Heart Disease Prediction using Machine Learning & Flask

Table of Contents

---

1. Introduction

Heart disease is one of the leading causes of death worldwide. Early detection can significantly improve treatment outcomes. This project aims to develop a **machine learning-based web application** that predicts the likelihood of heart disease based on three key patient attributes: **age, chest pain type, and maximum heart rate achieved.**

The application is built using **Flask** as the backend framework and a **Logistic Regression model** trained on a public heart disease dataset.

---

2. Project Objectives

The primary objectives of this project are:

- To develop a **machine learning model** that predicts heart disease.

- To build an **interactive Flask web application** for user input and predictions.

- To ensure a **user-friendly interface** with a visually appealing design.

- To deploy the application online for accessibility.

---

3. Dataset Description

The dataset used for training the model consists of **patient medical data**, including:

| Feature | Description |

|---------|------------|

| age | Patient's age |

| cp | Chest pain type (0-3) |

| thalach | Maximum heart rate achieved |

| target | 1: Disease Present, 0: No Disease |

The dataset was preprocessed to remove missing values and scale features appropriately before training the model.

---

4. Machine Learning Model

Model Used: Logistic Regression

Training Process:

1. **Data Preprocessing:** Cleaning, handling missing values, and feature selection.

2. **Splitting Data:** 80% training, 20% testing using `train_test_split()`.

3. **Training the Model:** Logistic Regression is trained using the `sklearn` library.

4. **Model Evaluation:** Accuracy and confusion matrix were used to evaluate performance.

Model Accuracy: ~85% on test data.

Model Storage: The trained model is saved using `joblib` as `heart_disease_model.pkl`.

---

5. Flask Web Application Development

Backend:

- Developed using **Flask** to handle HTTP requests and responses.

- `joblib` is used to load the pre-trained machine learning model.

- User inputs are processed, and predictions are returned dynamically.

Frontend:

- HTML, CSS for UI design.

- A simple form allows users to input their age, chest pain type, and heart rate.

- A **"Predict"** button processes the input and returns results.

---

6. User Interface & Features

- **Simple, Clean, and Responsive Design**

- **Colorful Background or Image-Based UI**

- **Form Input Fields:** Age, Chest Pain Type, Max Heart Rate

- **Predict Button**: Triggers prediction and displays the result

- **Instant Output:** Displays **"Heart Disease Present"** or **"No Heart Disease"**

---

7. Deployment Strategy

Local Deployment:

1. Install required libraries:

    ```bash

    pip install flask joblib pandas scikit-learn

    ```

2. Run Flask app:

    ```bash

    python app.py

    ```

Cloud Deployment (Heroku):

1. Install `gunicorn` for Heroku compatibility:

    ```bash

    pip install gunicorn

    ```

2. Create a `Procfile` for Heroku:

    ```bash
    ```

```
echo "web: gunicorn app:app" > Procfile
```

3. Push to Heroku and deploy:

```bash
git init
git add .
git commit -m "Initial deployment"
heroku create
git push heroku master
```

4. Open the app online:

```bash
heroku open
```

---

8. Challenges & Solutions

Challenges Faced:

1. **Data Imbalance:** Some classes were underrepresented in the dataset.

2. **Model Overfitting:** Optimized using cross-validation.

3. **Flask Integration Issues:** Debugging API errors and ensuring smooth frontend-backend communication.

Solutions:

- Used **balanced dataset techniques** for better model generalization.

- **Hyperparameter tuning** to improve prediction accuracy.

- Debugging and testing using **Postman** for API validation.

---

9. Future Scope

This project can be enhanced with:

- **More Features** (e.g., cholesterol levels, blood pressure, diabetes history).

- **Deep Learning Models** for improved accuracy.

- **Mobile App Integration** for wider accessibility.

- **Cloud Database** to store user history and analytics.

---

10. Conclusion

This project successfully developed an **interactive and user-friendly heart disease prediction system** using Flask and Machine Learning. It demonstrates the potential of AI in healthcare and serves as a foundation for more advanced predictive models in medical diagnostics.

📌 **Keywords:** Heart Disease Prediction, Flask Web App, Machine Learning, Logistic Regression, Healthcare AI.