Previsão de doença cardíaca

Aplicações de Inteligência Artificial

–

Universidade de Aveiro

2022/2023

João Martins (NMec 93304)
Mestrado Integrado em Engenharia Computacional
Departamento de Física
Aveiro, Portugal
joao.paul@ua.pt

Miguel Marques (NMec 98532)
Licenciatura em Engenharia Computacional
Departamento de Física
Aveiro, Portugal
miguel.rosas@ua.pt

***Resumo*—A doença cardíaca é a maior causa de mortes para pessoas dos Estados Unidos, a cada 34 segundos uma pessoa morre nos Estados Unidos, no resto do mundo também é uma das principais causas de morte.**

O nosso trabalho tem como objetivo prever se uma determinada pessoa sofre tem uma doença cardíaca.

O modelo que teve melhor desempenho foi *Neural Networks* em que obtivemos um valor de accuracy de 88.7640% para o conjunto de dados de treino, 83.3333% para o conjunto de dados de validação e 88.1356% para o conjunto de dados de teste

I. Introdução

O desenvolvimento de novas tecnologias assim como métodos de inteligência artificial têm tido um crescimento acentuado.

Este projeto foi realizado no âmbito da Unidade Curricular "Aplicações de Inteligência Artificial".

Tendo em conta o nosso objetivo de prever se uma determinada pessoa tem doença cardíaca, utilizamos Neural Networks, para isto foi necessário uma base de dados [1], utilizamos o conjunto de dados pertencente a Cleveland, processado. Este conjunto de dados continha os dados de várias pessoas relativamente a algumas features.

II. Análise do conjunto de dados

O conjunto de dados contém 14 features, dos quais um representa se tem doença de coração e a sua intensidade.

A. Descrição de features

- **Age:** idade da pessoa em anos
- **Sex:** sexo da pessoa (1 - homem; 0 - mulher)
- **cp:** tipo de dor no peito (1 - angina típica, 2 - angina atípica, 3 - não anginosa, 4 - assintomática)
- **trestbps:** pressão arterial em repouso (mm Hg)
- **chol:** soro de colesterol (mg/dl)
- **fbs:** açucar no sangue em jejum ¿ 120 mg/dl (1 - verdade, 0 - falso)
- **restecg:** resultados eletrocardiográficos em repouso (0 - normal, 1 tem anormalidade da onda ST-T, 2 - provavelmente tem ou definitivamente tem hipertrofia ventricular esquerda pelo critério de Estes)
- **thalach:** frequência cardíaca máxima atingida
- **exang:** angina induzida por exercício
- **oldpeak:** depressão ST induzida pelo exercício em relação ao repouso
- **slope:** a inclinação do segmento ST no pico do exercício (1 - ascendente, 2 - plano, 3 - descendente)
- **ca:** número de vasos principais (0-3) coloridos por fluorosopia
- **thal:** 3 - normal; 6 - defeito corrigido; 7 - defeito reversível
- **num:** diagnóse de doença cardíaca (0 - não tem doença cardíaca, (1-4) tem doença cardíaca e a sua intesidade)

III. Pré-processamento dos dados

A. Tratamento do data set

Removemos pessoas que tinham valores em falta, e para além disso convertemos a intensidade da doença cardíaca em um so valor que significava que tinha doença cardíaca (os valores 1,2,3,4 ficaram todos 1 que significa que sofre de doença cardíaca enquanto que o 0 representa uma pessoa que não sofre da doença). Algumas das nossas features tinham valores numéricos como strings, pelo que tivemos que as converter para valores numéricos.

B. Análise dos dados

Após o pré-processamento em que desprezamos a intensidade da doença cardíaca e resumimos a intensidade a apenas tem doença. Obtivemos o seguinte gráfico:

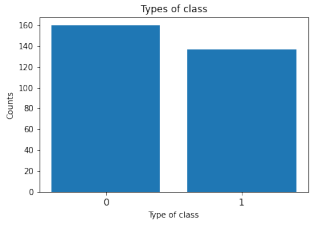

Figura 1. Quantidade de entradas em cada classe.

Como podemos observar ambas as classes tem aproximadamente o mesmo número de entradas pelo que podemos dizer que o nosso data set está balanceado.

Agora passando para a análise da distribuição das features:

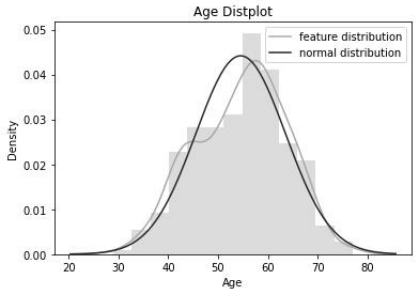

Figura 2. Histograma dos valores da feature Age, com um fit de uma função normal

É possível observar que a distribuição dos dados desta feature está próxima da de uma distribuição normal.

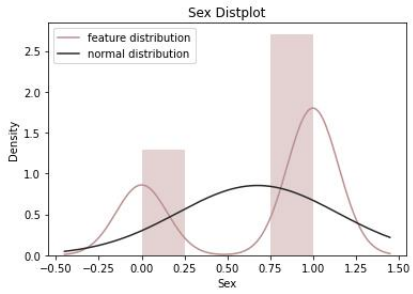

Figura 3. Histograma dos valores da feature Sex, com um fit de uma função normal

Analisando o gráfico verificamos que maior parte dos dados pertencem a homens. O conjunto de dados não segue uma distribuição normal, no entanto só há 2 valores possiveis para esta feature.

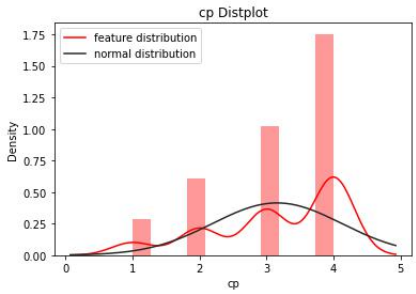

Figura 4. Histograma dos valores da feature cp, com um fit de uma função normal

Observando esta figura concluimos que maior parte das pessoas tem uma dor assintomática seguido por uma dor não anginosa, dor angina átipica e por fim a apenas uma pequena quantidade de pessoas sofre de uma dor angina típica. Mais uma vez esta feature não segue uma distribuição normal uma vez que só tem 4 valores possiveis.

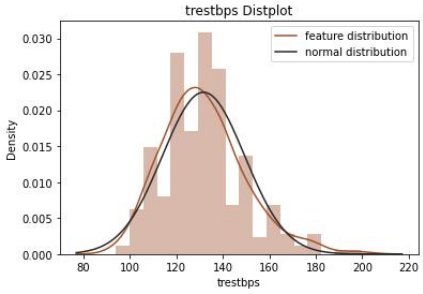

Figura 5. Histograma dos valores da feature trestbps, com um fit de uma função normal

Observamos que a distribuição dos dados desta feature está próxima da de uma distribuição normal.

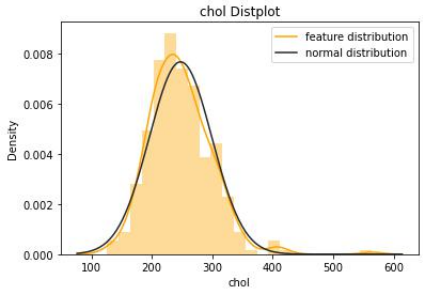

Figura 6. Histograma dos valores da feature chol, com um fit de uma função normal

A distribuição dos dados desta feature está próxima da de uma distribuição normal.

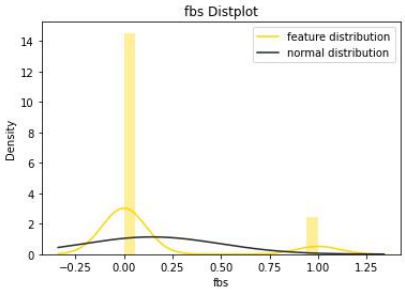

Figura 7. Histograma dos valores da feature fbs, com um fit de uma função normal

Podemos verificar que maior parte das pessoas tinha menos de 120 mg/dl de açucar no sangue em jejum.

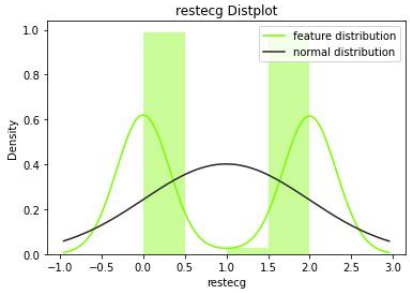

Figura 8. Histograma dos valores da feature restecg, com um fit de uma função normal

Analisando a figura verificamos aproximadamente metade das pessoas obteve resultados normal e outra metade provavelmente tem ou tem hipertrofia ventricular esquerda pelo critério de Estes, uma pequena quantidade de pessoas teve uma anormalidade nas ondas ST-T.

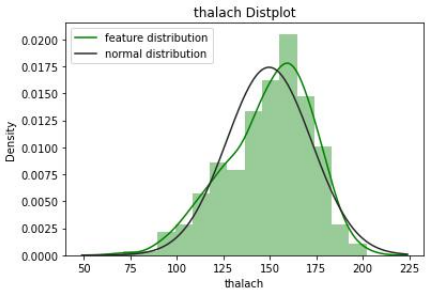

Figura 9. Histograma dos valores da feature thalach, com um fit de uma função normal

A distribuição dos dados desta feature está próxima da de uma distribuição normal.

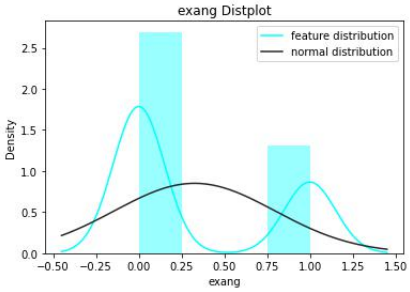

Figura 10. Histograma dos valores da feature exang, com um fit de uma função normal

Podemos verificar existe metade das pessoas tem angina induzida pelo exercicio comparativamente às que não tem.

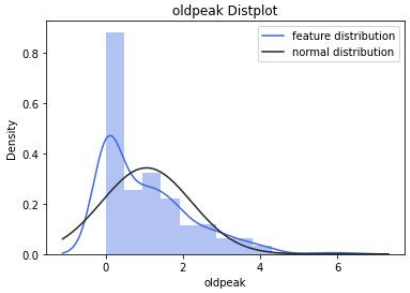

Figura 11. Histograma dos valores da feature oldpeak, com um fit de uma função normal

Analisando o gráfico verificamos que os dados não seguem uma distribuição normal.

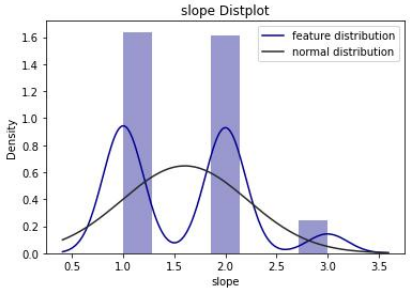

Figura 12. Histograma dos valores da feature slope, com um fit de uma função normal

Observando o gráfico verificamos que aproximadamente metade das pessoas obtiveram uma inclinação positiva, outra metade plana e uma pequena quantidade descendente, no segmento ST no pico do exercicío.

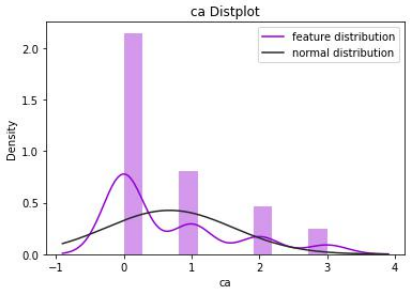

Figura 13. Histograma dos valores da feature ca, com um fit de uma função normal

Analisando o gráfico podemos concluir que quanto mais vasos estão coloridos por por fluorosopia, menor é a quantidade de pessoas que os tem.

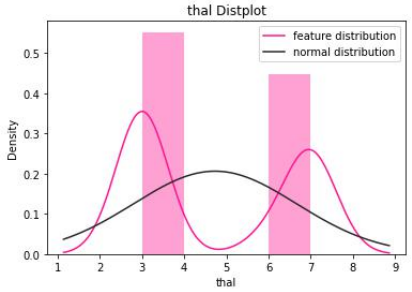

Figura 14. Histograma dos valores da feature thal, com um fit de uma função normal

Observando esta figura verificamos que dos 3 valores possiveis para thal, so estão presentes 2 no gráfico, aproximadamente metade das pessoas estão normais, enquanto que o resto tem um defeito corrigido.

Tendo em conta as ánalises anteriores podemos verificar que a escala dos valores das features diferem bastante entre elas. Excluindo as features em que tinham apenas um número de valores possíveis reduzido, apenas uma das features não segue uma distribuição normal. Por estes motivos decidimos fazer a *standardization* dos dados.

IV. Modelos utilizados

A. Neural Networks

1) Descrição do modelo: Para atingirmos o nosso objetivo utilizamos Neural Networks.

Neural Networks são compostas por node layers, uma de input e uma ou mais hidden layers e no final uma layer de output. Cada layer é conectada a outra e tem associado a ela uma peso e em treshold. Se o output de uma node individual é superior ao valor especificado do treshold, esta node é ativada e passa daods para a próxima layer caso contrário não transmitido dados para a próxima layer.

Para a implementação da nossa Neural Network utilizamos a biblioteca Keras.

2) Separação dos dados em dados de treino, validação e teste: Separamos o nosso conjunto de dados original em 3 conjuntos de dados, um de treino, um para a validação e outro para o teste, como o nosso conjunto de dados tem apenas 297 pessoas, o conjunto de treino ficou com 60%, o conjunto de validação tem 40% e o conjunto de teste ficou com os restantes 20%. O conjunto de treino irá ser utilizado para treinar o modelo, o conjunto de validação será utilizado para definir os melhores hiperparâmetros do modelo enquanto que o conjunto de teste será utilizado para testar o modelo.

3) Definição das layers: O modelo tem as layers organizadas de forma sequencial, ou seja, o output de uma layers será o input da próxima para além disso os tipos de layers que utilizamos foi *Dense* em que as layers estão totalmente conectadas. O modelo é composto por 4 layers, as 3 primeiras tem uma função *ReLU* e a última layer do modelo tem uma função *Softmax*.

A função *ReLu* retorna o valor se este for maior que 0, caso contrário retorna 0.

A função *Softmax* converte um vetor com K números reais em uma distribuição de probabilidade proporcional à exponencial dos números de input. Cada componente do vetor final estará no intervalo (0,1). A soma das probabilidades tem que ser 1, e cada probabilidade representa a probabilidade representa uma classe.

Como o modelo é sequencial o output de uma layers será o input da próxima layer. A primeira layer (*ReLU*) tem como input uma dimensão 13 que corresponde ao nosso número de *features*, as 2 layers seguintes (*ReLU*) tem como output uma dimensão de 7 e 4 respetivamente. Por fim a última layer *Softmax* tem como output uma dimensão 2 uma vez que temos apenas 2 classes.

4) Compilação do modelo: Na cosntrução do nosso modelo utilizamos o optimizer *Adam*, a loss function *binary crossentropy* uma vez que temos apenas 2 tipos de classes 0 e 1.

O optimizer utilizado *Adam* consiste na combinação de 2 metodologias de *gradient descent*, *Adaptive Gradient Algorithm (AdaGrad)* e *Root Mean Square Propagation (RMSProp)*.
O *AdaGrad* mantém um learning rate por feature, que melhora o desempenho em problemas com gradientes dispersos.
O *RMSProp* também mantém um learning rate por feature, que são adaptados baseado na média das recentes magnitudes do gradiente para o peso.

A loss function *binary crossentropy* é utilizada para classificar os dados prevendo a probabilidade de pertencer a cada uma das classes, neste caso especifico a probabilidade uma determinada pessoa ter ou não doença cardíaca.

5) Treino do modelo: Definimos uma função callback que será utilizada no modelo de treino em conjunto com os dados de validação para parar o treino do modelo assim que o valor loss do conjunto de dados de validação começar a aumentar. Desta forma prevenimos o overfitting e ficamos com perto dos melhores parâmetros do modelo.

Para encontrar os melhores hiperparâmetros do modelo escolhi um valor alto de 1000 *epochs* uma vez que o treino irá parar assim que o valor loss para o conjunto de dados de validação começa a aumentar. Como temos poucas features e apenas os dados de 297 pessoas escolhi um *batch size* de 180 que irá para englobar todos os dados de treino.

6) Resultados do modelo: Obtivemos os seguintes resultados para Neural Networks:

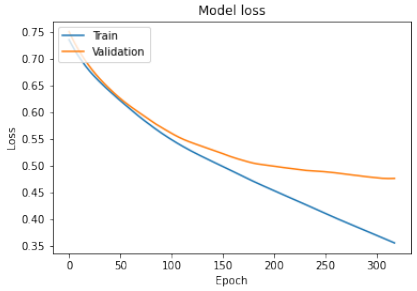

Figura 15. Gráfico da loss para o conjunto de treino e validação ao longo dos epochs

Como é possível observar no gráfico, o treino parou próximo dos 700 epochs, que foi quando o valor de loss do conjunto de dados de validação começou a aumentar.

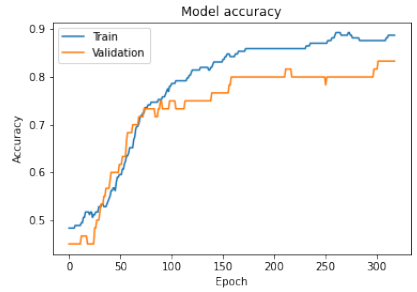

Figura 16. Gráfico da accuracy para o conjunto de treino e validação ao longo dos epochs

Podemos observar que a accuracy sobe ao longo dos epochs tanto para o conjunto de dados de treino como para o conjunto de dados de validação.

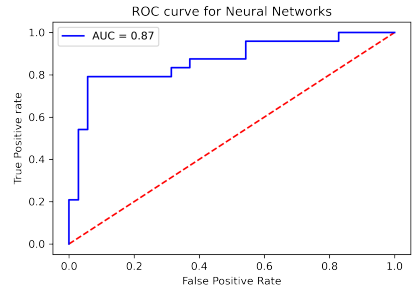

Figura 17. Gráfico da ROC Curve

Podemos observar que a AUC obtido para a ROC curve foi de 0.87, que é um resultado bom.

	0	1
0	33	2
1	5	19

Figura 18. Matriz confusão do modelo.

Analisando a matriz confusão verificamos que obtivemos 33 casos *True positive*, 2 casos *False Negative*, 5 casos *False Positive* e 19 casos *True Negative*.

Os valores obtidos para a precisão foram 88.7640% para o conjunto de dados de treino, 83.3333% para o conjunto de dados de validação e 88.1356% para o conjunto de dados de teste.

Os valores obtidos para loss foram 0.359812 para o conjunto de dados de treino, 0.476869 para o conjunto de dados de validação e 0.454766 para o conjunto de dados de teste.

Obtivemos um valor de 88.32% para a precisão, 88.14% para o recall e 87.98% para o f1-score.

B. Logistic Regression

1) Descrição do modelo: Neste modelo de classificação, baseado em aprendizagem supervisionada é calculada a soma pesada das *features* multiplicadas por pesos. O seu resultado é sempre entre zero e um , com recurso à função sigmoid consideramos que para valores de h(x) superior a 0.5 há ocurrência.O objetivo é atualizar o pesos de modo a reduzir a função de custo , ou seja, as que o número de previsões corretas seja máximo.

2) Separação dos dados em dados de treino e teste: Neste modelo separamos o nosso dataset original em 2 datasets, um deles será utilizado para treino e outro para teste. O dataset de treino terá 70% dos dados enquanto que o de teste terá os restantes 30%.

3) Criação do modelo: Criamos uma série de valores para o hiperparâmetro e usamos esta série para avaliar o modelo usando *Cross-Validation* para encontrar o melhor hiperparâmetro. Posteriormente a isto utilizamos o melhor hiperparâmetro obtido para criar o modelo.

4) Resultados do modelo: Obtivemos os seguintes resultados para *Logistic Regression*.

	0	1
0	46	8
1	12	24

Figura 19. Matriz confusão do modelo.

Analisando a matriz confusão verificamos que obtivemos 46 casos *True positive*, 8 casos *False Negative*, 12 casos *False Positive* e 24 casos *True Negative*.

Para este modelo obtivemos uma precisão de 77.5862%, um recall de 77.7778%, um valor de f1-score 77.5210% e uma accuracy de 77.9%.

C. Random Forest

1) Descrição do modelo: Para este modelo de aprendizagem supervisionada são usadas diversas árvores de decisão,daí a terminologia Random Forest. Para cada exemplo o valor previsão será o valor mais votado, ou seja o valor com mais ocurrência da lista de valores obtidos por todas as árvores de decisão.

2) Separação dos dados em dados de treino e teste: Neste modelo separamos o nosso dataset original em 2 datasets, um deles será utilizado para treino e outro para teste. O dataset de treino terá 70% dos dados enquanto que o de teste terá os restantes 30%.

3) Criação do modelo: Criamos uma série de valores para o hiperparâmetro e usamos esta série para avaliar o modelo usando *Cross-Validation* para encontrar o melhor hiperparâmetro. Posteriormente a isto utilizamos o melhor hiperparâmetro obtido para criar o modelo.

4) Resultados do modelo: Obtimemos os seguintes resultados para *Random Forest Classifier*.

	0	1
0	49	5
1	12	24

Figura 20. Matriz confusão do modelo.

Analisando a matriz confusão verificamos que obtivemos 49 casos *True positive*, 5 casos *False Negative*, 12 casos *False Positive* e 24 casos *True Negative*.

Para este modelo obtivemos uma precisão de 81.3002%, um recall de 81.1111%, um valor de f1-score 80.6689% e uma accuracy de 81.1%.

D. Decision Tree Classifier

Modelo de aprendizagem supervisionada que consiste na devisão recursiva do conjunto de dados de treino até que a maioria dos elementos de uma dada partição sejam da mesma classe.

1) Separação dos dados em dados de treino e teste: Neste modelo separamos o nosso dataset original em 2 datasets, um deles será utilizado para treino e outro para teste. O dataset de treino terá 70% dos dados enquanto que o de teste terá os restantes 30%.

2) Criação do modelo: Criamos uma série de valores para o hiperparâmetro e usamos esta série para avaliar o modelo usando *Cross-Validation* para encontrar o melhor hiperparâmetro. Posteriormente a isto utilizamos o melhor hiperparâmetro obtido para criar o modelo.

3) Resultados do modelo: Obtivemos os seguinte resultados para o *Decision Tree Classifier*.

	0	1
0	49	5
1	12	24

Figura 21. Matriz confusão do modelo.

Analisando a matriz confusão verificamos que obtivemos 49 casos *True positive*, 5 casos *False Negative*, 12 casos *False Positive* e 24 casos *True Negative*.

Para este modelo obtivemos uma precisão de 77.9%, um recall de 76.7%, um valor de f1-score 76.5% e uma accuracy de 77.8%.

E. Comparação entre os modelos

	precision	recall	f1-score	accuracy
NN	77.9	76.7	76.5	88.8
LR	77.6	77.8	77.5	77.8
RFC	81.3	81.1	80.7	81.1
DTC	77.9	76.7	76.5	77.8

Figura 22. Valores de precision, recall, f1-score e accuracy dos modelos implementados.

Podemos observar que o modelo que obteve a melhor accuracy foi *Neural Networks* com o valor de 88.8%, o modelo com o melhor f1-score foi *Random Forest Classifier*, o valor que obteve o melhor recall foi *Random Forest Classifier* e por fim o modelo que obteve a maior precision foi *Random Forest Classifier*.

V. Conclusão

O nosso melhor modelo foi *Neural Networks* em que obtivemos para a previsão de doença cardíaca obteve um valor de accuracy de 88.7640% para o conjunto de dados de treino, 83.3333% para o conjunto de dados de validação e 88.1356% para o conjunto de dados de teste o que é bom, para além disso obtivemos um valor de 0.93 para a AUC que também é um bom resultado. O modelo *Random Forest Classifier* também obteve um bom valor de accuracy 81.1 que foi bom e obteve os melhores valores para precision, recall e f1-score entre os modelos, 81.3%, 81.1%, 80.7% respetivamente. Podemos então concluir que atingimos o nosso objetivo de distinguir a presença de doença cardíaca.

Comparando os nossos resultados com o trabalho [2] que utilizou *Neural Networks*, neste trabalho foi obtido uma accuracy de 86.4% para os dados de treino e 89.011% para os dados de validação, obtivemos um resultado ligeiramente superior para o conjunto de dados de treino e obtivemos piores resultados para o conjunto de dados de validação comparativamente aos resultados obtidos do trabalho mencionado.

Analisando os resultados do trabalho [3], eles utilizaram vários métodos, *SVM*, *Gaussian Naive Bayes*, *Logistic regression*, *LightGBM*, *XGBoost* e *Random Forest*, entre estes

métodos utilizados por eles, o que apresentou melhores resultados foi *Random Forest* que obteve uma accuracy de 100% para o data set de treino e 88.5% para o data set de teste. Este resultados são superiores aos obtidos no nosso trabalho, comparando com o nosso modelo *Random Forest*, obtivemos piores resultados tanto para os dados de treino como para os dados de teste.

Para melhorar os nossos resultados no modelo *Neural Networks* poderíamos alterar o número de layers assim como alterar os parâmetros das mesmas. Para melhorar os nossos resultados no modelo *Random Forest Classifier* podemos aumentar o número de valores dos hiperparâmetros usados no *Cross Validation* para encontrar melhores hiperparâmetros e utiliza-los então na construção do modelo.

REFERÊNCIAS

[1] https://archive.ics.uci.edu/ml/datasets/Heart+Disease

[2] https://www.researchgate.net/publication/220215545_Effective_diagnosis_of_heart_disease_through_neural_networks_ensembles

[3] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9085310/