Decifrando o Código Cardíaco

Ana Fernanda Souza Cancado
PUC Minas
afscancado@sga.pucminas.br

Arthur de Sá Braz de Matos
PUC Minas
amatos@sga.pucminas.br

Gabriel Praes Bernardes Nunes
PUC Minas
gabriel.praes@gmail.com

Guilherme Otávio de Oliveira
PUC Minas
gooliveira@sga.pucminas.br

Pedro Augusto Gomes Ferreira de Albuquerque
PUC Minas
pedroaugusto19017@gmail.com

Vitória Símil Araújo
PUC Minas
vitoria.araujo.1321449@sga.pucminas.br

ABSTRACT

VERSÃO FINAL

KEYWORDS

Machine Learning, doenças cardíacas, análise de dados, big data, algoritmos de aprendizado de máquina

ACM Reference Format:
Ana Fernanda Souza Cancado, Arthur de Sá Braz de Matos, Gabriel Praes Bernardes Nunes, Guilherme Otávio de Oliveira, Pedro Augusto Gomes Ferreira de Albuquerque, and Vitória Símil Araújo. 2024. Decifrando o Código Cardíaco. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUÇÃO

VERSÃO FINAL

2 MATERIAIS

Base de Dados

VERSÃO FINAL

3 METODOLOGIA

O que foi feito

Após a análise da base de dados, que aborda diversos fatores relacionados à suscetibilidade de uma pessoa a ataques cardíacos, foi realizado um processo de pré-processamento, detalhado na seção seguinte. Em seguida, aplicamos diferentes métodos de Machine Learning para treinar o modelo, capacitando-o a classificar novas instâncias com base no risco de ocorrência de um ataque cardíaco. Foram utilizados bibliotecas Python para realizar tais tarefas, que tiveram sucesso em todos os casos, com pequenas diferenças nos resultados.

Pré-Processamento

Na visualização inicial da base de dados, foi perceptível que ela já estava balanceada em relação ao "target". Ao utilizar a função data.info() do Python para obter uma visão geral do tipo de dado em cada coluna e se existiam valores nulos, foi constatado que não havia valores ausentes e que apenas o atributo "oldpeak" era contínuo, enquanto outros eram todos discretos.

Ao criar um gráfico de boxplot para pressão arterial em repouso (trestbps), colesterol (chol), e frequência cardíaca máxima atingida (thalach), foi possível identificar os seguintes outliers:

(1) "Trestbps": possui alguns outliers acima do valor de 180.
(2) "Chol": apresenta outliers significativos acima de 300, com alguns valores extremos que chegam até 500.
(3) "Thalach": apresenta poucos outliers abaixo de 100

Após essa análise, foi realizada a normalização de todas as colunas de x_{treino} e x_{teste}, ou seja, das colunas que contêm os atributos que não são o alvo da classificação. Essa normalização foi feita utilizando o MinMaxScaler, que transforma os valores de cada coluna para um intervalo entre 0 e 1, com base nos valores mínimos e máximos de cada coluna. A normalização é importante para evitar que variáveis com diferentes escalas influenciem de forma desproporcional o desempenho dos modelos.

4 ALGORITMOS USADOS

Geral

(1) **Pandas:** Utilizado para manipulação e análise de dados.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

(2) **Matplotlib:** Visualização de dados.
(3) **Accuracy Score:** Proporção de previsões corretas em relação ao total de previsões realizadas.
(4) **Confusion Matrix e ConfusionMatrixDisplay:** Criação e visualização da matriz de confusão.
(5) **RandomizedSearchCV e GridSearchCV:** Métodos de otimização de hiperparâmetros.
(6) **Classification Report:** Ferramenta para avaliar o desempenho do modelo de classificação.

Decision Tree

(1) **DecisionTreeClassifier:** Modelo de treinamento.

Table 1: Resultados obtidos para a classe positiva

	Precisão	Recall	F1-Score	Acurácia
Normal	0.92	0.75	0.83	0.84
Grid Search	0.84	0.81	0.83	0.82
Random Search	0.79	0.72	0.75	0.75

Para a Árvore de Decisão, o modelo possui um recall de 0.93 para a classe 0, indicando que é ótimo para detecção de casos sem doença cardíaca. A precisão dessa mesma classe é de 0.77, o que mostra que 23% dos casos foram classificados "sem doença cardíaca" de maneira incorreta. Para a classe 1, a precisão é de 0.92, ou seja, o modelo quase sempre acerta ao classificar alguém como "com doença", porém, com um recall de 0.75, mostrando que detecta corretamente 75% dos casos de doença. Com isso, é perceptível que esse modelo é bom para evitar falsos positivos, mas possui uma taxa de falsos negativos alta, o que não é ideal para o contexto médico da base de dados. Ao utilizar o Random Search, não foi observado melhorias. Houve queda na taxa de detecção de casos sem doença e de casos com doença. Já com o Grid Search, apesar de ter uma queda no recall da classe 0 e na precisão da classe 1, foi obtido as menores taxas de falso positivo e de falso negativo. Ou seja, mesmo com a acurácia sendo um pouco inferior, ao utilizar o Grid Search na Árvore de Decisão, é possível minimizar falsos diagnóstivos de doença cardíaca e melhorar a detecção de casos reais, o que é o preferível nesse caso.

Random Forest

(1) **RandomForestClassifier:** Modelo de treinamento.
(2) **Randint:** Definir distribuições aleatórias de valores inteiros dentro de um intervalo.

Com o modelo de Random Forest, para a classe 0, o recall obtido foi de 0.83, ou seja, os casos de ausência de doenças estão sendo bem capturados. A precisão também foi 0.83, mostrando que 17% dos casos que foram classificados como sem doença estão incorretos. Na classe 1, os resultados foram parecidos, o recall foi de 0.84, indicando que o modelo está falhando em 16% dos casos positivos e com a precisão de 0.84, conclui-se que 16% estão como falsos positivos. Com isso, é possível ver que o modelo está balanceado, com taxas de detecção de doença e de ausência de doença muito próximas. Porém, com taxa de falso negativo alta para um contexto médico. Ao implementar o Random Search, o recall e a precisão das duas classes melhoraram. A taxa de falsos negativos caiu, tornando o modelo mais eficaz. Com o Grid Search, a precisão da classe 0 e o recall da classe 1 tiveram uma pequena redução, apesar de ainda apresentar um bom desempenho. Com essa análise, conclui-se que o melhor resultado com a Random Forest foi obtido utilizando Random Search, já que conseguimos uma diminuição de falsos positivos e de falsos negativos.

Table 2: Resultados obtidos para a classe positiva

	Precisão	Recall	F1-Score	Acurácia
Normal	0.84	0.84	0.84	0.84
Grid Search	0.84	0.81	0.83	0.82
Random Search	0.88	0.91	0.89	0.89

Naive Bayes

(1) **GaussianNB:** Modelo de treinamento.
(2) **Logspace:** Criação uma sequência de números espaçados logaritmicamente em uma escala especificada.

Table 3: Resultados obtidos para a classe positiva

	Precisão	Recall	F1-Score	Acurácia
Normal	0.90	0.84	0.87	0.87
Grid Search	0.88	0.91	0.89	0.89
Random Search	0.88	0.91	0.89	0.89

Ao analisar o resultado do modelo Naive Bayes, é possível concluir que para a classe 0, (sem doença cardíaca) o modelo tem um recall alto, de 0.90, ou seja, está está detectando corretamente a maioria dos casos negativos. Porém, possui um precisão de 0.84, isso quer dizer que, entre todos os exemplos que foram classificados como sem doença cardíaca, 16% foram classificados incorretamente. Para a classe 1 (com doença cardíaca), a precisão é de 0.90, ou seja, quando ele classifica um caso

como tendo a doença, na maioria das vezes está correto. Porém, com o recall sendo 0,84, modelo não consegue detectar cerca de 16% dos casos reais de doença. Portanto, o modelo é bom em minimizar falsos positivos, o que é importante para evitar diagnósticos incorretos de doença cardíaca. No entanto, apresenta uma taxa maior de falsos negativos, que pode ser um risco para uma aplicação médica. Ao implementar o Random Search e o Grid Search, observou-se uma melhoria na detecção de casos da classe 1. A taxa de falsos negativos dessa classe caiu, mostrando que os modelos otimizados são melhores para identificar pacientes que realmente tem a doença. Porém, houve um pequeno aumento nos falsos positivos, mas essa troca é justificada pela redução dos falsos negativos na classe.

5 CÓDIGOS DESENVOLVIDOS

Decision Tree

Clique aqui para visualizar o algoritmo.

Random Forest

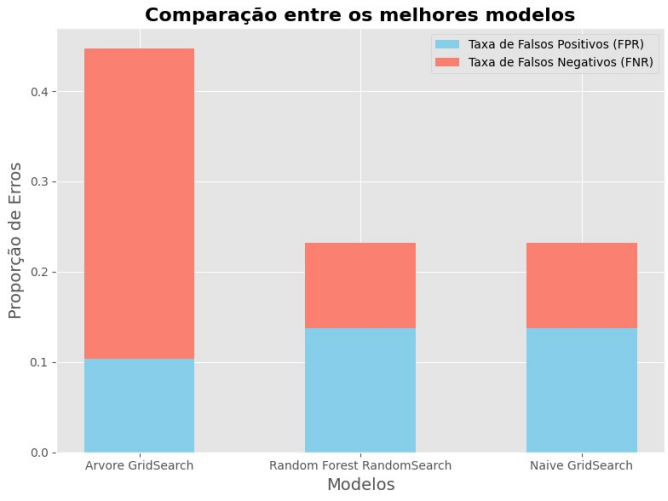
Clique aqui para visualizar o algoritmo.

Naive Bayes

Clique aqui para visualizar o algoritmo.

6 RESULTADOS E DISCUSSÕES

Métricas de Avaliação e Qualidade



Ao observar o gráfico comparando as taxas de falsos positivos (TFP) e falsos negativos (TFN) para os três modelos avaliados, que são os melhores para cada algoritmo, fica evidente que a Árvore de Decisão com GridSearch possui a maior proporção de erros totais, com uma TFN de aproximadamente 0.43, o que corrobora com a análise de que o modelo apresenta dificuldades em evitar falsos negativos, o que é crítico para aplicações médicas. Embora tenha boa precisão para a classe 0 (sem doença), a taxa de falsos negativos alta compromete seu uso. O modelo de Random Forest com RandomSearch apresenta a menor taxa de erros, com uma TFN e TFP equilibradas, sugerindo que esse método otimizado melhora a detecção de casos sem doença, além de reduzir os falsos negativos, o que é preferível. Assim, é o mais eficiente em reduzir diagnósticos incorretos, especialmente para pacientes com doença, ao mesmo tempo que mantém uma taxa aceitável de falsos positivos. Por fim, o modelo Naive Bayes com GridSearch também teve uma redução significativa de falsos negativos, como evidenciado pela menor TFN em comparação com a árvore de decisão, porém, com um leve aumento na taxa de falsos positivos, o que é uma troca aceitável dado o contexto clínico. Portanto, o Random Forest com RandomSearch se destaca como o método mais balanceado e ideal, apresentando um desempenho superior tanto na minimização de falsos positivos quanto falsos negativos, enquanto a Árvore de Decisão otimizada apresentou a maior taxa de erro.

7 UTILIZAÇÃO DO CHAT GPT

VERSÃO FINAL

REFERENCES

VERSÃO FINAL