“ML-Driven Early Detection for Optimal Health – Empowering You with Accurate Predictive Health Analytics”

# SUMMER INTERNSHIP SOPHOMORES PROJECT SPJ 2001­­­­­­

**AP20232410000130**

## Summer Id: 20240007

**First Review**

# UNDER THE GUIDANCE OF

Prof. Anil Vitthalrao Turukmane

**DEPARTMENT OF SCOPE (Speci. Networking And Security)**

# SUBMITTED BY

Madhu Alapaka - 21MIS7022.

Kothamasu Kaarthikeya - 21MIS7039.

Pulluru Nikhilesh - 21MIS7087.

**DEPARTMENT OF SCOPE(Integ M.TechSpec.in Software Engineering)**



[**SYSTEM REQUIREMENT SPECIFICATIONS (“S.R.S”)**](1)First-Reveiw%20-)

“ML-Driven Early Detection for Optimal Health – Empowering You with Accurate Predictive Health Analytics”

**1. Purpose of the System:**

The purpose of the Multiple Disease Prediction System using Machine Learning is to provide early detection and risk assessment for multiple diseases, such as Diabetes, Heart Disease, and Parkinson's Disease. This system leverages advanced machine learning algorithms to analyze patient data and predict the likelihood of these conditions, enabling timely medical intervention and personalized healthcare management. The deployment interface is built on Streamlit, offering a user-friendly web application where users can input health data and receive instant predictions. This proactive approach aims to improve health outcomes and reduce the burden on healthcare systems by facilitating early diagnosis and treatment.

Certainly! Here is a detailed description of the system requirements for the Multiple Disease Prediction System using machine learning techniques:

* **System Requirements Specification:**

**Software Requirements**

1. Operating System

- Windows 10/11 (64-bit): Ensure compatibility with various software tools and frameworks.

2. Programming Language

- Python: Primary language for implementing machine learning algorithms and developing the system.

3. Machine Learning Framework

- ML Modules, Scikit-Learn: Libraries for building and training machine learning models.

4. IDEs Tool

- Visual Studio Code: An extensible code editor for development.

- Jupyter Notebook: An interactive environment for running and visualizing code.

5. Version Control System

- Git: A system for tracking changes in source code during software development.

6. Data Manipulation

- NumPy: A library for numerical operations.

- Pandas: A library for data manipulation and analysis.

7. Model Evaluation

- Streamlit: A framework for building interactive web applications to display model results.

8. Package Management and Build Tools

- OpenCV: If used, it provides tools for image processing, which might be relevant for certain data preprocessing tasks.

* **Hardware Requirements**

1. Processor

- Intel i5/i7 (8th generation or newer recommended): Provides adequate performance for running machine learning algorithms efficiently.

2. Memory (RAM)

- At least 4 GB or More: Sufficient memory for handling data processing tasks and running models. More RAM is recommended for better performance.

3. Storage

- SSD with at least 250 GB free space: Enough storage for software, project files, and datasets. An SSD ensures faster read/write speeds compared to traditional HDDs.

4. Hardware

- Laptop: A portable and versatile device for development and demonstration purposes.

5. GPU

- NVIDIA RTX 3080 with 2GB VRAM: A powerful GPU to accelerate machine learning model training, especially deep learning models.

6. Cooling

- High-performance CPU cooler and adequate case fans or an external cooling pad: Ensures the laptop or desktop remains cool during intensive processing tasks.

7. Ports and Connectivity

- Wi-Fi 6: Provides faster and more reliable wireless connectivity for downloading datasets and software packages.

8. Keyboard and Touchpad

- Backlit Keyboard: Useful for working in low-light environments.

- Precision Touchpad or External Mouse: Ensures accurate control during development.

- High Display (HD): For clear visualization of code, data, and results.

This detailed description covers both the software and hardware requirements necessary to develop and run a Multiple Disease Prediction System using machine learning techniques.

**Definitions, Acronyms, and Abbreviations:**

* **ML:** Machine Learning
* **GUI:** Graphical User Interface
* **API:** Application Programming Interface
* **SRS:** Software Requirements Specification
* **SDA:** System Design Architecture

**References:**

* Python and Streamlit official documentation
* Machine learning research papers on disease prediction
* Medical literature on Diabetes, Heart Disease, and Parkinson's Disease

**2. Problems in the Existing System :**

* **Lack of Early Detection:** Current systems may not provide timely detection of multiple diseases, leading to delayed treatment, which can result in worsening conditions and reduced treatment effectiveness.
* **Limited Access to Diagnostics:** Not all patients have access to comprehensive diagnostic facilities due to geographical or financial constraints, limiting their ability to get timely and accurate diagnoses.
* **Fragmented Data:** Patient health data is often scattered across various sources, making it difficult to aggregate and analyze comprehensively for predictive analysis. This fragmentation can lead to incomplete or inaccurate predictions.
* **Complexity of Use:** Existing predictive tools can be complex and not user-friendly, limiting their accessibility to healthcare providers and patients. This complexity can result in the underutilization of these tools.
* **Limited Accuracy**: Traditional methods may not leverage advanced machine learning techniques, resulting in lower prediction accuracy. This can lead to misdiagnoses or missed diagnoses, affecting patient care.

**3. Solution to These Problems:**

* + **Early Detection:** The system provides early disease detection, allowing for timely medical interventions and better health outcomes. This is achieved through the use of advanced machine learning algorithms that analyze patient data to identify early signs of disease.
  + **Accessibility:** An online interface (Streamlit) that can be accessed from anywhere allows users to input their data and get predictions. This increases the reach of diagnostic tools, making them available to a wider audience.
  + **Integrated System:** The proposed system integrates data from various sources, providing a comprehensive dataset for analysis. This integration improves the quality and completeness of the data used for predictions.
  + **User-Friendly Interface:** Deploys a web application using Streamlit for easy data input and result visualization. The intuitive interface ensures that both healthcare providers and patients can use the system with minimal training.
  + **Machine Learning Algorithms:** Utilizes advanced machine learning algorithms to improve prediction accuracy. These algorithms are trained on large datasets to learn patterns and correlations that indicate the presence of disease.

**4. Scope of the Project:**

* + **Disease Coverage:** Focuses on predicting Diabetes, Heart Disease, and Parkinson's Disease, with potential expansion to other diseases. The system is designed to be scalable and adaptable, allowing for the addition of new disease models in the future.
  + **User Base:** Designed for individuals and healthcare providers. The system aims to serve a diverse user base, including patients seeking self-assessment and healthcare professionals looking for diagnostic support.
  + **Geographical Reach:** Global, as the web application can be accessed from any location with internet connectivity. This broad reach ensures that the system can benefit users worldwide, regardless of their location.
  + **Scalability:** Capable of integrating additional diseases and scaling the infrastructure as needed. The system's architecture is designed to handle increased user load and data volume, ensuring consistent performance even as demand grows.

**5. Functional Components of the Project**

* **Data Ingestion and Preprocessing Module**

1. Data Acquisition: Collects raw data from various sources such as medical datasets (e.g., UCI Machine Learning Repository, CDC, WHO). The data includes patient demographics, medical history, lab results, and other relevant health indicators.

2. Data Refinement: Cleans and preprocesses data to ensure quality. This includes handling missing values, normalizing data, and removing outliers to prepare the data for analysis.

3. Feature Creation: Develop new features to enhance model performance. This involves selecting relevant variables and creating new features through techniques like feature extraction and engineering.

* **Machine Learning Model Building Module**

1. Model Selection: Chooses appropriate machine learning algorithms based on the nature of the disease and the data characteristics. Examples include Logistic Regression, Support Vector Machines (SVM), Random Forest, and Deep Learning models.

2. Model Training: Trains the chosen model on a portion of the data. This involves splitting the data into training and testing sets and using the training data to build the model.

* **Model Validation Module**

1. Validation Engine: Conducts thorough validation of predictive models. This includes techniques like cross-validation to ensure the model's generalizability and robustness.

* **User Interface with Streamlit Module**

1. Interactive Dashboard: Provides a user-friendly interface for data input and result visualization. Users can easily navigate through the dashboard to enter their health data and view predictions.

2. User Input Interface: Facilitates user data entry and parameter adjustments. The interface is designed to be intuitive and accessible, minimizing the effort required from users.

3. Prediction Display: Shows prediction outcomes and risk assessments interactively. Users receive clear and concise information about their health risks, along with recommendations for further action.

* **Reporting Module**

1. Report Generation: Generates comprehensive reports on model performance metrics. These reports can be used by healthcare providers to understand the model's accuracy and reliability.

* **Disease-Specific Predictive Models Module**

1. Heart/Cardio Disease Prediction: Utilizes algorithms like Logistic Regression, Decision Trees, Random Forest, and SVM. Key factors include cholesterol levels, blood pressure, age, and smoking status.

2. Diabetes Prediction: Uses models such as Logistic Regression, K-Nearest Neighbors, and Random Forest. Important predictors include BMI, glucose levels, insulin levels, and age.

3. Parkinson's Disease Prediction: Implements techniques like SVM, Random Forest, and Neural Networks. Critical features include motor symptoms, voice patterns, and other clinical assessments.

* **Deployment Module**

1. Integration: Integrates the final models into a user-friendly web application using Streamlit. The integration ensures seamless interaction between the model and the user interface.

2. Interactive Platform: Provides an interactive platform for users to input their health data and receive instant predictions. The platform is designed to be responsive and accessible, ensuring a smooth user experience.

**6. Study of the System:**

1. **Functional Requirements:**

**6.1 Main Functional Requirements:**

* **User Input:**  
  The system shall provide forms for users to input their health data.
* **Data Preprocessing:**  
  The system shall preprocess the input data, handling missing values and scaling features as necessary.
* **Disease Prediction:**  
  The system shall use trained machine learning models to predict the risk of Diabetes, Heart Disease, and Parkinson's Disease.
* **Result Display:**  
  The system shall display the prediction results in an easy-to-understand format, including risk levels and relevant health advice.

**6.2 Classification of functional requirements:**

* **Data Sources**

1. Data Selection: Analyze and select datasets that provide diverse and comprehensive medical data. The datasets should cover a wide range of patient demographics and health conditions to ensure the model's applicability.

2. Data Quality: Ensure datasets have high-quality, labeled data for training machine learning models. Data quality is crucial for building accurate and reliable predictive models.

* **Machine Learning Algorithms**

1. Algorithm Evaluation: Evaluate different algorithms for their suitability to predict each specific disease. This involves comparing the performance of various models to identify the best approach.

2. Model Comparison: Compare model performances to select the best approach for each disease. Metrics such as accuracy, precision, and recall are used to assess the models.

* **Web Framework**

1. Streamlit Assessment: Assess Streamlit for its capabilities in developing interactive and user-friendly web applications. Streamlit's simplicity and flexibility make it a suitable choice for deploying machine learning models.

2. Framework Support: Ensure the framework supports necessary integrations and provides a smooth user experience. The framework should facilitate easy deployment and maintenance of the application.

* **User Interaction**

1. Interface Design: Design an intuitive user interface that simplifies data input and result interpretation. The interface should be user-centric, focusing on ease of use and accessibility.

2. Usability Testing: Conduct usability testing to refine the user experience. Feedback from users helps identify areas for improvement and ensures the application meets user needs.

**6.3 External Interface Requirements:**

* **User Interface:**  
  The system shall have a Streamlit-based GUI with input forms and result display sections.
* **Software Interfaces:**  
  The backend shall be built in Python and communicate with the frontend through APIs.

**6.4 System Features:**

* **User Registration and Login:**  
  (Optional) Users can register and log in to save their health data and prediction history.
* **Health Data Input:**  
  Users can enter health-related metrics such as age, BMI, blood pressure, etc.
* **Prediction Results:**  
  Display of the predicted risk levels along with actionable insights.

1. **Non-Functional Requirements**:

1. Performance: The application should respond to user inputs and generate predictions within an acceptable time frame.

2. Usability: The Streamlit interface should be user-friendly and easy to navigate.

3. Scalability: Although there is no database, the application should handle different sizes of input data efficiently.

4. Reliability: The application should consistently produce accurate predictions and visualizations without errors.

5. Maintainability: The code should be well-documented and modular to allow for easy updates and maintenance.

6. Portability: The application should be easily deployable on different systems with minimal configuration.

**Input / Output and Major Functions of the Multiple Disease Prediction System:**

#### **Main Inputs:**

1. **Patient Data**
   * Demographic information (e.g., age, gender)
   * Clinical data (e.g., lab results, medical history)
   * Lifestyle factors (e.g., smoking status, physical activity)
2. **Machine Learning Models**
   * Trained models for disease prediction (e.g., Diabetes, Heart Disease, Parkinson's Disease)
   * Models integrated with algorithms and parameters for inference

#### **Main Outputs:**

1. **Prediction Results**
   * Likelihood or risk assessment of diseases (e.g., probability scores, risk categories)
   * Diagnostic outcomes based on input data

**7. Performance Requirements**

**1. Accuracy**

Goal: Achieve a high predictive accuracy (at least 85%) for each disease model.

Measure: Calculate and monitor the accuracy metric during model evaluation and after deployment to ensure consistent performance.

**2. Response Time**

Goal: Ensure the system provides predictions within 5 seconds of data input.

Measure: Implement efficient data processing and model inference techniques to minimize latency.

**3. Scalability**

Goal: Design the platform to handle multiple concurrent users without performance degradation.

Measure: Use scalable cloud infrastructure and load balancing to manage increased user load and data volume effectively.

**4. Reliability**

Goal: Maintain consistent performance with minimal downtime.

Measure: Ensure high availability through robust infrastructure, regular maintenance, and monitoring. Target uptime should be at least 99.9%.

**5. Robustness**

Goal: Ensure the system can handle a wide range of data inputs and maintain performance.

Measure: Implement extensive testing, including edge cases and stress tests, to validate system robustness.

**6. User Satisfaction**

Goal: Achieve a high level of user satisfaction with the system's performance and usability.

Measure: Collect and analyze user feedback regularly to identify and address any performance-related issues. Aim for a user satisfaction rate of at least 90%.

**8. Feasibility Report:**

* **Technical Feasibility**

1. Availability of Tools: The system is technically feasible given the availability of machine learning frameworks (e.g., TensorFlow, Keras), data manipulation libraries (e.g., NumPy, Pandas), and deployment tools (e.g., Streamlit). These tools provide the necessary infrastructure for developing and deploying the system.

2. Computational Resources: The necessary computational resources, including high-performance laptops and cloud platforms, are accessible. These resources ensure the system can handle large datasets and complex computations.

* **Economic Feasibility**

1. Cost Management: The project is economically feasible as the costs involved in development, deployment, and maintenance are manageable within the allocated budget. Open-source tools and frameworks help reduce costs.

2. Healthcare Savings: Potential savings in healthcare costs due to early disease detection and management can offset the initial investment. Early intervention can reduce the need for expensive treatments and improve patient outcomes.

* **Operational Feasibility**

1. Workflow Integration: The system is operationally feasible as it aligns with the workflows of healthcare providers and patients. The system is designed to integrate smoothly into existing healthcare processes.

2. User-Friendly Interface: The user-friendly interface ensures ease of use, making it accessible to both patients and healthcare providers. Minimal training is required to use the system effectively.

* **Schedule Feasibility**

1. Realistic Timeline: Develop a realistic timeline for data collection, model training, system testing, and deployment. A well-planned schedule ensures that the project stays on track and meets deadlines.

2. Resource Allocation: Allocate sufficient resources to each phase to ensure timely completion. Adequate staffing and budget allocation are essential for the project's success.

### **Overall Description:**

**1 Product Perspective:** The system is designed as a standalone web application with modular components for each disease prediction. It will utilize various machine learning models tailored to the specific characteristics of each disease.

**2 Product Functions:**

* User registration and login (optional)
* Health data input (e.g., age, weight, glucose levels, etc.)
* Data preprocessing and feature engineering
* Disease risk prediction using ML models
* Display of prediction results with interpretive insights

**3 User Characteristics:**

* **End-users:** Individuals looking for early disease detection

**4 Constraints:**

* Requires internet access
* Dependence on the accuracy of user-provided data
* Performance is tied to the efficiency of machine learning models

**5 Assumptions and Dependencies:**

* Accurate health data input by users
* Use of up-to-date medical and technical resources for model training
* Dependencies include Python, Streamlit, scikit-learn, pandas, and other ML libraries