Summary and Analysis

Data Collection

We download the dataset and convert into a pandas dataframe. Firstly, plot a countplot of the samples in each target label as shown in Figure 1.

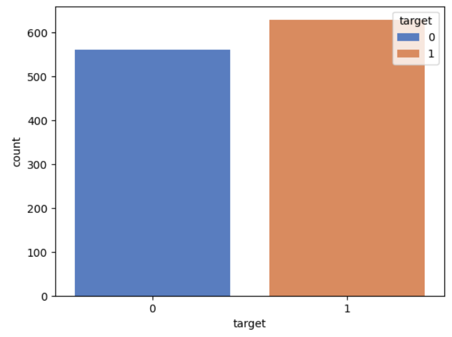

Figure 1: Countplot of different target labels

From figure 1, we see that there is approximately equal samples in each label, giving us the assurance that we have adequate samples to train a model for successful binary classification.

Next, we plot the distribution of categorical features for each label as shown in Figure 2.

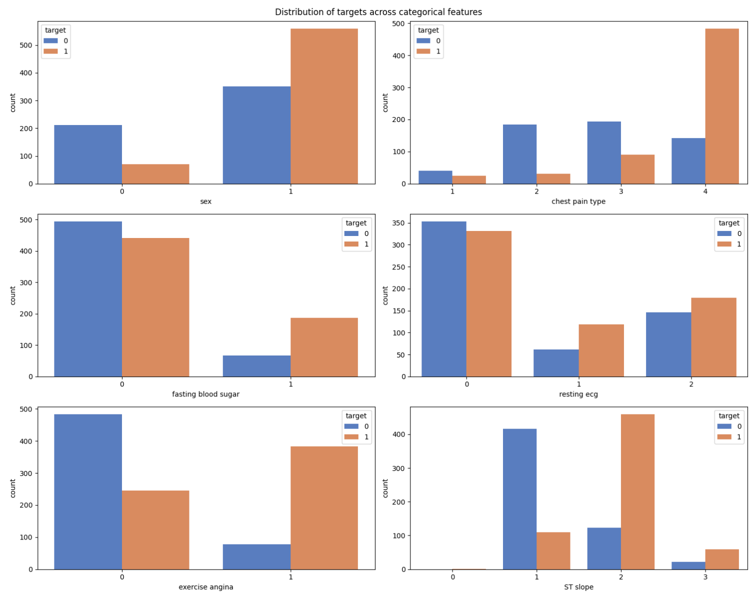

Figure 2: Countplot of categorical features for each label

Some key observations

- There are more male samples and a higher proportion of males have the disease
- Majority of patients with chest pain type 4 have the disease
- It is more probable for patients with fasting blood sugar to have the disease
- Majority of patients with exercise angina have the disease
- Majority of patients with ST slope 2 and 3 have the disease

As such we see that all categorical features have some correlation with the response. Next, we plot a heatmap of all the continuous features in Figure 3.

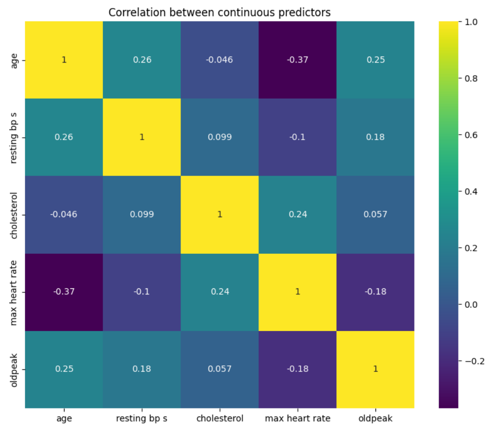

Figure 3: Heatmap of continuous features

From Figure 3, we see that no two of the features have a strong correlation. Hence we do not need to drop any repeated columns. Lastly, we plot the correlation between these features and the response.

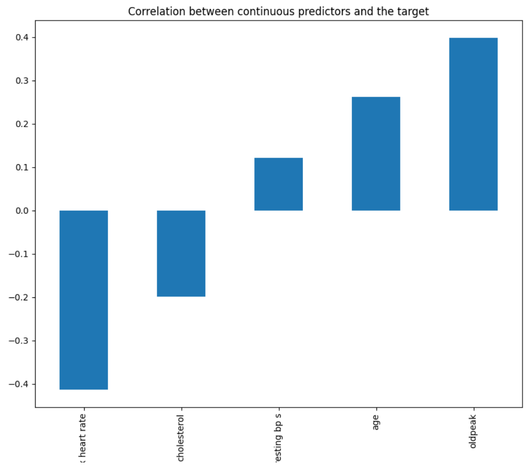

Figure 4: Barplot of correlation between features and target

Some key observations

- Maximum heart rate is the most negatively correlated to the presence of the disease
- Old peak is the most positively correlated to the presence of the disease

Data Preprocessing

We first need to convert all the categorical columns to one hot encoding so that it is compatible with a neural network. Next, we want to scale all our variables to be in the range $[0, 1]$. Lastly, we will split our data into training and testing samples with a split of 80 to 20 using a seed of 81. We will only use the testing samples for our model evaluation.

Hyperparameter tuning

There are many hyperparameters that we can tune in a neural network and it would be unrealistic to conduct at exhaustive search over them due to computational restraints. Hence we choose to fix some hyperparameters while varying a selected few. In this experiment, these are the hyperparameters fixed

- 3 hidden layers with a relu activation
- a dropout later after each hidden layer
- a final layer with 1 neuron and a sigmoid activation
- binary crossentropy loss
- adam optimiser with default parameters

We vary the neurons in the 3 layers, the dropout rate, the batch size and the number of epochs. We consider these values

- 1st layer neuron: 40, 80, 160
- 2nd layer neurons: 20, 40, 80
- 3rd layer neurons 10, 20, 40
- dropout rate 0.1, 0.2
- batch size 32, 64
- number of epochs 20, 30

We conduct a grid search over these parameter values and use k-fold cross validation with 3 folds to select the best set of parameters. Again, a seed is set for reproducibility. On a CPU, the search took about 56 minutes and the best set of hyperparameters we find are 80, 80 and 20 neurons in the first 3 layers respectively, 0.1 dropout rate, a batch size of 32 trained using 30 epochs.

Model training

Using this set of hyperparameters, we train our model and we find that it achieves a training accuracy of 89%. We then save our model to a json file and our trained model weights to a H5 file

Model Evaluation

We now use our trained model to make predictions on our test set. The output is a prediction probability and hence to assign label classes, we use a threshold of 0.5, thus any output with a prediction probability of greater than 0.5 is assigned class 1 and class 0 otherwise. The results are shown in the table in Figure 5.

	Precision	Recall	f1-score	Support
0	0.77	0.87	0.81	101
1	0.89	0.80	0.85	137
accuracy			0.83	238
macro avg	0.83	0.84	0.83	238
weighted avg	0.84	0.83	0.83	238

Figure 5: Evaluation Metrics of trained neural network

We observe that our model achieves a decent accuracy of 83% in classifying the samples to their correct labels. However, in the context of heart diseases, we might want to identify all the patients with a heart disease as there might be graver consequences in identifying a patient to have no heart disease when they have it. Hence we might want not be satisfied with the label 1 recall score of 80%. To improve this, we can either lower the threshold of 0.5 so that more points will be classified as having label 1 or we can use a custom loss function that penalises more heavily for misclassifying a patient with heart disease.