Rajeev Kandpal
2021UME0223

# Introduction to IoT Project Report

## **Title**: Heart Disease Alert Suggesting AI Model on ESP32

**Dataset Information**:

The project uses the open-source Heart Disease dataset from the UCI Machine Learning Repository:
https://archive.ics.uci.edu/dataset/45/heart+disease

This dataset consists of 14 input features commonly used in heart disease prediction. The full list of features is as follows:

1. age
2. sex
3. cp (chest pain type)
4. trestbps (resting blood pressure)
5. chol (serum cholesterol)
6. fbs (fasting blood sugar)
7. restecg (resting electrocardiographic results)
8. thalach (maximum heart rate achieved)
9. exang (exercise induced angina)
10. oldpeak (ST depression)
11. slope (slope of the ST segment)
12. ca (number of major vessels)

13. thal (thalassemia)

14. target (presence of heart disease: 0 = No, 1 = Yes)

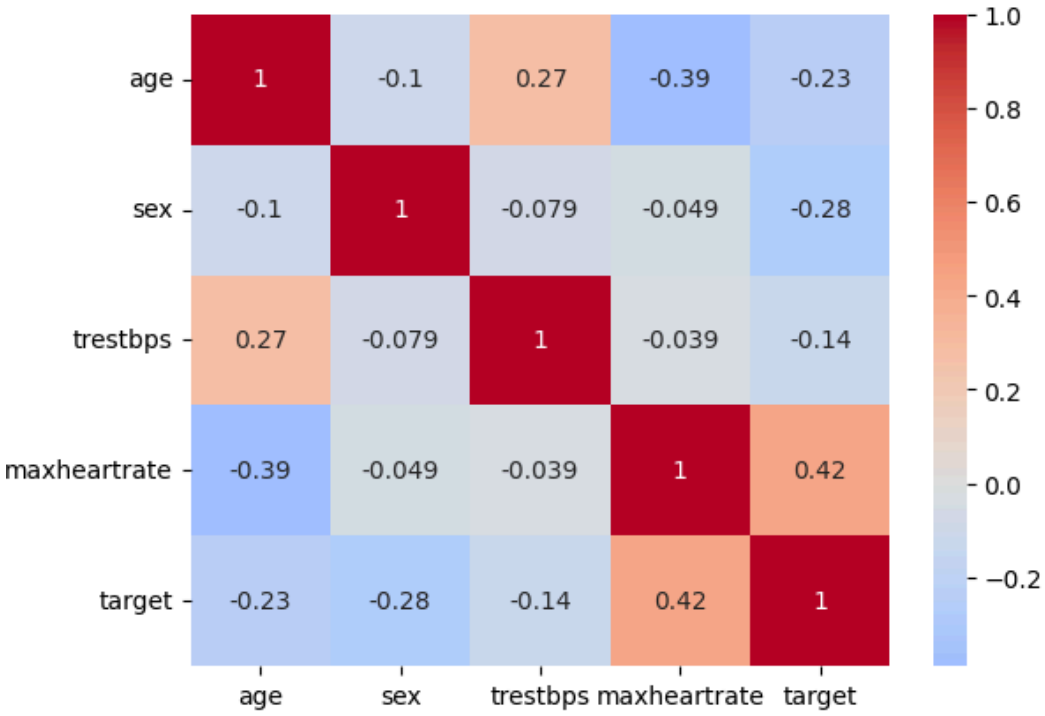
---

**Selected Features for the Project**:
Out of the 14 features, only the following 4 input features were used:

- **age**
- **sex**
- **trestbps** (resting systolic blood pressure (mmHg) )
- **thalach** (renamed as maxheartrate)

The output label used is:

- **target** (indicates heart disease presence)




Correlation heatmap between different features

---

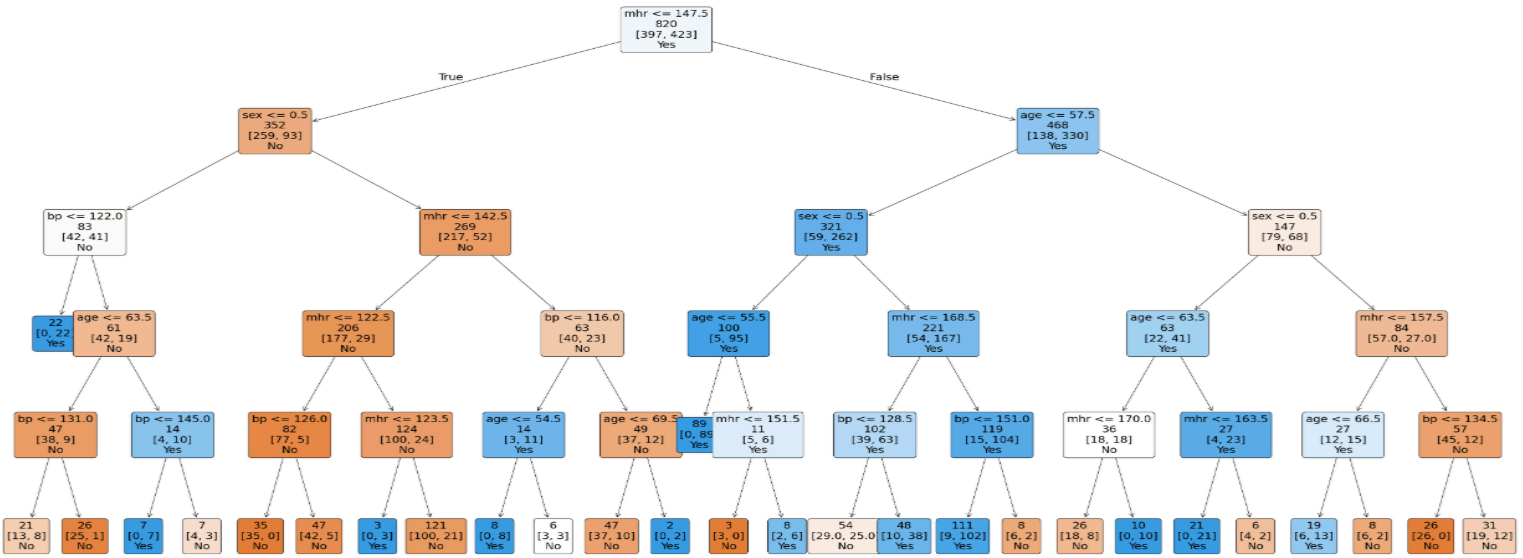
**Motivation**:
The idea for this project came from observing the growing capabilities of smartwatches (e.g., Samsung and Huawei) to monitor blood pressure and heart rate. As these features become more accessible and cheaper, wearable devices could help in early heart health monitoring. This project aims to use such health data to suggest early alerts based on a trained model.

---

**Model Development**:

- Model Type: Decision Tree
- Library Used: Scikit-learn (Python)
- Training/Test Split: 80/20
- Accuracy (Test): 79.5%
- Cross-validation Accuracy: 77%
- Max Depth: 5 (to avoid overfitting on small dataset)
- Number of data points used: 1025

Since the model uses fewer features and a simple structure, it performs reasonably well for its size and is suitable for embedded deployment.

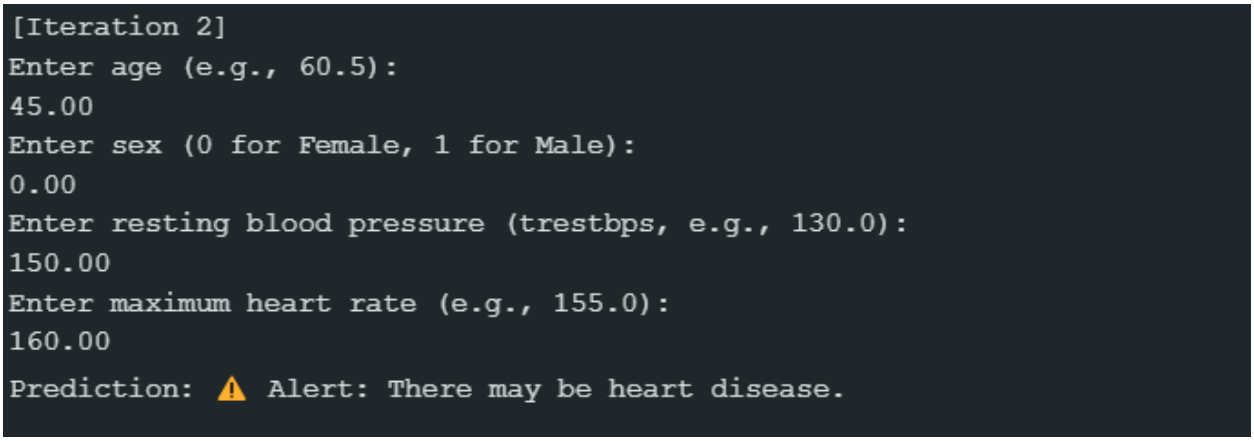



Decision tree with depth 5 and accuracy = 79.5%

---

**Deployment on ESP32**:

- The decision tree logic was manually converted to C++ and deployed on ESP32.
- Due to lack of direct sklearn-to-ESP conversion tools, the decision rules were written by hand.
- Input is provided via Serial Monitor.
- Output is displayed on the Serial Monitor after inference.
- The deployed model performs as expected during tests.




Output on ESP32

Link for the model training notebook and esp code file:
https://drive.google.com/drive/folders/1BdoNmf3pxwoX-EVMicq78_8mLIt6EonG?usp=drive_link