DISEASE PREDICTION USING MACHINE LEARNING

A PROJECT SYNOPSIS
for
Full Stack Development with Java (ID-202B)
Session (2025-26)

Submitted by

Harshita Chauhan
(202410116100090)
Himanshi Sharma
(202410116100091)
Gunjan Sharma
(202410116100080)

Submitted in partial fulfilment of the
Requirements for the Degree of

MASTER OF COMPUTER APPLICATION

Under the Supervision of
Ms. Divya Singhal
Assistant Professor

Submitted to

DEPARTMENT OF COMPUTER APPLICATIONS
KIET Group of Institutions, Ghaziabad
Uttar Pradesh-201206

2025

DISEASE PREDICTION USING MACHINE LEARNING

Harshita Chauhan
Himanshi Sharma
Gunjan Sharma

ABSTRACT

With the growing demand for accessible and efficient healthcare solutions, artificial intelligence has emerged as a key player in assisting medical diagnosis. This project explores the application of machine learning techniques for disease prediction based on symptoms provided by users. A system is developed using the Random Forest Classifier to analyze a large dataset containing symptoms and corresponding diseases, enabling accurate predictions.

The objective of this system is to assist users in identifying potential health conditions early, based on common symptoms, and to provide preliminary advice before consulting a healthcare professional. The model was trained on a comprehensive dataset and achieved high prediction accuracy through rigorous evaluation metrics like accuracy, precision, and recall.

A user-friendly graphical interface was built using Tkinter, allowing users to select symptoms through categorized dropdown menus (eyes, skin, tongue, hair, etc.) and receive real-time predictions. This system demonstrates how machine learning can complement traditional healthcare practices by enhancing early diagnosis and making health insights more accessible.

While the model is not intended to replace professional medical advice, it serves as an effective tool for educational purposes and preliminary health screening. Future enhancements may include the use of natural language processing, integration with medical databases, and deployment as a web or mobile application.

TABLE OF CONTENTS

1. Introduction

In recent years, the integration of artificial intelligence into the healthcare industry has revolutionized diagnostic processes. Machine learning (ML), a subfield of AI, offers predictive capabilities that can assist medical professionals in diagnosing diseases more accurately and rapidly. This project focuses on developing a disease prediction system using machine learning algorithms, specifically the Random Forest Classifier, to predict diseases based on user-input symptoms.

In today's fast-paced world, the timely and accurate diagnosis of diseases is critical to effective healthcare. However, many individuals often overlook early symptoms or delay seeking medical advice due to lack of awareness or access to healthcare services. This gap in early diagnosis can lead to serious health complications. In response to this challenge, technology—particularly artificial intelligence (AI) and machine learning (ML)—has emerged as a powerful tool to support the healthcare industry.

Machine learning, a subset of AI, focuses on developing algorithms that enable systems to learn patterns from data and make decisions or predictions without being explicitly programmed. In recent years, machine learning has shown great promise in various medical applications, such as image analysis, drug discovery, patient monitoring, and disease diagnosis.

This project explores the application of machine learning for disease prediction based on symptoms. By analyzing a large dataset of diseases and their associated symptoms, a predictive model can identify the most likely disease a user might have. This system is not designed to replace doctors or clinical diagnosis, but rather to assist individuals in understanding their symptoms and encouraging them to seek professional help when necessary.

The implementation involves training a machine learning model—specifically, a Random Forest Classifier—on a structured dataset. The model is then integrated into a user-friendly interface built with Tkinter, enabling users to input symptoms and receive predicted diseases in real time.

The significance of this project lies in its potential to:

- Promote early awareness of health conditions.
- Offer preliminary assessments in areas with limited medical resources.
- Reduce the burden on healthcare professionals for non-critical cases.

Through this project, we demonstrate how machine learning can be utilized to build smart, accessible, and assistive tools for personal health management.

2. Problem Statement

Accurate and timely diagnosis of diseases is a critical component of effective healthcare. However, many people either lack access to qualified medical professionals or delay seeking help due to uncertainty about the seriousness of their symptoms. In such cases, early signs of potentially serious conditions may go unnoticed, leading to delayed treatment and poorer health outcomes.

Traditional diagnosis methods rely heavily on a doctor's experience, physical examination, and diagnostic tests, which may not always be immediately accessible, especially in remote or underdeveloped areas. Furthermore, manual analysis of symptoms can be time-consuming and prone to human error, especially when symptoms are vague or overlap across multiple diseases.

Given these challenges, there is a pressing need for an intelligent system that can assist users by providing a preliminary diagnosis based on their symptoms. Machine learning, with its ability to recognize complex patterns in large datasets, presents a promising solution.

This project addresses the following core problems:

- **Lack of accessibility** to early diagnostic tools for individuals in remote or underserved areas.
- **Difficulty in identifying possible diseases** based on multiple, overlapping symptoms.
- **Need for a fast and scalable system** that can assist users in making informed decisions about seeking medical attention.

The goal is to design and implement a machine learning-based system that can:

- Accept a set of symptoms as input.
- Predict the most probable disease(s) based on historical medical data.
- Provide users with preliminary information and encourage professional medical consultation when necessary.

By addressing these problems, this project aims to improve early disease detection, enhance awareness, and support healthcare systems with an assistive diagnostic tool.

3. Objectives of the Project

The primary objective of this project is to develop an intelligent and user-friendly system that leverages machine learning techniques to predict diseases based on user-input symptoms. This system is intended to assist users in understanding potential health conditions and to encourage timely medical consultation.

Specific Objectives:

1. **To build a machine learning model** (using algorithms such as Random Forest) that can accurately predict diseases based on a given set of symptoms.
2. **To preprocess and analyze a medical dataset**, extracting relevant features (symptoms and disease labels) to improve prediction accuracy.
3. **To design a user interface** using Python's Tkinter library that allows users to input symptoms interactively.
4. **To categorize symptoms** based on body parts (e.g., eyes, skin, tongue, hair, body changes) for easier navigation and better usability.
5. **To evaluate the model's performance** using standard metrics like accuracy, precision, recall, and F1-score to ensure reliability.
6. **To offer a preliminary diagnostic tool** that aids users in identifying probable diseases and encourages professional medical follow-up.
7. **To lay the foundation for future enhancements**, such as integration with real-time medical databases, natural language processing (NLP) for free-text symptom input, and deployment on web/mobile platforms.

By achieving these objectives, the project aims to bridge the gap between users and early healthcare intervention through the power of machine learning.

4. Scope of the Project

This project aims to demonstrate how machine learning techniques can be effectively applied to predict diseases based on symptoms provided by users. The system is designed as a preliminary diagnostic tool that enhances awareness and encourages timely medical consultation. The scope of this project is both technical and functional, focusing on core areas of machine learning, healthcare, and user interaction.

In-Scope (What the project covers):

1. Symptom-Based Disease Prediction
   The system can predict diseases by analyzing user-selected symptoms using a trained machine learning model.
2. Categorized Symptom Input
   Symptoms are organized into intuitive categories (e.g., eyes, nose, skin, tongue, hair, body changes) for better user experience.
3. Use of Random Forest Classifier
   A Random Forest model is used due to its high accuracy and robustness in classification problems.
4. Graphical User Interface (GUI)
   A simple, interactive GUI built with Tkinter allows users to select symptoms and view prediction results in real time.
5. Evaluation Metrics
   The system includes performance evaluation using accuracy, precision, recall, and F1-score.
6. Offline, Standalone Application
   The current version runs locally and does not require internet connectivity, making it suitable for areas with limited resources.

5. Technology Stack

The development of the Disease Prediction System involves a combination of tools, libraries, and technologies that work together to enable machine learning model development, data processing, and user interaction.

1. Programming Language

- **Python**
  - Chosen for its simplicity, extensive library support, and popularity in data science and machine learning.

2. Machine Learning & Data Processing

- **Scikit-learn**
  - Used for implementing the Random Forest Classifier and other ML algorithms.
  - Provides built-in functions for training, testing, model evaluation, and cross-validation.
- **Pandas**
  - Used for data manipulation and analysis.
  - Helpful in handling large datasets and converting symptom-disease data into model-friendly formats.
- **NumPy**
  - Provides support for efficient numerical computations and array operations.

3. User Interface

- **Tkinter (Python Standard Library)**
  - Used to build the desktop GUI.
  - Allows users to interact with the model by selecting symptoms through dropdown menus and viewing predictions.

4. Data Visualization (Optional/For Results Analysis)

- **Matplotlib / Seaborn**
  - Can be used to visualize confusion matrices, accuracy charts, or symptom distribution (if included in the project).

5. Environment & Tools

- **Jupyter Notebook / Google Colab**
  - For experimenting with code, training models, and performing data analysis.

6. System Architecture

- The architecture of the Disease Prediction System is designed to be modular, scalable, and user-friendly. It includes three main layers: **User Interface Layer**, **Processing Layer**, and **Machine Learning Layer**. Each layer handles a specific part of the workflow, from symptom input to disease prediction.

- **1. User Interface Layer**
- **Component**: GUI built with Tkinter.
- **Function**:
- Allows users to select symptoms from categorized dropdown menus (e.g., skin, hair, eyes, etc.).
- Displays the predicted disease and provides feedback.
- **Input**: User-selected symptoms.
- **Output**: Predicted disease name.

- **2. Processing Layer**
- **Component**: Backend logic (Python).
- **Function**:
- Maps user-selected symptoms into the correct feature vector format.
- Validates and encodes the input data.
- Sends the encoded input to the trained ML model.
- **Technologies Used**: Python, Pandas, NumPy.

- **3. Machine Learning Layer**
- **Component**: Trained Random Forest Classifier model.
- **Function**:
- Takes encoded symptom input.
- Uses pattern recognition learned from training data to predict the most likely disease.
- Returns prediction result to the Processing Layer.
-
- Assigns delivery partners based on location and availability.
- Implements real-time GPS tracking for accurate delivery estimation.

7. Security Features:

- **Session Management & Authentication:** Ensures secure user login and prevents unauthorized access.
- **Data Encryption:** Protects sensitive user information and payment details.
- **Firewall & Access Control:** Prevents unauthorized access to backend servers.

Workflow of the System:

1. Data Flow Summary
2. Symptom Input – User selects symptoms via GUI.
3. Input Encoding – Backend converts symptoms into binary feature vector.
4. Model Prediction – Random Forest Classifier predicts the disease.
5. Result Display – GUI presents the predicted disease to the user.
6. ---
7. Optional Component: Data Layer
8. Function: Stores symptom-disease mappings and training dataset.
9. Format: CSV file or DataFrame in memory.

7. Methodology

The methodology followed in this project involves a series of systematic steps to design, develop, train, and evaluate a disease prediction system using machine learning. The approach is primarily data-driven, leveraging symptom-disease associations to train a predictive model. Below is a breakdown of the complete methodology:

---

1. Data Collection

- A structured dataset containing a list of symptoms and corresponding diseases is obtained.
- Each record in the dataset represents a combination of symptoms mapped to a specific disease.
- Example source: Publicly available datasets such as those from the **UCI Machine Learning Repository** or simulated datasets based on medical references.

---

2. Data Preprocessing

- **Cleaning**: Removal of duplicate, irrelevant, or inconsistent data entries.
- **Encoding**: Symptoms are converted into binary format (1 = symptom present, 0 = symptom absent).
- **Feature Selection**: Most relevant symptoms are selected based on frequency or correlation with target labels (diseases).
- **Label Encoding**: Disease names are converted to numeric format for classification purposes.

---

3. Model Selection

- Various classification algorithms were considered, including:
  - Decision Tree
  - Naive Bayes
  - Support Vector Machine (SVM)
  - **Random Forest** *(selected for this project due to high accuracy and robustness)*

---

4. Model Training

- The Random Forest Classifier is trained on the preprocessed dataset.

- **Training/Test Split**: The data is split into training and test sets (e.g., 80/20) to validate performance.
- **Cross-validation**: Optional k-fold cross-validation is performed to ensure reliability.

---

5. Model Evaluation

- The model is evaluated using metrics such as:
  - **Accuracy**
  - **Precision**
  - **Recall**
  - **F1-Score**
- A **confusion matrix** is used to visualize model performance and misclassification.

---

6. GUI Integration

- A graphical user interface is developed using **Tkinter**.
- Users can select symptoms from dropdown lists categorized by body regions (e.g., eyes, nose, skin, etc.).
- The selected symptoms are passed to the model, and the predicted disease is displayed in real-time.

---

7. Testing & Validation

- The system is tested using sample symptom sets to ensure the prediction is logical and aligned with known symptom-disease patterns.
- Edge cases and invalid inputs are also tested to make the system more robust.

8. Features of the Project

- The Disease Prediction System is designed to be simple, efficient, and helpful to users by providing a preliminary diagnosis based on symptoms. Below are the core features of the system:

1. Symptom-Based Input

- Users can input their symptoms manually via dropdown menus.
- Symptoms are organized into categories (e.g., **eyes**, **nose**, **skin**, **hair**, **tongue**, **body changes**) for easy navigation.

2. Intelligent Disease Prediction

- Utilizes a trained **Random Forest Classifier** to predict diseases based on the combination of symptoms selected.
- Capable of handling multiple symptoms simultaneously.
- Provides fast and accurate predictions.

3. User-Friendly Graphical Interface

- Built using **Tkinter**, the GUI is clean, intuitive, and interactive.
- Includes dropdowns, buttons, and a result display area.
- No need for technical knowledge to use the system.

4. Real-Time Prediction

- The system instantly displays the most probable disease after symptoms are selected.
- Reduces waiting time compared to traditional diagnosis.

5. Lightweight and Offline Capability

- Works as a **standalone desktop application**.
- No internet connection required for making predictions.
- Suitable for use in remote or low-resource areas.

9. Implementation Details

The implementation phase brings together all components of the disease prediction system — from data preprocessing and model training to user interface integration. This section outlines how the system was built step-by-step.

---

1. Data Preparation

- A dataset consisting of diseases and their associated symptoms is loaded (in .csv or .xlsx format).
- Each row represents a disease instance, with columns for symptoms and the corresponding disease label.
- Missing or duplicate values are handled during preprocessing.

Tools Used:

- **Pandas** for reading and cleaning the data.
- **NumPy** for numerical operations.

---

2. Feature Engineering

- Symptoms are converted into a **binary feature vector**: 1 if a symptom is present, 0 otherwise.
- The target column (prognosis or disease) is **label-encoded** for compatibility with machine learning algorithms.

---

3. Model Training

- A **Random Forest Classifier** from Scikit-learn is used for training.
- Data is split into **training** and **testing** sets (e.g., 80:20).
- The model is trained on the training set and evaluated on the test set.

Key Parameters:

- n_estimators: Number of trees in the forest.
- criterion: Used to measure the quality of a split (e.g., "gini").

4. Model Evaluation

- After training, the model is evaluated using:
  - **Accuracy**
  - **Precision**
  - **Recall**
  - **F1-score**
- A **confusion matrix** is plotted to assess prediction reliability

10. Future Scope

The current system provides a basic yet functional framework for disease prediction using user-input symptoms and a machine learning model. However, there is significant potential to enhance the system in terms of features, scalability, and real-world application. Below are several future improvements and directions this project can take:

---

1. Natural Language Processing (NLP) Integration

- Allow users to **enter symptoms in plain text**, which the system can interpret using NLP techniques.
- Improves user experience by removing the need for fixed dropdowns.

---

2. Mobile and Web-Based Deployment

- Develop a **web or mobile application** version for easier and broader access.
- Use frameworks like **Flask/Django** (for web) or **React Native/Kivy** (for mobile).

---

3. Real-Time Medical Data Integration

- Integrate with **healthcare APIs or hospital databases** to fetch real-time patient data for improved prediction accuracy.
- Enable dynamic updates to the symptom-disease database.

- **4. Multi-Disease Prediction with Probability Scores**
  - Enhance the model to predict **multiple potential diseases** along with **confidence percentages**.
  - Helps users understand the likelihood of each outcome.

- **5. Inclusion of Demographic and Medical History Data**
  - Add support for age, gender, medical history, and lifestyle factors to make predictions more personalized and accurate.

- **6. Voice-Based Interaction**
  - Implement **speech recognition** for users who prefer or need to speak symptoms instead of typing/selecting them.

11. Conclusion

This project demonstrates the potential of machine learning in healthcare, particularly for disease prediction. While it does not replace professional medical advice, it can serve as a helpful preliminary tool for users seeking insights into their symptoms. With further refinement, this system can be extended for broader medical applications.With advancements in technology, this system can be further enhanced through **AI-driven recommendations, blockchain security, and smart automation** for deliveries. The

12. References

1. **Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.**
2. **Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer.**
3. **Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.**
4. **Dey, R., & Raihan, M. (2020). Disease Prediction using Machine Learning Algorithms: A Study. *International Journal of Computer Applications*, 975, 8887.**
5. **UCI Machine Learning Repository. (n.d.). *https://archive.ics.uci.edu/ml/index.php***
6. **Python Software Foundation. (n.d.). *Python Programming Language*. https://www.python.org**
7. **Tkinter Documentation – *Python GUI Programming*. https://docs.python.org/3/library/tkinter.html**
8. **Kaggle. (n.d.). Symptom-Disease Dataset. *https://www.kaggle.com* (used for sample or simulated datasets)**