CPSC 501 Assignment 2

Report

Stage 1

Source

The data, Heart Failure Prediction Dataset, is from https://www.kaggle.com/datasets/fedesoriano/heart-failure-prediction. This dataset was used with Open Database license.

Combining/cleaning dataset

The dataset had only one table, so there was no need to combine files. However, there were some data that needed to be cleaned up. Some of the data was missing the cholesterol level (value of 0), which were removed to reduce abnormalities. Additionally, as the data relates to the health, everything else was written to be cleaned just in case (such as non-negative values for age, heart rate, etc., and possible abnormalities like heart rate that is too high).

Description of plot1.jpg

The first visualization is a histogram that counts the number of individuals in the age group. It shows that the data has high numbers for individuals between age of 50 and 60. As the data was collected from hospitals and medical centres, this may show that the age group has the highest risk of developing a heart failure.

Description of plot2.jpg

The second visualization is a line graph that shows average maximum heart rate of each age group. It gradually goes down as the age group gets higher, so it shows that individuals with a higher age tend to display lower heart rate in generally.

Description of plot3.jpg

The third visualization is a pie chart. It gathers all individuals with heart failure, and divides into individuals that have angina and individuals that don’t have angina. The pie chart displays that the number of individuals with angina and heart failure is much higher than the number of individuals without angina and with heart failure, which implies a correlation between the two.

Stage 2

Choice of columns

For making this model, the columns Age, RestingBP, Cholesterol, FastingBS, MaxHR, and Oldpeak were chosen as inputs as they were the numerical values that relates to the output. The dataset has HeartDisease clearly indicated as the only output variable, which is why this column was chosen to be the output variable.

Splitting data

The cleaned data has a total of 746 rows of data. I wanted to have as much training data as possible, but I did not want to have too little test data. The train_test_split method from scikit-learn was used to split data, and it was split 80-20 for training-test data. I thought having around 150 rows of data for testing was sufficient.

Model design

The training model has input shape of 6x1 as it is 6 columns of 1 dimensional input. The hidden layer from part 1 has a slightly lower number of 400 neurons, and the output layer is changed to 1 neuron as it is a binary output.

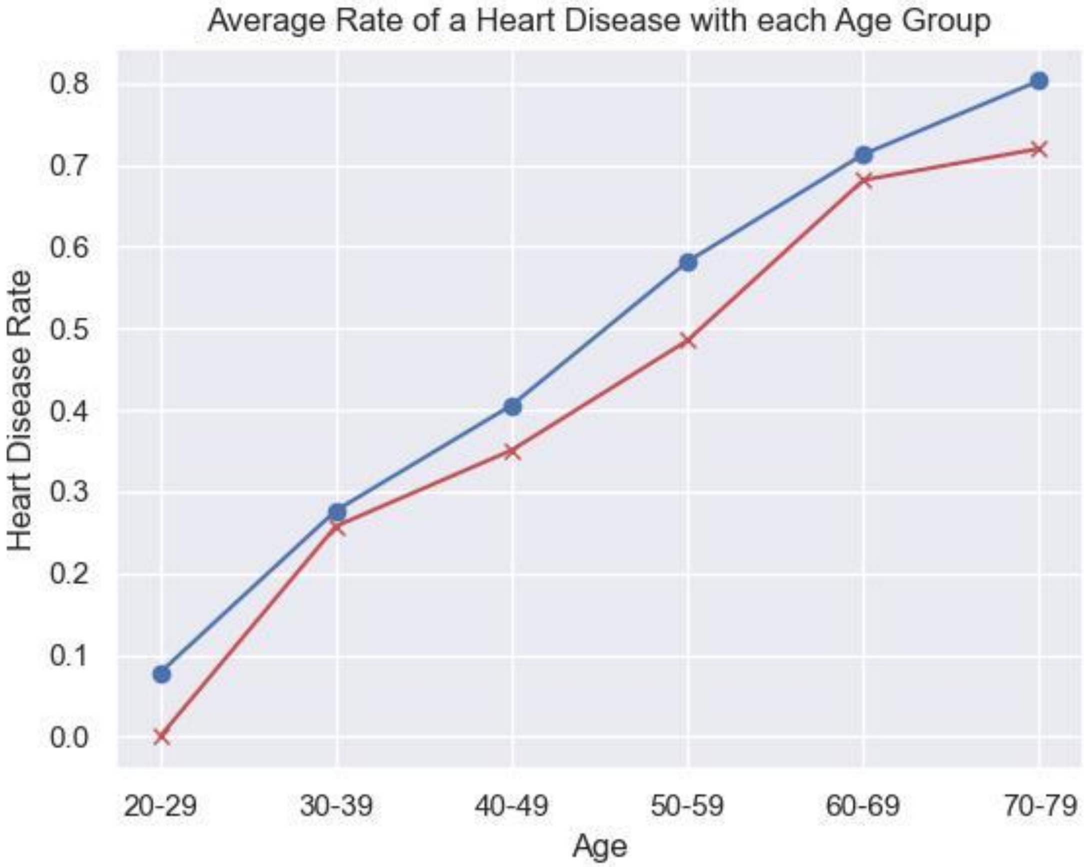
For hyper-parameters, the activation of the output layer is changed to use sigmoid and the loss function is changed to binary cross-entropy, as the dataset has a binary output variable. Epochs stayed at 10 as it performed the best, and a different value resulted in either a severe overfitting or an underfitting problem.

Accuracy

The accuracy of the model on the training data fluctuates between 56-78%, and the accuracy on the test data is between 57-78%. I feel like as the dataset itself contains 5 columns as text/category input variables, I believe relying solely on numerical data is causing the model to be inconsistent.

Stage 3

Accuracy



The above graph shows the relationship between age and the prediction rate of a heart disease occurrence. The blue line is the prediction data from the trained model, and the red line is the data from the dataset itself. The two lines show differences on its magnitudes, with the model having a higher prediction rate of a heart disease, but both follow the same trend of being older resulting in a higher chance of a heart disease.

Changes to the model

Some of the layers from part 2 was implemented. Convolution layer was not implemented as it was difficult to implement without errors. Convolution layers were omitted; only the dropout layer after the single hidden layer was implemented, and its dropout rate and epochs values were adjusted to train the model better with a much lower number of data compared to the notMNIST dataset. Dropout rate was adjusted to 0.2, and epochs were adjusted to 100.

The changes resulted in a slightly better maximum accuracy, but there was a huge improvement in stability, as there was almost no fluctuation and Train Accuracy ranged at 75-78% while Test Accuracy ranged at 76-79%. I think dropout combined with increasing epochs gave the model more data to train, resulting in a more consistent output.

Stage 4

Integrating non-numeric data

Code from part 3 was copied; groupby.ngroup() method was used for every text column in order to encode them to integer values.

Overfitting/underfitting

With the existing model from Stage 3, with 100 epochs, a hidden layer of 400 neurons and a dropout layer with a 0.2 drop rate, there was an overall underfit in the model; a training accuracy of 83-86% and a testing accuracy of 83-88% were present, with an average underfit of around 2%.

In order to address the underfit, I tried adding more hidden layers, adjusting neurons numbers, lowering drop rates, and increasing epoch, without trying to hurt accuracy. The result was that the drop rate was decreased to 0.1 and epoch was increased to 200, with a training accuracy of 82-88% and a training accuracy of 84-88%, with an average of around 0.05% underfit over 20 runs.

Other choices

There were no other choices made here; the change I made above was sufficient, and adding other factors either resulted in more overfit/underfit or a decrease in accuracy.