**Lucas Meller**

Predição de Doença Cardíaca

**26/11/2024**

Link do Repositório: <https://github.com/LucasMeller13/ml_abp>

# Introdução do Problema

Com o aumento do sedentarismo e consumo de alimentos não saudáveis, ocorrências de doenças cardíacas estão em ascensão prejudicando a saúde e longevidade daqueles que praticam tais hábitos. Com isso, a equipe decidiu desenvolver um sistema classificatório que fizesse a predição de quais pacientes iriam ter doenças cardíacas por meio de métricas de avaliação de saúde.

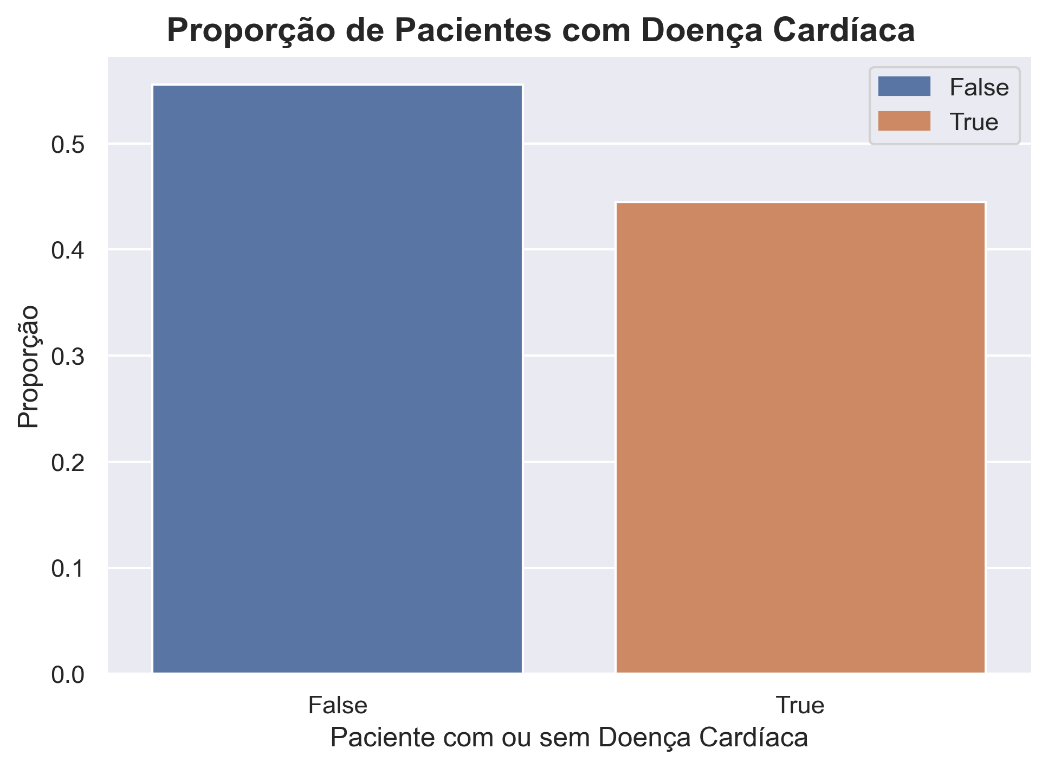
Para tal, foi escolhido um dataset da *University of California Irvine’s ML Repository* com as seguintes informações:

|  |  |
| --- | --- |
| * Idade * Sexo * Tipo de Dor no Peito * Pressão Arterial * Colesterol * Glicose em Jejum acima de 120 * Resultados EKG | * Frequência Cardíaca Máxima * Angina Induzida por Exercício * Depressão ST * Inclinação de ST * Número de Vasos Fluoroscópicos * Talio * Doença Cardíaca (*target*) |

Para avaliação de performance dos modelos, a métrica *recall* foi a escolhida já que busca minimizar os falsos negativos e, desse forma, evitar que alguém com indícios de problemas cardíacos seja diagnosticado como saudável e descubra tardiamente sua doença.

# Análise do Dataset

Foram desenvolvidos gráficos para análise de quais variáveis possuíam correlação com o *target* e, por conseguinte, avaliar quais modelos seriam mais adequados aos dados. No conjunto de dados escolhido, os mesmos já estavam tratados e com o *target* balanceado de forma satisfatória.



*Figura 1. Proporção variável target*

Para avaliação de distribuição das variáveis contínuas com relação ao *target*, foram feitas as seguintes análises:

Gráfico, Gráfico de linhas

Descrição gerada automaticamente

Figura . Relação Variáveis Independentes Contínuas vs Target

Nota-se que a distribuição do *target* com relação as variáveis Idade, Frequência Cardíaca Máxima e Depressão ST diferem entre pacientes com ou sem doença, significando dessa forma uma possível capacidade de separação de futuros clientes usando tais colunas.

Além disso, foram criados gráficos para análise das relações entre *target* e variáveis discretas para as colunas Sexo, Tipo de Dor no Peito, Glicose em Jejum Acima de 120, Resultados EKG, Angina Induzida por Exercício, Inclinação ST, Número de Vasos Fluoroscópicos e Talio.

Gráfico, Gráfico de barras

Descrição gerada automaticamente

Figura . Relação Variáveis Independentes Discretas vs Target

De acordo com a Figura 3, nos gráficos relacionados ao Talio, Tipo de Dor no Peito e Número de Vasos Fluoroscópicos há uma clara separação entre clientes com e sem doença cardíaca e, consequentemente, tornando possível a classificação de novos clientes usando essas colunas como será visto posteriormente no modelo RandomForestClassifier.

Para verificação das relações entre todas as variáveis presentes no *dataset* foi utilizada a matriz de correlação para verificação de tais métricas.

Gráfico, Gráfico de mapa de árvore

Descrição gerada automaticamente

Figura . Matriz de Correlação

Por fim, decidiu-se usar a separação *Train-Test-Val* no qual o subgrupo de teste foi usado para calibração de hiper parâmetros e o de validação para verificação de performance final.

# Seleção de Features e Feature Engineering

Para seleção de *features*, decidiu-se usar todas as colunas já que não havia dados nulos e que iriam atrapalhar para criação dos modelos. Já para *feature engineering*, foram usados *pipelines* com PolynomialFeatures para criação de novos colunas e StandardScaler para padronização de média e desvio padrão, evitando que variáveis se interferissem.

Texto

Descrição gerada automaticamente

Figura . Colunas antes do StandardScaler

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Figura . Colunas após StandardScaler

# Treinamento e Fine Tuning (Grid Search obrigatório)

Para o treinamento, foi usado um pipeline com PolynomialFeatures, StandardScaler, PCA e o modelo para padronização de procedimentos.

Texto

Descrição gerada automaticamente

Figura . Exemplo de treinamento de modelo

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

Figura . Parâmetros utilizados nos modelos

Como visto acima, foram testados 5 modelos (LogisticRegression, SVC, RandomForest, KNN e CatBoost) com intervalos de parâmetros específicos a cada um. Para cada intervalo, o GridSearch testa valores e retorna o melhor modelo dada a métrica escolhida, sendo nesse caso o *recall*. Optou-se pelo RandomizedSearchCV já que usa amostragens representativas dos parâmetros mesmo sem testar todos do intervalo, reduzindo de forma significativa o tempo de treinamento.

Inicialmente, todos os modelos foram testados apenas com o StandardScaler como pré-processamento dos dados para definição de uma baseline de performance de cada modelo.

Gráfico, Gráfico de barras

Descrição gerada automaticamente

Figura . Performance dos Modelos apenas com StandardScaler

Após isso, aplicou-se PolynomialFeatures de grau 3 em cada pipeline, resultando na seguinte performance:

Gráfico, Gráfico de barras

Descrição gerada automaticamente

Figura . Performance dos Modelos com StandardScaler e PolynomialFeatures

Por fim, foi aplicado PCA em cada pipeline com 30 componentes, resultando na seguinte performance:

Gráfico, Gráfico de barras

Descrição gerada automaticamente

Figura . Performance dos Modelos com PolynomialFeatures, PCA e StandardScaler

Após os testes citados, chegou-se à conclusão de que o melhor modelo treinado foi o RandomForestClassifier com PolynomialFeatures e StandardScaler (foi mantido por padronização de pipelines, mas para modelos de árvores classificatórios a padronização da distribuição dos dados não faz diferença).

Para melhor compreensão de quais variáveis foram importantes para o modelo escolhido, foram plotados gráficos com e sem PolynomialFeatures para compreensão das escolhas tomadas para redução da entropia presente.

Gráfico, Gráfico de barras

Descrição gerada automaticamente

Figura . Importância das Features para modelo - Sem PolynomialFeatures

# Gráfico, Gráfico de barras Descrição gerada automaticamente

Figura . Importância das Features para o modelo - Com PolynomialFeatures

Como visto na Figura 12, as cinco variáveis que trouxeram maior poder preditivo para o modelo foram a coluna Talio, Tipo de Dor no Peito, Depressão ST, Número de Vasos Fluoroscópicos e Frequência Cardíaca Máxima. Contudo, usando o método de criação de novas variáveis polinomiais é possível descobrir novas relações entre as colunas que trazem ainda mais capacidade preditiva ao modelo.

Na Figura 13, após aplicação do método PolynomialFeatures de grau 3, o modelo passou a ter uma maior pontuação de *recall* já que com as novas variáveis, novas distribuições possibilitaram o modelo de distinguir com mais clareza qual paciente tem doenças cardíacas. Segue abaixo um gráfico com as distribuições das quatro melhores variáveis polinomiais, seguidos pela variável com pior distribuição.

Gráfico, Gráfico de linhas

Descrição gerada automaticamente

Figura . Distruição das Top 4 Variáveis Polinomiais do RandomForest

Gráfico, Gráfico de linhas

Descrição gerada automaticamente

Figura . Distribuição da Pior Variável Polinomial do RandomForest

# Avaliação

A avaliação da pontuação de *recall* para cada modelo foi feita por meio do *split* de validação, os quais performaram seguindo as Figuras 9,10 e 11. Como citado anteriormente, o melhor modelo treinado foi o RandomForest com PolynomialFeatures e StandardScaler, sendo o escolhido para a API.

Gráfico, Gráfico de barras

Descrição gerada automaticamente

Figura . Pontuação de Recall

# Modelo em Produção

Com o modelo exportado em formato *pickle*, foi criado uma API em Flask a qual recebe dados via requisição POST e retorna 0 (sem doença cardíaca) ou 1 (com doença cardíaca).

Texto

Descrição gerada automaticamente

Figura . Código API

|  |  |
| --- | --- |
| Texto  Descrição gerada automaticamente | Tela de celular com fundo preto  Descrição gerada automaticamente |

Tabela . Requisição POST e resposta API

# 

# Conclusão

Apesar do *dataset* possuir poucos cadastros (270 pacientes), por meio de técnicas de separação de dados e processamento de dados foi possível demonstrar a criação de modelos com performances consideráveis para o ramo médico, prevenindo dessa forma que pacientes identifiquem problemas médicos tardiamente.