Multiple Disease Prediction System Documentation

This project is a complete machine learning based health prediction system that provides quick evaluations for three important medical conditions.

Diabetes

Heart disease

Parkinsons disease

The application is built using Python and Streamlit and is designed to offer a simple interactive and accessible platform where users can input health related values and instantly receive prediction results. The project operates entirely on the browser and does not require any external tools beyond a trained model and Streamlit.

Overview of the System

The system consists of three independently trained machine learning models. Each model is responsible for predicting a single health condition and each uses different sets of medical features. These models are stored as pickle files and loaded into the application at startup.

The app uses a sidebar for navigation. Users can switch between the three disease prediction modules. Each module presents a dedicated input section that corresponds to the medical parameters required by the chosen model.

Models Loaded

diabetes_model

heartdisease_model

parkinsons_model

These models are created offline using scikit learn and saved using pickle. They are then reused inside the Streamlit app for real time inference.

Technology Stack

Python is used as the core language that handles the logic for model loading and prediction

NumPy is used indirectly for numerical array formatting

Pickle is used to load the pre trained ML models stored as sav files

Streamlit creates the front end layout and handles user interaction

Streamlit Option Menu provides the sidebar navigation for the three prediction modules

Scikit Learn models trained offline before being exported as pickle files

Application Flow

When the app starts the three models are loaded into memory through pickle load.

A sidebar menu appears allowing the user to choose between

Diabetes Prediction

Heart Disease Prediction

Parkinsons Prediction

When a user selects a module Streamlit displays the required input fields in a structured layout using multiple column blocks for readability.

After entering values the user clicks a button that triggers the prediction function for the selected disease. The app then

Collects the inputs

Converts the values to numeric form if needed

Structures them into a list

Reshapes the input into the correct array format

Passes the array to the respective model

Displays the output result in a clean success message

Diabetes Prediction Module

This module accepts eight medical features from the user.

Number of pregnancies

Glucose level

Blood pressure

Skin thickness

Insulin level

BMI value

Diabetes pedigree function

Age

The model expects the input in a two dimensional structure which is constructed as a list inside another list. The model returns zero or one where one indicates diabetic condition.

Heart Disease Prediction Module

This module contains thirteen medical attributes that represent cardiac related conditions and patient health indicators.

Age

Sex

Chest pain type

Resting blood pressure

Cholesterol

Fasting blood sugar

Resting ECG results

Maximum heart rate

Exercise induced angina

Oldpeak

Slope

Number of major vessels

Thal value

Inputs are validated and converted to numeric form. The model returns zero or one where one indicates heart disease.

Parkinsons Prediction Module

This module uses twenty two vocal and neuromuscular biomedical parameters. These include

MDVP frequency values

Jitter measures

Shimmer measures

NHR

HNR

RPDE

DFA

Spread values

D2

PPE

These parameters reflect vocal stability, harmonic structure and nonlinear signal behavior. The model returns zero for healthy individuals and one for Parkinsons disease.

Preprocessing and Internal Logic

User input values are collected through text input fields

Before prediction values are converted into float numeric types

Inputs are grouped inside a list

This list is reshaped as a two dimensional structure because all scikit learn models expect input in the format one sample by many features

Models make prediction instantly and return a binary result

The app displays a readable message to the user

Streamlit Interface Design

The interface is arranged in multiple columns for better readability

A sidebar is used for switching between prediction modules

Each prediction output is displayed with st.success which creates a prominent green highlight

If any input is invalid such as non numeric text in numeric fields an error message is displayed

Advantages of the System

Simple and extremely fast health awareness tool

Centralized interface for predicting three different diseases

Models are reusable and can be updated without changing the UI

Easy to deploy on Streamlit Cloud or integrate into larger applications

Educational resource for understanding ML model deployment

Real time prediction capability without the need for heavy infrastructure

Prediction Workflow

User selects disease from the sidebar

User enters medical values

System validates and converts inputs

Model processes the values

Prediction is shown immediately

Possible Deployments

Streamlit Cloud for public access

Flask or FastAPI for full backend integration

Hospital or clinic kiosks for awareness drives

Community screening portals

Educational and demo based environments

Future Improvements

Automatic numeric validation for better reliability

Use of sliders or dropdowns instead of plain text inputs

Addition of charts or detailed statistics

Mobile responsive interface

SHAP explanations for model interpretability

Improved visual design and usability enhancements

Conclusion

This multiple disease prediction system demonstrates how machine learning models can be combined into a unified interactive application that supports early health awareness. The project brings together trained ML models, user friendly UI design and real time inference to build a practical and extendable tool. With future enhancements the app can evolve into a comprehensive digital health assistant capable of supporting more conditions and providing deeper insights.

Author

Satyam Gajjar