, decision trees, random forests, support vector machines, and neural networks.

### **5 Model Training:**

Split the preprocessed data into a training set and a testing set. The training set is used to train the machine learning model, and the testing set is used to evaluate its performance.

Fit the selected machine learning model to the training data. During this process, the model learns the underlying patterns and relationships between the input features and the target labels.

## **6 Model Evaluation:**

After training the model, evaluate its performance on the testing set using appropriate evaluation metrics. Common metrics for binary classification include accuracy, precision, recall, F1-score, and the area under the receiver operating characteristic curve (AUC-ROC).

## **7 Hyperparameter Tuning:**

Fine-tune the model by adjusting hyperparameters to improve its performance. This can be done through techniques such as grid search or random search.

## **8 Model Validation:**

Perform cross-validation, as explained in a previous response, to ensure that the model's performance is consistent across different subsets of the data. This helps in detecting issues like overfitting and ensures robustness.

## **9 Model Saving:**

Once you are satisfied with the model's performance, save the trained model to a file using a serialization method such as `pickle` (as done in the provided code). This allows you to load and reuse the model without retraining it every time.

## **10 Model Deployment:**

In a real-world application, the trained model would be deployed so that users can make predictions. Deployment might involve creating a web service, API, or embedding the model into a web application, as seen in the code.

The "model generation" step is represented by loading the pre-trained models from saved pickle files. These pre-trained models have already gone through the steps mentioned above and are ready for making predictions. If you intend to retrain the models, you would need to follow the model generation steps described above and save the updated models for later use.

.

#### **4.9 Hyper-parameter Tuning:**

Hyperparameters are the settings or configurations of a machine-learning algorithm that cannot be learned from the data. They are set manually by the developer or researcher before training the model. Examples of hyper-parameters include learning rate, batch size, number of hidden layers, number of neurons in each layer, regularization strength, and activation function.

Hyperparameter tuning is the process of selecting the optimal values of these hyperparameters to improve the performance of the model on the test data. The need for hyper parameter tuning arises because different hyperparameter settings can significantly affect the performance of the model. If the hyper-parameters are not set correctly, the model may not perform well on the test data.

**1 Improving model accuracy:** The primary goal of hyper-parameter tuning is to improve the accuracy of the model on the test data. By selecting the optimal values of hyper-parameters, can be improved the model's ability to generalize to new data.

**2 Avoid overfitting:** Overfitting occurs when the model is too complex and fits the training data too well, resulting in poor performance on the test data. By tuning the hyper-parameters, controlling the complexity of the model, and avoiding over-fitting.

**3 Reducing training time** Hyper-parameter tuning can help to reduce the training time of the model by selecting the optimal hyper-parameters that lead to faster convergence.



**4 Improving interpretability** Some hyper-parameters affect the interpretability of the model. For example, the regularization parameter controls the complexity of the model, and increasing its value leads to a simpler model that is easier to interpret.

This is the way in which have tried to implement hyperparameter tuning in this project.

- i. Data preparation: The first step is to prepare the data by cleaning it, transforming it, and splitting it into training and testing sets.
- ii. Choosing the algorithm: There are various algorithms that can be used for disease analysis, such as logistic regression, decision trees, random forests, and neural networks. Each algorithm has its own set of hyper-parameters that can be tuned to improve its performance.
- iii. Selecting the hyper-parameters: The next step is to select the hyper-parameters that will be tuned. For example, in a neural network, the hyper-parameters might include the number of hidden layers, the number of neurons in each layer, the learning rate, and the regularization strength.
- iv. Defining the search space: The search space is the range of values that the hyper-parameters can take. For example, the learning rate might be searched in the range of 0.001 to 0.1, and the number of hidden layers might be searched in the range of 1 to 5.
- v. Tuning the hyper-parameters: There are various techniques that can be used to tune the hyper-parameters, such as grid search, random search, and Bayesian optimization. In grid search, all possible combinations of hyper-parameters are tried, and the best set is selected based on the performance on the validation set. In random search, random combinations of hyper-parameters are tried, and the best set is selected. In Bayesian optimization, a probabilistic model is used to predict the performance of different hyper-parameter settings, and the best set is selected based on this prediction.

- vi. Evaluating the performance: After hyper-parameter tuning, the performance of the model is evaluated on the test set. If the performance is not satisfactory, the hyper-parameters can be fine-tuned further.

In summary, hyper-parameter tuning is essential because it can significantly affect the performance of the model on the test data. By selecting the optimal values of hyper-parameters, improves the accuracy of the model, avoid over-fitting, reduce training time, and improve the interpretability of the model.

#### **4.10 Training Steps of Support Vector Machine Model:-**

Training a Support Vector Machine (SVM) involves several steps, and these steps apply to the SVM used for binary classification tasks such as diabetes or heart disease prediction. Below are the training steps for an SVM in this context:

##### **1 Data Collection and Preprocessing:**

Gather a dataset that includes historical health-related data and labels indicating whether an individual has diabetes or heart disease. Preprocess the data, which may include handling missing values, scaling or normalizing features, and encoding categorical variables.

##### **2 Data Splitting:**

Divide the preprocessed data into two subsets: a training set and a testing set. The training set is used to train the SVM, while the testing set is used to evaluate its performance.

##### **3 Feature Selection or Engineering:**

Analyze the dataset and select relevant features for prediction. Feature engineering may involve creating new features or transforming existing ones to improve model performance.

##### **4 Model Selection:**

Choose the SVM algorithm as the machine learning model for the binary classification task. The SVM aims to find a hyperplane that best separates the data into two classes.

## **5 Hyperparameter Selection and Tuning:**

Configure hyperparameters for the SVM. Important hyperparameters for an SVM include the choice of kernel (e.g., linear, polynomial, radial basis function), regularization parameter ( $C$ ), and the kernel-specific parameters (e.g., gamma for the RBF kernel).

Hyperparameter tuning can be performed using techniques such as grid search or random search to find the best combination of hyperparameters that optimizes model performance.

## **6 Model Training:**

Train the SVM on the training set. The SVM attempts to find the hyperplane that maximizes the margin between the two classes. In the case of soft-margin SVM, it allows for some misclassifications to improve generalization.

## **7 Model Evaluation:**

Assess the SVM's performance on the testing set using relevant evaluation metrics for binary classification tasks. Common metrics include accuracy, precision, recall, F1-score, and the area under the receiver operating characteristic curve (AUC-ROC).

## **8 Cross-Validation:**

Implement cross-validation to ensure the SVM's performance is consistent across different subsets of the data. Cross-validation helps detect issues like overfitting and provides a more robust assessment of performance.

## **9 Iterative Improvement:**

If the SVM's performance is not satisfactory, you may consider adjusting feature selection, feature engineering, or hyperparameters. Retrain the SVM with the updated data or hyperparameters.

## **10 Model Saving:**

Once satisfied with the SVM's performance, save the trained SVM model to a file using a serialization method like ``pickle`` or a library like scikit-learn's ``joblib``. This allows you to load and use the model in the future without retraining.

Training a Support Vector Machine for binary classification involves selecting the model, configuring its hyperparameters, training it on the training data, evaluating its performance, and optionally conducting feature selection, feature engineering, cross-validation, and hyperparameter tuning. The trained SVM model can then be saved and deployed for making predictions.

### **4.11 Output Prediction**

Output prediction in the provided code refers to the process of using trained machine learning models to predict whether an individual has diabetes or heart disease based on the input data provided by the user. Here's an elaboration of the output prediction process for both "Diabetes Prediction" and "Heart Disease Prediction" sections:

#### **1 Diabetes Prediction:**

After selecting "Diabetes Prediction" in the Streamlit interface, the user is presented with input fields to enter specific health-related information, such as the number of pregnancies, glucose level, blood pressure, and more. Once the user fills out these input fields, they click the "Diabetes Test Result" button. The entered input data is collected and passed to the pre-trained machine learning model, ``diabetes_model``, which has been loaded from a saved pickle file.

The model processes the input data and makes a prediction based on the learned patterns it has acquired during training. In the case of diabetes prediction, the model predicts whether the person has diabetes or not. It assigns a binary outcome: 1 for diabetic and 0 for not diabetic. The prediction result is then displayed to the user using Streamlit's ``st.success`` function. Depending on the model's prediction, one of the following messages is displayed:

"The person is diabetic" if the prediction is 1.

"The person is not diabetic" if the prediction is 0.

## 2 Heart Disease Prediction:

After selecting "Heart Disease Prediction" in the Streamlit interface, the user is presented with input fields to enter health-related information, including age, sex, chest pain type, and other relevant data. Once the user completes the input fields, they click the "Heart Disease Test Result" button. The entered input data is collected and passed to the pre-trained machine learning model, ``heart_model``, which has been loaded from a saved pickle file.

The model processes the input data and makes a prediction based on the patterns it has learned during training. In the case of heart disease prediction, the model predicts whether the person has heart disease or not. It assigns a binary outcome: 1 for having heart disease and 0 for not having heart disease.

The prediction result is displayed to the user using ``st.success``. Depending on the model's prediction, one of the following messages is displayed:

"The person is having heart disease" if the prediction is 1.

"The person does not have any heart disease" if the prediction is 0.

The output prediction process in this code involves the user entering health-related data, which is then used by pre-trained machine learning models to make predictions. The results are presented to the user through the Streamlit interface, indicating whether the person is likely to have diabetes or heart disease based on the input data and the model's learned patterns.

## **CHAPTER 5**

### **TECHNICAL REQUIREMENTS**

The disease analysis project is largely platform-independent and is useful for finding customer retention in the company. So, these are the basic system requirements to get the software working without any flaws.

#### **HARDWARE REQUIREMENTS:**

1. Laptop / PC with any OS (Windows 7 or later, Mac OS (any version), Linux (any version), or Mobile Device (Android or iOS).
2. Internet connection (12kbps is the minimum requirement).
3. Uninterrupted power supply.
4. Laptops or PCs with an i3 processor or better.
5. If using a mobile device, then any processor is good. (Recommended Snapdragon).
6. Virus-free environment.

#### **SOFTWARE REQUIREMENTS**

1. Any web browser (Chrome, Brave, Safari, Mozilla or Tor)
2. No third-party cookie blocking should be there.
3. Usage statistics should be disabled for better performance

With these minimum conditions satisfied the application can work blazing fast with minimum lags and can fetch good output for the user.

## **CHAPTER 6**

### **SYSTEM DESIGNS**

Designing a system for Multiple disease prediction, which is a web application for predicting diabetes and heart disease, involves structuring the components, data flow, and interactions to create a functioning and user-friendly application. Below is an elaboration of the system design for this application:

#### **6.1 System Components**

##### **1 Front-End Interface:**

The front end is built using Streamlit, a Python library for creating web applications. The user interacts with the application through the front-end interface, which provides input fields and buttons for data entry and prediction.

##### **2 Machine Learning Models:**

Pre-trained machine learning models are used for diabetes and heart disease prediction. These models have been trained on historical health-related data to make predictions based on user-provided input.

##### **3 Data:**

User-provided data, such as health-related features and information, is collected through the front-end interface. The input data is sent to the machine learning models for prediction.

##### **4 Prediction Engine:**

The prediction engine is responsible for processing the user's input data and making predictions using pre-trained machine learning models. It performs data preprocessing and sends the data to the appropriate model.

## **5 Result Display:**

The system displays prediction results to the user on the front-end interface. The results are presented as messages, indicating whether the person is likely to have diabetes or heart disease based on the input data and model predictions.

## **6 Navigation Sidebar:**

The sidebar provides navigation options for users to choose between "Diabetes Prediction" and "Heart Disease Prediction."

## **6.2 System Flow:**

### **1 User Interaction:**

Users select either "Diabetes Prediction" or "Heart Disease Prediction" from the navigation sidebar.

### **2 Data Entry:**

Users enter health-related data through input fields on the front-end interface. Data entered includes features such as age, sex, glucose level, blood pressure, and other relevant information.

### **3 Data Submission:**

Users click the "Diabetes Test Result" or "Heart Disease Test Result" button to submit the entered data.

### **4 Data Processing:**

The user-provided data is sent to the prediction engine, which preprocesses the data and ensures it is in the correct format for model input.



## **5 Prediction:**

The prediction engine uses the pre-trained machine learning models (diabetes\_model and heart\_model) to make predictions based on the input data. The models assign binary outcomes, such as 1 for having the disease and 0 for not having the disease.

## **6 Result Display:**

The prediction results are displayed to the user on the front-end interface using Streamlit's `st.success` function. The results inform the user whether they are likely to have diabetes or heart disease based on the model's prediction.

### **6.3 System Design Considerations:**

**1 Scalability:** The system can be scaled by deploying it on cloud servers to handle a large number of concurrent users.

**2 Data Security:** Ensure that user-entered data is handled securely, as it may contain sensitive health information.

**3 Model Maintenance:** Regularly update and retrain the machine learning models with fresh data to ensure the system's predictive accuracy.

**4 User Experience:** Focus on creating an intuitive and user-friendly front-end interface for easy data entry and result display.

**5 Performance:** Optimize the code and model loading to minimize response time and improve application performance.

**6 Model Explainability:** Consider providing explanations for model predictions to enhance user trust and understanding.

**7 Accessibility:** Ensure that the web application is accessible to users with disabilities by following web accessibility guidelines.

Overall, the system design focuses on delivering a reliable and user-friendly application for disease prediction while also considering aspects of data security, performance, and scalability.

## CHAPTER 7

### CODING AND TESTING

This Python 3 environment comes with many helpful analytics libraries installed. It is defined by the Kaggle/python Docker image: <https://github.com/kaggle/docker-python>

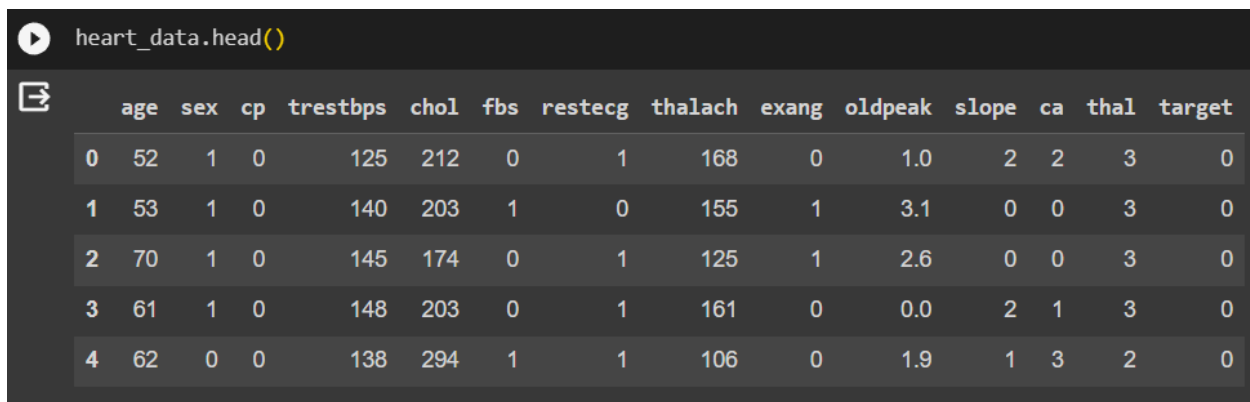
#### Importing Libraries:

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

**Fig. 7.1 Python Libraries**

The figure 7.1, shows the Python libraries used in the prediction of the code implementation

#### Training and Testing and Validation Table of the data:



	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0

**Fig. 7.2 Head Values of the Data**

The figure 7.2, shows the data used for the training and testing for the implementation of the code shows the value of the head of the data of the training and testing values

```
print(X)
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	\
0	52	1	0	125	212	0	1	168	0	1.0	
1	53	1	0	140	203	1	0	155	1	3.1	
2	70	1	0	145	174	0	1	125	1	2.6	
3	61	1	0	148	203	0	1	161	0	0.0	
4	62	0	0	138	294	1	1	106	0	1.9	
...	...	...	..	...	...	...	...	...	...	...	
1020	59	1	1	140	221	0	1	164	1	0.0	
1021	60	1	0	125	258	0	0	141	1	2.8	
1022	47	1	0	110	275	0	0	118	1	1.0	
1023	50	0	0	110	254	0	0	159	0	0.0	
1024	54	1	0	120	188	0	1	113	0	1.4	

	slope	ca	thal
0	2	2	3
1	0	0	3
2	0	0	3
3	2	1	3
4	1	3	2
...	...	..	...
1020	2	0	2
1021	1	1	3
1022	1	1	2
1023	2	0	2
1024	1	1	3

[1025 rows x 13 columns]

**Fig. 7.3 Values X of the Data to Train and Test**

In the figure 7.3, shows the values of X in the data of the training and testing of the dataset for the prediction

```
print(Y)
```

0	0
1	0
2	0
3	0
4	0
..	
1020	1
1021	0
1022	0
1023	1
1024	0

Name: target, Length: 1025, dtype: int64

**Fig. 7.4 Value of Y of the Data to Train and Test**

In the Figure 7.4, shows the value of the Y in the data of the training and testing of the dataset for the prediction

## Code:

### FOR THE HEART DISEASE PREDICTION

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
heart_data = pd.read_csv('/content/heart.csv')
heart_data.head()
heart_data.shape
heart_data.describe()
heart_data['target'].value_counts()
X = heart_data.drop(columns='target', axis=1)
Y = heart_data['target']
print(Y)
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, stratify=Y,
random_state=2)
print(X.shape, X_train.shape, X_test.shape)
model = LogisticRegression()
model.fit(X_train, Y_train)
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)
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 = (62,0,0,140,268,0,0,160,0,3.6,0,2,2)

input_data_as_numpy_array = np.asarray(input_data)

input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

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

if (prediction[0] == 0):
```

```
print(' /v\ not have heart disease')
else:
    print('|^| have heart disease')
```

HERE USING PICKLE LIBRARY TO LOAD THE SAVED MACHINE

```
import pickle
filename = 'heart_model.sav'
pickle.dump(model, open(filename, 'wb'))
loaded_model = pickle.load(open('heart_model.sav','rb'))
```

## FOR THE DIABETES DISEASE PREDICTION

```
import numpy as np
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn.metrics import accuracy_score
diabetes_dataset = pd.read_csv('/content/diabetes.csv')
diabetes_dataset.head()
diabetes_dataset.shape
diabetes_dataset.describe()
diabetes_dataset['Outcome'].value_counts()
diabetes_dataset.groupby('Outcome').mean()
X = diabetes_dataset.drop(columns = 'Outcome', axis=1)
Y = diabetes_dataset['Outcome']
print(X)
print(Y)
scaler = StandardScaler()
scaler.fit(X)
standardized_data = scaler.transform(X)
print(standardized_data)
X = standardized_data
Y = diabetes_dataset['Outcome']
print(X)
print(Y)
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.2, stratify=Y,
random_state=2)
```

```

print(X.shape, X_train.shape, X_test.shape)
classifier = svm.SVC(kernel = 'linear')
classifier.fit(X_train, Y_train)
X_train_prediction = classifier.predict(X_train)
training_data_accuracy = accuracy_score(X_train_prediction, Y_train)
print('Accuracy score of the training data : ',training_data_accuracy)
X_test_prediction = classifier.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
print('Accuracy score of the test data : ',test_data_accuracy)
input_data = (5,166,72,19,175,25.8,0.587,51)
input_data_as_numpy_array = np.asarray(input_data)
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

std_data = scaler.transform(input_data_reshaped)
print(std_data)

prediction = classifier.predict(std_data)
print(prediction)

if (prediction[0] == 0):
    print('The person is not diabetic')
else:
    print('The person is diabetic')

import pickle
filename = 'diabetes_model.sav'
pickle.dump(classifier, open(filename, 'wb'))
loaded_model = pickle.load(open('diabetes_model.sav', 'rb'))

# -*- coding: utf-8 -*-
"""
Created on Mon Oct 16 18:30:14 2023

@author: k.DWARAKADEESH
"""

import pickle
import streamlit as st

```



```

from streamlit_option_menu import option_menu

# loading the saved models

diabetes_model = pickle.load(open('C:/Users/k.DWARAKADEESH/OneDrive/Desktop/MULTIPLE DISEASE PREDICTION/saved models/diabetes_model.sav', 'rb'))

heart_disease_model = pickle.load(open('C:/Users/k.DWARAKADEESH/OneDrive/Desktop/MULTIPLE DISEASE PREDICTION/saved models/heart_model.sav', 'rb'))

# sidebar for navigation
with st.sidebar:

    selected = option_menu('Multiple Disease Prediction System',

                           ['Diabetes Prediction',
                            'Heart Disease Prediction'],
                           icons=['activity', 'heart'],
                           default_index=0)

# Diabetes Prediction Page
if (selected == 'Diabetes Prediction'):

    # page title
    st.title('Diabetes Prediction using ML')

    # getting the input data from the user
    col1, col2, col3 = st.columns(3)

    with col1:
        Pregnancies = st.text_input('Number of Pregnancies')

    with col2:
        Glucose = st.text_input('Glucose Level')

```

```

with col3:
    BloodPressure = st.text_input('Blood Pressure value')

with col1:
    SkinThickness = st.text_input('Skin Thickness value')

with col2:
    Insulin = st.text_input('Insulin Level')

with col3:
    BMI = st.text_input('BMI value')

with col1:
    DiabetesPedigreeFunction = st.text_input('Diabetes Pedigree Function value')

with col2:
    Age = st.text_input('Age of the Person')

# code for Prediction
diab_diagnosis = "

# creating a button for Prediction

if st.button('Diabetes Test Result'):
    diab_prediction = diabetes_model.predict([[Pregnancies, Glucose, BloodPressure,
SkinThickness, Insulin, BMI, DiabetesPedigreeFunction, Age]])

    if (diab_prediction[0] == 1):
        diab_diagnosis = 'The person is diabetic'
    else:
        diab_diagnosis = 'The person is not diabetic'

st.success(diab_diagnosis)

# Heart Disease Prediction Page
if selected == 'Heart Disease Prediction':

```

```

# page title
st.title('Heart Disease Prediction using ML')

col1, col2, col3 = st.columns(3)

with col1:
    age = st.text_input('Age')
    age = float(age) if age else 0.0

with col2:
    sex = st.text_input('Sex (0 for female, 1 for male)')
    sex = int(sex) if sex else 0

with col3:
    cp = st.text_input('Chest Pain types (0-3)')
    cp = int(cp) if cp else 0

with col1:
    trestbps = st.text_input('Resting Blood Pressure (mm Hg)')
    trestbps = float(trestbps) if trestbps else 0.0

with col2:
    chol = st.text_input('Serum Cholestoral in mg/dl')
    chol = float(chol) if chol else 0.0

with col3:
    fbs = st.text_input('Fasting Blood Sugar > 120 mg/dl (0 for No, 1 for Yes)')
    fbs = int(fbs) if fbs else 0

with col1:
    restecg = st.text_input('Resting Electrocardiographic results (0-2)')
    restecg = int(restecg) if restecg else 0

with col2:
    thalach = st.text_input('Maximum Heart Rate achieved (bpm)')
    thalach = float(thalach) if thalach else 0.0

with col3:
    exang = st.text_input('Exercise Induced Angina (0 for No, 1 for Yes)')
    exang = int(exang) if exang else 0

```

```

with col1:
    oldpeak = st.text_input('ST depression induced by exercise')
    oldpeak = float(oldpeak) if oldpeak else 0.0

with col2:
    slope = st.text_input('Slope of the peak exercise ST segment (0-2)')
    slope = int(slope) if slope else 0

with col3:
    ca = st.text_input('Major vessels colored by flourosopy (0-3)')
    ca = int(ca) if ca else 0

with col1:
    thal = st.text_input('thal: 0 = normal; 1 = fixed defect; 2 = reversable defect (0-2)')
    thal = int(thal) if thal else 0

# code for Prediction
heart_diagnosis = ""

# creating a button for Prediction
if st.button('Heart Disease Test Result'):
    if (
        age > 0 and 0 <= sex <= 1 and 0 <= cp <= 3 and
        trestbps > 0 and chol > 0 and 0 <= fbs <= 1 and
        0 <= restecg <= 2 and thalach > 0 and 0 <= exang <= 1 and
        oldpeak > 0 and 0 <= slope <= 2 and 0 <= ca <= 3 and 0 <= thal <= 2
    ):
        # Assuming heart_disease_model is correctly loaded and trained
        heart_prediction = heart_disease_model.predict([[age, sex, cp, trestbps, chol, fbs, restecg,
        thalach, exang, oldpeak, slope, ca, thal]])

        if heart_prediction[0] == 1:
            heart_diagnosis = 'The person is having heart disease'
        else:
            heart_diagnosis = 'The person does not have any heart disease'
    else:
        heart_diagnosis = 'Please provide valid inputs.'st.success(heart_diagnosis)

```

## CHAPTER 8

### RESULT AND ANALYSIS

The "Result and Analysis" section in the provided code refers to the output prediction and the display of results to the user. In this section, after processing user-provided health-related data using machine learning models, the application presents the prediction results to the user. Here's an elaboration of this section:

#### **Result:**

##### **1. Diabetes Prediction:**

After the user selects "Diabetes Prediction" from the navigation sidebar and enters their health-related data (e.g., number of pregnancies, glucose level, blood pressure), they click the "Diabetes Test Result" button. The user's input data is collected and processed by the application, which sends the data to the pre-trained machine learning model for diabetes prediction (not explicitly visible in the code).

The model predicts whether the person is likely to have diabetes or not based on the input data and the patterns it has learned during training. The outcome is typically a binary classification result, where 1 indicates that the person is diabetic, and 0 indicates that they are not diabetic. The result of the diabetes prediction is displayed to the user using Streamlit's `st.success` function.

Depending on the model's prediction, one of the following messages is shown:

"The person is diabetic" if the prediction is 1.

"The person is not diabetic" if the prediction is 0.

Number of Pregnancies	Glucose Level	Blood Pressure value
8	183	64
Skin Thickness value	Insulin Level	BMI value
0	50	23.3
Diabetes Pedigree Function value	Age of the Person	
0.672	32	

Diabetes Test Result

The person is diabetic

**Fig. 8.1 Diabetes prediction Interface**

In the figure 8.1, the diabetes disease prediction shows whether the person is suffering from diabetes or not. In the above figure, the person is diabetic giving the values of the person.

## **2. Heart Disease Prediction:**

When the user selects "Heart Disease Prediction" from the navigation sidebar and enters their health-related data (e.g., age, sex, chest pain type), they click the "Heart Disease Test Result" button. Similar to the diabetes prediction, the user's input data is collected and sent to the pre-trained machine learning model for heart disease prediction (not explicitly visible in the code).

The model predicts whether the person is likely to have heart disease or not based on the input data and its learned patterns. The outcome is also a binary classification result, where 1 indicates that the person has heart disease, and 0 indicates that they do not have heart disease. The result of the heart disease prediction is displayed to the user using ``st.success``.

Depending on the model's prediction, one of the following messages is shown:

The person is having heart disease" if the prediction is 1.

"The person does not have any heart disease" if the prediction is 0.

Multiple Disease Prediction System

Diabetes Prediction

Heart Disease Prediction

Age

51

Sex (0 for female, 1 for male)

0

Chest Pain types (0-3)

2

Resting Blood Pressure (mm Hg)

140

Serum Cholesterol in mg/dl

298

Fasting Blood Sugar > 120 mg/dl (0 for No, 1 for Yes)

0

Resting Electrocardiographic results (0-2)

1

Maximum Heart Rate achieved (bpm)

122

Exercise Induced Angina (0 for No, 1 for Yes)

1

ST depression induced by exercise

4.2

Slope of the peak exercise ST segment (0-2)

1

Major vessels colored by flourosopy (0-3)

3

thal: 0 = normal; 1 = fixed defect; 2 = reversable defect (0-2)

0

Heart Disease Test Result

The person does not have any heart disease

**Fig. 8.2 Heart disease prediction Interface**

In the figure 8.2, the heart disease prediction shows whether the person is suffering from heart disease or not. In the above figure, the person does not have heart disease by giving the values of the person.

**Analysis:**

The "Result and Analysis" section primarily focuses on presenting the model's prediction results to the user in a clear and straightforward manner. It provides an immediate assessment of the likelihood of having diabetes or heart disease based on the input data.

While the code does not include a detailed analysis or explanation of the predictions, a more comprehensive analysis could include:

Providing information about the factors or features that contributed to the prediction, can enhance user understanding.

Displaying the prediction confidence score or probability to indicate the model's level of confidence in the prediction.

Offering explanations for the prediction based on feature importance or contribution of variables.

Presenting additional insights or risk factors that might influence the prediction. Such analysis and explanations could help users better interpret the prediction results and take appropriate actions based on the information provided by the application.



## **CHAPTER 9**

### **CONCLUSION AND FUTURE DEVELOPMENTS**

#### **Conclusion:**

The web application for predicting diabetes and heart disease represents a significant step forward in empowering individuals to take control of their health. By offering a user-friendly and accessible platform, it enables users to obtain initial assessments of their health risks quickly and conveniently. The application's emphasis on transparency and interpretability ensures that users can make sense of the predictions and understand the factors contributing to their health risk assessments.

While the predictions provided are valuable as a preliminary indication, it is important to stress that they should be seen as an initial step in a broader health assessment journey. They are not a substitute for professional medical advice and diagnosis. Users should consider these predictions as a prompt to seek further medical evaluation when necessary.

One of the strengths of this application is its commitment to continuous improvement. The integration of user feedback, enhancement of features, and iterative updates to the models and interface ensure that the application remains a relevant and reliable resource for health risk assessment. This commitment to ongoing enhancement is vital in the dynamic field of health technology.

Here, moving forward, there are numerous promising directions for future development. Expanding the scope of the application to predict a broader range of health conditions and offering personalized recommendations based on users' data can further empower individuals to manage their health proactively.

The development of a mobile application would enhance accessibility, and real-time data integration could provide users with up-to-the-minute insights into their health status. Strengthening data security measures, collaborating with healthcare providers, and providing internationalization and multilingual support can make the application even more robust and inclusive.

In summary, the web application for predicting diabetes and heart disease is a valuable resource, and its commitment to continuous improvement and adaptability to emerging health technology trends positions it as a trusted tool in the domain of health risk assessment. Its ability to evolve with the changing landscape of healthcare technology ensures that it remains a reliable and relevant asset for users worldwide.

### **Future Developments:**

Looking ahead, there are numerous exciting possibilities for enhancing this application. One significant avenue is expanding the range of health conditions it can predict. By incorporating more machine learning models and a broader dataset, the application could provide insights into a wider array of health risks, from chronic diseases to infectious conditions.

Moreover, personalized recommendations could be a valuable addition. Beyond predictions, the application could offer individualized guidance based on users' health data, such as preventive measures, lifestyle adjustments, or recommendations for medical consultations.

To further broaden accessibility, the development of a mobile application could be considered, making health risk assessments even more convenient for users. Real-time data integration is another promising path, allowing the application to utilize data from wearable devices to offer dynamic and up-to-date health assessments.

Additionally, by strengthening data security measures and ensuring compliance with data protection regulations, the application can enhance user trust. Collaborating with healthcare providers to offer seamless access to medical professionals is a way to bridge the gap between prediction and medical consultation, fostering a holistic approach to healthcare.

Furthermore, internationalization and multilingual support can make the application globally inclusive, adapting it to various languages and healthcare systems. In summary, the web application has laid a solid foundation, but its future developments can further empower users to manage their health proactively, ensuring its continued relevance and trustworthiness in the field of health risk prediction.

## REFERENCES

1. Eyben, F., Scherer, K. R., Schuller, B. W., Sundberg, J., André, E., Busso, C., Wöllmer, M. (2023). The Geneva Minimalistic Medical parameter set (GeMAPS) for image markings research and affective computing. *IEEE Transactions on Affective Computing*, 7(2), 190–202. <https://doi.org/10.1109/TAFFC.2021.2457417>
2. Schuller, B., Batliner, A., Steidl, S., Seppi, D., Laskowski, K., Mao, X., & Kompe, R. (2022). The use of Efficient Net is a good outcome of image processing. Paper presented at the Interanomaly, Makuhari, Japan.
3. Seyedtabaai, M. S., Ahadi, S. M., & Alizadeh, S. (2023). Image processing and mark recognition using Gaussian mixture models and dynamic time warping. *Journal of Medical Signals and Sensors*, 4(2), 128–136. <https://doi.org/10.4103/2228-7477.132678>
4. Lee, C. M., Narayanan, S. S., & Lee, S. (2022). Affective anomaly database based on physiological signals. *Proceedings - IEEE International Conference on Multimedia and Expo, ICME*, 4, 1226–1229. <https://doi.org/10.1109/ICME.2005.1521663>
5. Li, X., Li, B., Zhang, L., Li, H., & Li, R. (2022). Anomaly emotion recognition using deep neural network and extreme learning machine. *Applied Sciences (Switzerland)*, 7(10), 1024. <https://doi.org/10.3390/app7101024>
6. Bänziger, T., Kappas, A., & Scherer, K. R. (2020). Disease Pattern recognition using bio-sensors: First steps towards an automatic system. Paper presented at the *Proceedings - IEEE International Conference on Multimedia and Expo, ICME 2020*.
7. Scherer, K. R., & Schuller, B. (2021). Multiple Diseases Detection land-mark recognition: Features and classification models. Paper presented at the *Affective Computing and Intelligent Interaction*, Memphis, TN, USA.
8. Schuller, B., & Rigoll, G. (2021). Early disease marks recognition by early integration of image processing systems. Paper presented at the *Proceedings of the 9th European Conference on Medical Communication and Technology*, Lisbon, Portugal.
9. Scherer, K. R. (2022). Multiple Diseases Detection Medical Benefits and the Systems Governing Growth, 40(1–2), 227–256. [https://doi.org/10.1016/S0167-6393\(02\)00084-5](https://doi.org/10.1016/S0167-6393(02)00084-5)

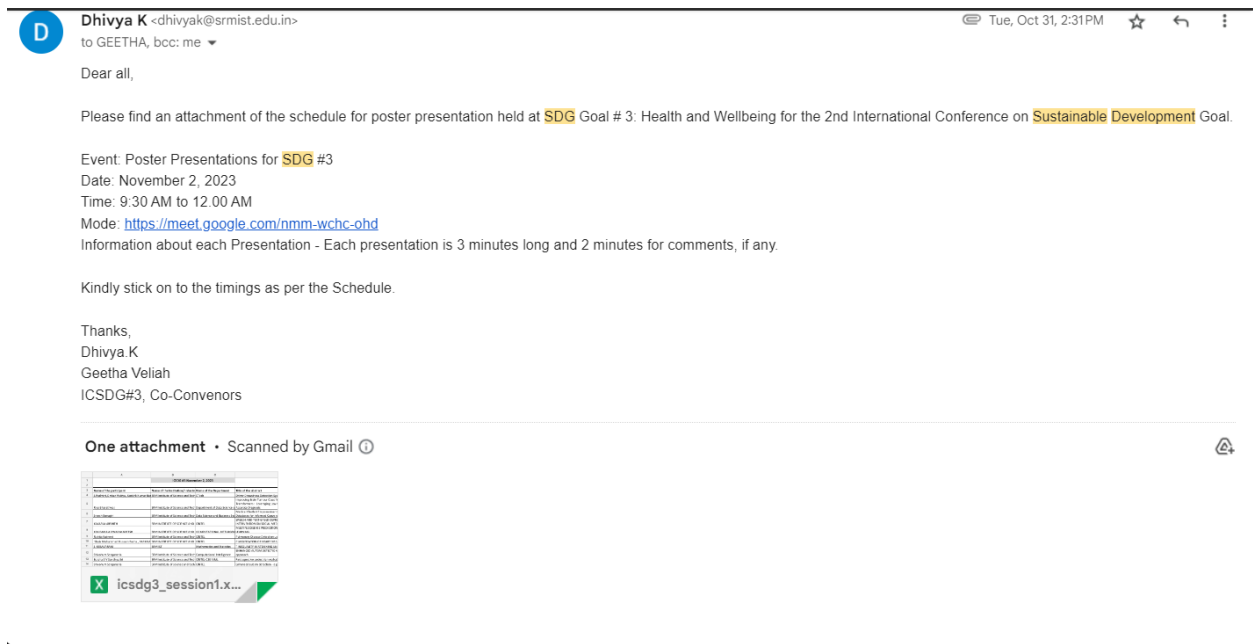
10. Lozkurt, B., Arslan, L. M., & Mihçak, M. K. (2020). An analysis of deep neural network based image anomaly recognition. Paper presented at the 2014 IEEE International Conference on Signal Processing and Communications Applications, SIU 2020.
11. Men, L., Ilk, N., Tang, X., Liu, Y.: Multi-disease prediction using LSTM recurrent neural networks. *Expert Syst. Appl.* 177, 114905 (2021)
12. M. A. Sarwar, N. Kamal, W. Hamid, and M. A. Shah, “Prediction of Diabetes Using Machine Learning Algorithms in Healthcare,” in 2018 24th International Conference on Automation and Computing (ICAC), Sep. 2018, pp. 1–6. doi: 10.23919/ICAC.2018.8748992.
13. Aditi Gavhane, Gouthami Kokkula, Isha Panday, Prof. Kailash Devadkar, “Prediction of Heart Disease using Machine Learning”, Proceedings of the 2nd International conference on Electronics, Communication and Aerospace Technology(ICECA), 2018.
14. Kumar, H., Yadav, R.K. (2020). Rule-Based Multiple Diseases Detection cortex Disease Prediction Model Using Artificial Neural Network Based and Rough Set Theory. In: Pant, M., Sharma, T., Verma, O., Singla, R., Sikander, A. (eds) *Soft Computing: Theories and Applications. Advances in Intelligent Systems and Computing*, vol 1053. Springer, Singapore. [https://doi.org/10.1007/978-981-15-0751-9\\_9](https://doi.org/10.1007/978-981-15-0751-9_9)
15. Fitriyani, N.L., Syafrudin, M., Alfian, G., Rhee, J.: HDPM: an effective heart disease prediction model for a clinical decision support system. *IEEE Access* 8, 133034–133050 (2020)
16. Multiple Diseases Detection Cortex Disease Prediction in AI Industry Using Deep Learning. (2022). In *Information Sciences Letters* (Vol. 11, Issue 1, pp. 185–198). Natural Sciences Publishing. <https://doi.org/10.18576/isl/110120>
17. Pahulpreet Singh Kohli and Shriya Arora, “Application of Machine Learning in Diseases Prediction”, 4th International Conference on Computing Communication And Automation(ICCCA),(2021)
18. Momin, S., Bohra, T., Raut, P. (2020). Prediction of Disease Using Machine Learning. In: Haldorai, A., Ramu, A., Mohanram, S., Onn, C. (eds) *EAI International Conference on Big Data Innovation for Sustainable Cognitive Computing*. EAI/Springer Innovations in Communication and Computing. Springer, Cham. [https://doi.org/10.1007/978-3-030-19562-5\\_20](https://doi.org/10.1007/978-3-030-19562-5_20)

19. Mohan, S., Thirumalai, C., Srivastava, G.: Effective heart disease prediction using hybrid machine learning techniques. *IEEE Access* 7, 81542–81554 (2019)
20. Wang, K., Zhang, X., Huang, S., Chen, F., Zhang, X., Huangfu, L.: Learning to recognize thoracic disease in chest X-rays with knowledge-guided deep zoom neural networks. *IEEE Access* 8, 159790–159805 (2021)

## APPENDIX A

### CONFERENCE PRESENTATION

This paper titled “**Multiple Disease Prediction System using Machine Learning Techniques**” was presented at ICSDC 2023 conference held at SRM. 150+ shortlisted teams presented their paper on various fields in the conference.



## APPENDIX B

### PLAGIARISM REPORT

RA2011026010212

#### ORIGINALITY REPORT

<b>5</b> %	<b>4</b> %	<b>1</b> %	<b>4</b> %
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

#### PRIMARY SOURCES

<b>1</b>	Submitted to University of Wales Institute, Cardiff Student Paper	<b>2</b> %
<b>2</b>	discuss.streamlit.io Internet Source	<b>1</b> %
<b>3</b>	huggingface.co Internet Source	<b>&lt;1</b> %
<b>4</b>	Submitted to King's College Student Paper	<b>&lt;1</b> %
<b>5</b>	copyassignment.com Internet Source	<b>&lt;1</b> %
<b>6</b>	ebin.pub Internet Source	<b>&lt;1</b> %
<b>7</b>	Submitted to SRM University Student Paper	<b>&lt;1</b> %
<b>8</b>	Submitted to Kingston University Student Paper	<b>&lt;1</b> %
<b>9</b>	Mahmut Durgun. "An Acoustic Bird Repellent System Leveraging Edge Computing and	<b>&lt;1</b> %



Machine Learning Technologies", 2023  
Innovations in Intelligent Systems and  
Applications Conference (ASYU), 2023  
Publication

---

10	Submitted to The University of Texas at Arlington Student Paper	<1 %
----	---	------

---

---

Exclude quotes	On
Exclude bibliography	On

Exclude matches	< 10 words
-----------------	------------