Final Project: Predicting Heart Failure

Dhiren Patel

May 8, 2025

Abstract

This is a supervised learning project that intends to train a model predicting the presence of heart failure using a neural network. In this work, we preprocess raw medical data and design, train, and evaluate multiple neural networks. We show that ∼88% accuracy and ∼92% recall in heart failure prediction can be achieved with these methods.

1 Introduction

Cardiovascular (heart) diseases are among the most prevalent causes of death in the world. The ability to detect and predict heart failure early can help provide patients an increased chance of survial and even an extended lifespan.

Machine Learning techniques like neural networks can help identify the complex relationships between health factors that lead to heart failure. In this project, we will train a predictive model that can understand relationships between these variables at a more complex degree than can be understood by humans.

2 Methodology

2.1 Data Collection and Description

For this project, we are utilizing the publicly available UCI Heart Failure Prediction Dataset. The dataset is a compilation of 5 independent Heart Failure datasts from Cleveland, Hungary, Switzerland, Long Beach (VA), and Stalog. There are a total of 918 unique observations with the following features:

- **Age:** in years

- **Sex:**
  - **M**: Male
  - **F**: Female
- **ChestPainType:**
  - **TA**: Typical Angina
  - **ATA**: Atypical Angina
  - **NAP**: Non-Anginal Pain
  - **ASY**: Asymptomatic
- **RestingBP:** in millimeters of mercury (mm Hg)
- **Cholesterol:** in milligrams per deciliter (mg/dl)
- **FastingBS:**
  - **1**: $>$ 120 mg/dl
  - **0**: $<$ 120 mg/dl
- **RestingECG:**
  - **Normal**
  - **ST**: ST-T wave abnormalities
  - **LVH**: Left Ventricular Hypertrophy
- **MaxHR:** in bpm
- **ExerciseAngina**
  - **Y**: chest pain during exercise
  - **N**: no chest pain during exercise
- **Oldpeak:** numerical, representing ST depression of exercise relative to rest
- **ST_Slope:** the slope of the peak exercise ST segment
  - **Up**: (normal)
  - **Flat**

    - **Down**

- **Heart Disease**, in this case, the target variable. HeartDisease is binary with 0 representing the lack of heart failure and 1 representing the presence of heart failure in a patient.

2.2 Preprocessing

Prior to training our model, it will be necessary for us to preprocess the data for ease of interpretability by our model, leading to greater efficiency and lower loss.

1. **Data Cleaning**: Impute or remove missing / incomplete data

2. **Encoding Categorical Features**: Use one-hot encoding to convert categorial data to numerical (e.g., ChestPainType, RestingECG).

3. **Scaling**: Standardize numeric features (Age, Cholesterol, Oldpeak) for improved training stability.

4. **Train-Test Split**: Split data into training / testing / validation. We will use the training data to train our model on, validation data to test our training data on during training, and testing to test our model on after training.

2.3 Neural Network Architecture

We will start with a standard neural network architecture as described below and fine-tune our model emperically, adding, subtracting, and modifying layers, activation functions, optimizers, and regularization functions.

- **Input Layer**: Send input features through a first layer with number of nodes equal to features after preprocessing.

- **Hidden Layers**: Send the output of the input layer into multiple fully connected layers with a ReLU activation function. We will emperically determine the specifics of these layers.

- **Output Layer**: The output layer will consist of one node with a sigmoid activation function that returns a probability from 0 to 1.

- **Loss Function**: We will use a binary cross-entropy loss function to inform our model training due to the existance of only 2 classes.

$$L(y, \hat{y}) = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$$

  where:

  - y_i is the true class (0 or 1)
  - $\hat{y}_i$ is the predicted probability for the value of 1

- **Optimizer**: We will use the Adam optimizer for its dynamic step size handling.

2.4 Model Training and Evaluation

- **Training Procedure**:

  1. Initialize weights randomly.
  2. Train the network over a series of epochs.
  3. Use test loss as a signal for stopping training at a specific epoch range empirically.

- **Evaluation Metrics**: We will use the following evaluation metrics to interpret the effectiveness of our model:

  - **Accuracy:** Proportion of predictions that are correct. *TP: True Positive, TN: True Negative, FP: False Positive, FN: False Negative*

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

  - **Precision:** Proportion of correct out of all positive identifications.

$$\text{Precision} = \frac{TP}{TP + FP}$$

  - **Recall:** Proportion correct out of all heart disease instances.

$$\text{Recall} = \frac{TP}{TP + FN}$$

– **F1-score:** Harmonic mean of precision and recall.

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

– **Confusion Matrix:** A matrix which helps identify the number of FPs, FNs, TPs, and TNs.

3 Results

3.1 Model Performance

We elected to train two models - one simple model (HeartNN) with 3 linear layers with ReLU activation followed by a sigmoid classifier, and one deeper model (Heart2NN) with 4 linear layers activated by the ReLU function but with dropout layers after the first and second layers. We chose to implement this "deeper" model in an attempt to increase the feature learning ability of the model while improving or maintaining generalizability.

We identified 300 epochs to be the ideal stopping point for our model training (See Figures 1 and 2). Due to the limited data provided in the UCI dataset (N = 918), we elected to validate on test data. The following is our evaluation on test data after 300 epochs:

Model	**Accuracy**	**Precision**	**Recall**	**F1-score**
HeartNN	0.880	0.870	0.922	0.895
Heart2NN	0.875	0.869	0.912	0.890

Table 1: Test performance of each model at Epoch 300

3.2 Confusion Matrix

Actual / Predicted	**No Disease**	**Disease**
No Disease	68	14
Disease	8	94

Table 2: Confusion Matrix of HeartNN

Actual / Predicted	**No Disease**	**Disease**
No Disease	68	14
Disease	9	93

Table 3: Confusion Matrix of Heart2NN

3.3 Performance Over Time

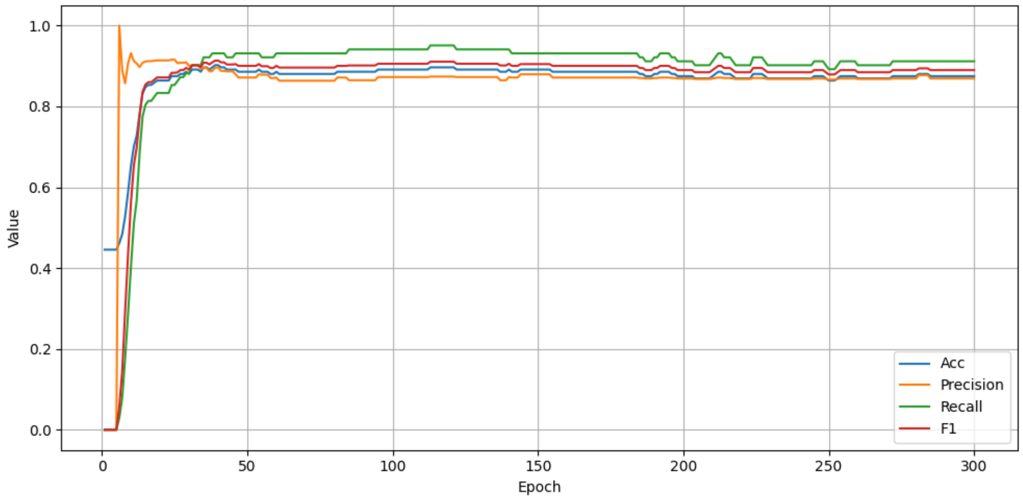

Figure 1: Accuracy, precision, recall, and F1-score of HeartNN

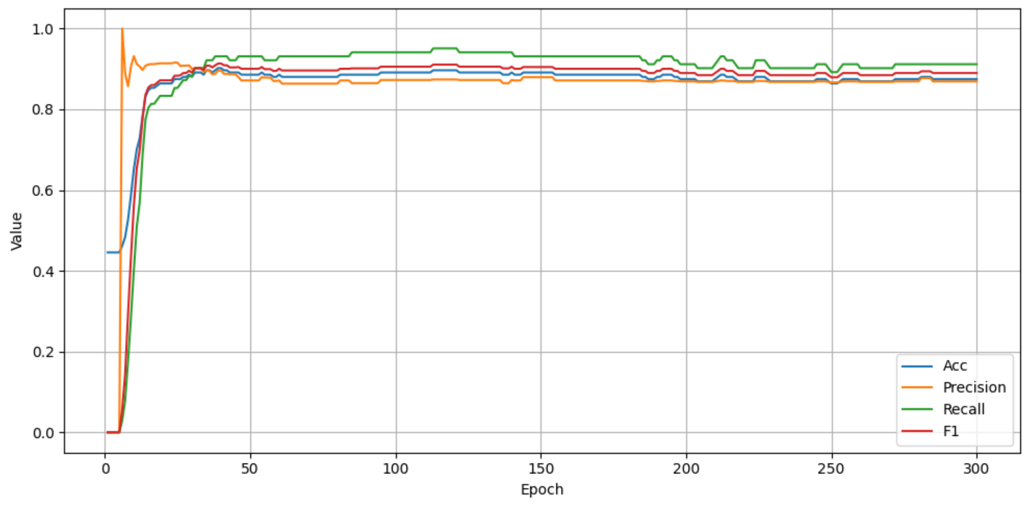

Figure 2: Accuracy, precision, recall, and F1-score of Heart2NN

4 Analysis

We identify high performance on our evaluation metrics across both HeartNN and Heart2NN. Recall rates are 92.2% for HeartNN and 91.2% for Heart2NN. In the context of identifying heart failure, recall is especially important because it provides a measure of how many positive cases of heart failure are correctly identified - an important metric when considering a life-threatening identification. Avoiding false negatives are especially important in medical applications.

Though HeartNN slightly outperformed Heart2NN in recall rates, the latter model had a higher F1 score, indicating a more stable balance between recall and precision. This is likely the result of our dropout regularization layers, which improve generalizability to the data. Nevertheless, we consider HeartNN to be a more helpful predictor of heart failure in clinical applications.

According to Figures 1 and 2, the models seem to converge at the optimal measures of accuracy, precision, recall, and F1 at ∼30 epochs. The confusion matrices of both models are identicial except for one greater TP and one lesser FN in HeartNN as compared to Heart2NN. This is consistent with higher recall rates of HeartNN.

Given the lack of high volumes of data, we suspect that the lower-depth HeartNN model is more viable in this scenario (given slightly higher recall rates), however, the

Heart2NN may outperform in deployment scenarios where a large database of data is available allowing increased depth and number of features to improve performance.

5 Conclusion

In this project, we successfully built and evaluated two neural nets with the intention of predicting heart failure based on patient health records from the UCI database. We achieved largely similar results using both a shallow and deep model, likely due to the lack of large amounts of data.

Nevertheless, we have proven that neural network architecture can achieve recall rates above 91% and F1 near 90%, suggesting that the use of ML systems in heart failure recognition give health providers a powerful tool for preliminary diagnostics.

In future work, we aim to moniter the effects of our evaluation metrics as dataset size increases past N = 918. We wonder if validating the model on external datasets can help improve reliability and performance, as this model is validated on the test data alone. Further, recent developments like SHAP and LIME can provide insight into the model's decision-making process and can be helpful for healthcare providers to interpret - increasing the likelyhood of responsible decisionmaking.

6 GitHub Repo

.

7 References

1. fedesoriano. (September 2021). Heart Failure Prediction Dataset. Retrieved [May 8, 2025] from
   `https://www.kaggle.com/fedesoriano/heart-failure-prediction`.