**Project AIM**

The project aims to develop a neural network that can predict the likelihood of a patient (user) having a heart attack in the near future based on his/her physical attributes that are collected from different medical tests or known already.

## 1. Preprocessing and EDA

### Dataset

The dataset is collected from kaggle. It encompasses information from four distinct databases: Cleveland, Hungary, Switzerland, and Long Beach V. Comprising 76 attributes, inclusive of the predicted attribute, the dataset has been predominantly utilized in published experiments focusing on a subset of 13 key features. The critical "output" variable denotes the major chance of heart attack risk in patients. Every description about the dataset is given here: Heart Attack Prediction

### Preprocessing

Feature Scaling: The features in the data are in various scales and there are differences in the variance as well, so z-score standardization is applied using StandardScaler() from sklearn.preprocessing. It computes the mean and standard deviation of a feature, then subtracts each data point in the feature with the mean, and divides the result with the standard deviation, making all of the features normally distributed with mean 0 and variance 1.

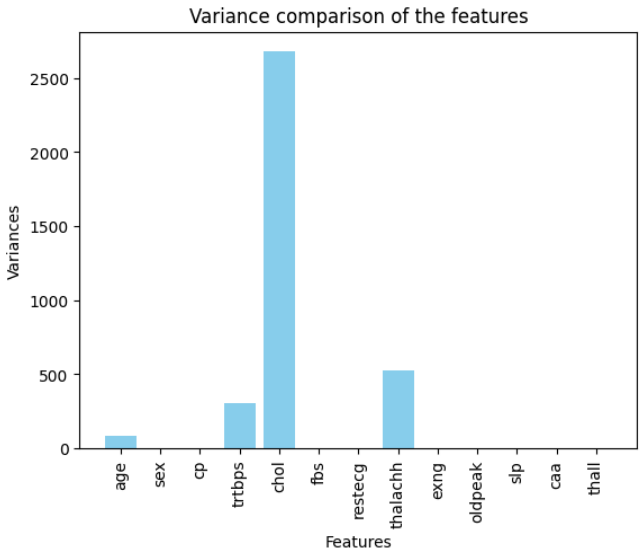


Fig 1.1: Difference of variance among the features. Bars that are not shown contain features that have variance within the range: $0.13 \leq \sigma^2 \leq 1.34$

Data Cleaning and Imputation: The dataset contains 1 duplicate row and 0 null values. Assuming that the duplicate row occurs naturally due to the similar physical characteristics of two patients, I decided not to drop it. Therefore, no cleaning or imputation process is done with the dataset.

Class Imbalance: In the dataset, there are 165 rows corresponding to the patients that had heart attack, labeled as 1. On the other hand, 138 rows corresponding to the patients that did not have a heart attack, labeled as 0.

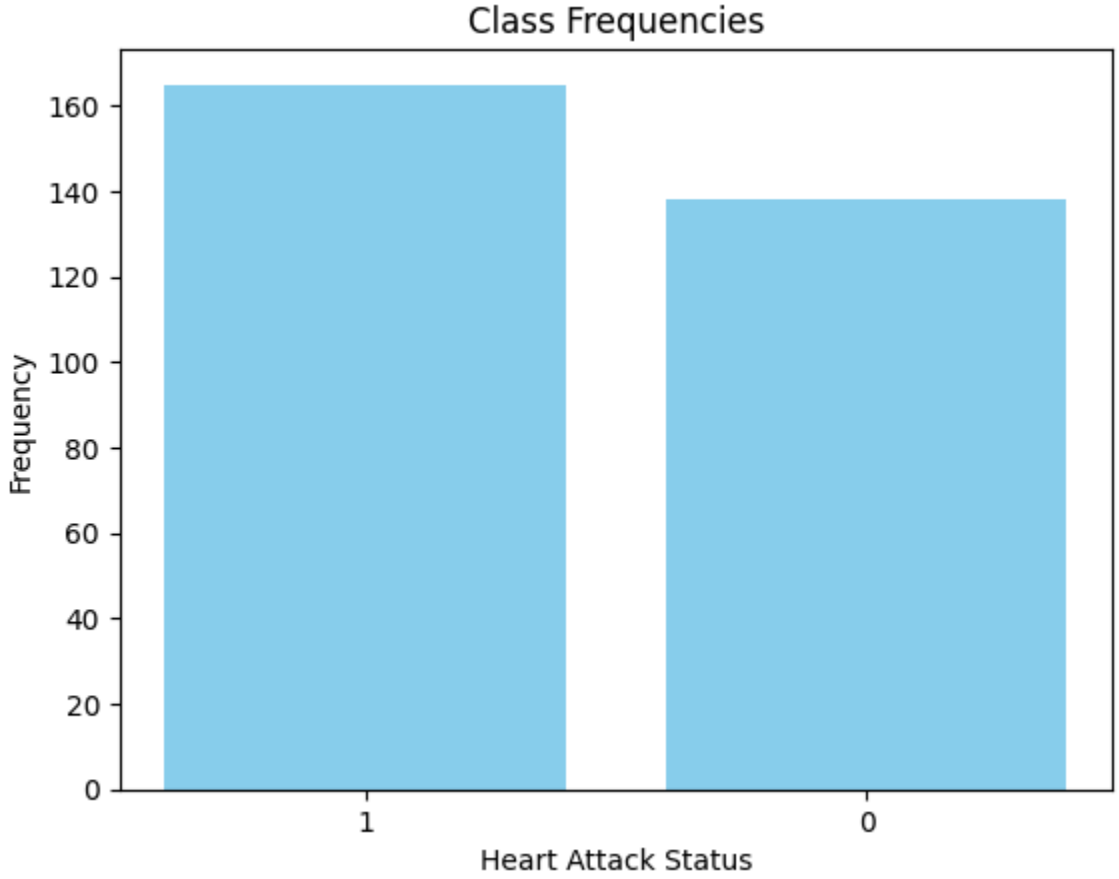


Fig 1.2: Bar chart of class frequencies

The imbalance is very little, so I decided not to apply any oversampling or undersampling techniques to balance the dataset. Neural networks or the traditional classifiers can easily handle this amount of class imbalance issues.

Dataset Splitting: The entire dataset is splitted into training set and testing set using train_test_split() function imported from sklearn.model_selection library. The ratio of training and testing set is 80:20.

## 2. Modeling

### Model Architecture

The model is a Sequential neural network designed for binary classification (predicting heart attack) with structured data containing 13 input features. It includes an input layer with 32 neurons and ReLU activation, followed by three hidden layers with 16, 4, and 2 neurons respectively, all using ReLU activations. Batch normalization is applied after the last hidden layer to stabilize training. The output layer has 1 unit with a sigmoid activation to produce probabilities for binary classification. However, the probability is later transformed into an integer number for better understanding of prediction. The model is compiled with the binary cross-entropy loss function, the Adam optimizer for adaptive learning, and accuracy as the evaluation metric, making it suitable for the given task.

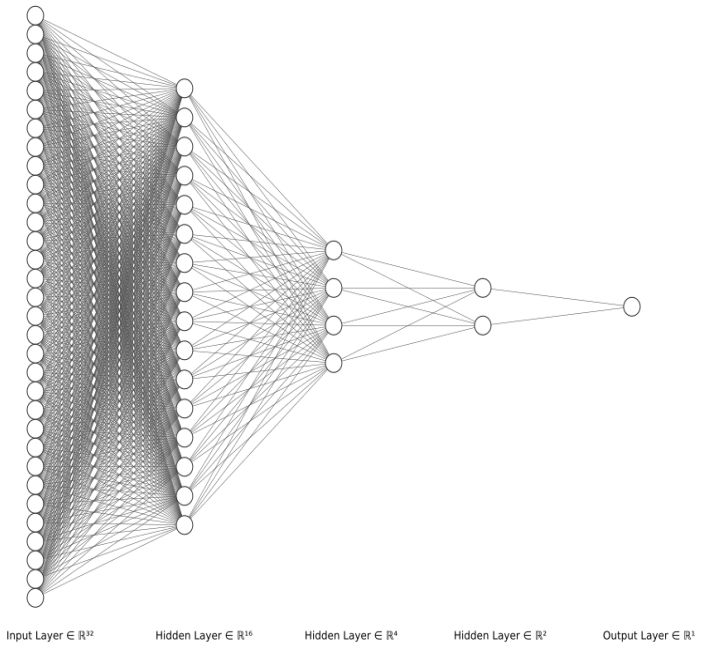


Fig 2.1: Architecture of the neural network

### Model Generalization

The model generalizes well with unseen data, as shown in Fig 1.4. There is a balance between bias and variance.

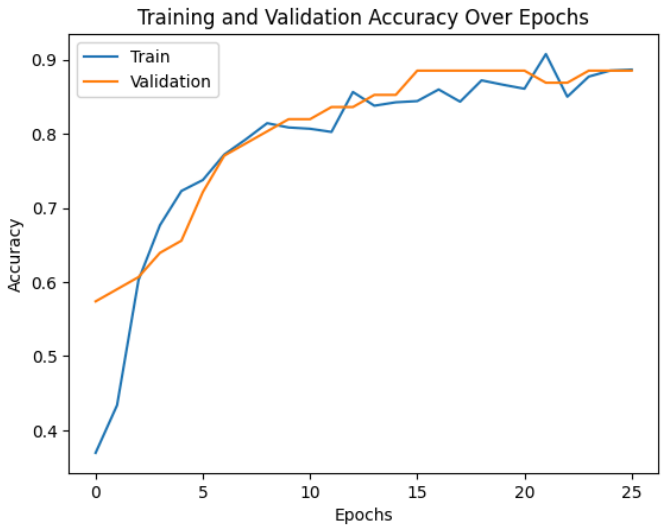


Fig 1.4: Training and validation accuracy of the neural network over epochs.

The model exhibits low variance which means that it performs similarly on both the training and validation sets. This suggests that it is not too sensitive to the specific details of the training data which could cause overfitting. A slight difference between the training and validation accuracy could indicate that the model may still have some bias (i.e., it's not perfectly fitting the data), but this is generally acceptable for achieving good generalization.

**Results**

The classification report of the model is shown below:

| Label | Precision | Recall | f1-score | Support |
|---|---|---|---|---|
| 0 | 0.95 | 0.76 | 0.84 | 25 |
| 1 | 0.85 | 0.97 | 0.91 | 36 |
| Accuracy | | | 0.89 | 61 |
| Macro avg | 0.90 | 0.87 | 0.88 | 61 |
| Weighted avg | 0.89 | 0.89 | 0.88 | 61 |

Table 2.1: Classification report of the model

*Kazi Sakib Hasan*
*Brac University, Department of CSE*