



Instituto Tecnológico de Aeronáutica - ITA
Data Science Practices - CMC-16

Alunos: Othon Daiki Ishiyi, Samir Nunes da Silva, Ulisses Lopes da Silva e Victor Antônio Batista Caus

Relatório do Projeto de CMC-16 - Heart Disease Prediction

1 Introdução

O objetivo do presente projeto é a criação de um modelo de aprendizado de máquina para realizar a previsão da presença de doenças cardíacas em pacientes, de maneira a auxiliar médicos no diagnóstico clínico, com base em resultados de exames realizados. Como requisitos estão a validação do modelo segundo métricas selecionadas e seu deploy.

A base de dados utilizada para o projeto está disponível no repositório de machine learning da Universidade da Califórnia em Irvine: <https://archive.ics.uci.edu/dataset/45/heart+disease>. Foram unidas as 4 bases de dados disponíveis no repositório: Cleveland, Hungary, Switzerland e VA Long Beach, totalizando 920 linhas de dados. O target consiste em uma variável binária que é 0 se não há doença e é 1 se há doença. Ademais, as variáveis utilizadas como features para a previsão são mostradas na Figura 1.

1. `age` : age in years
2. `sex` : sex (1 = male; 0 = female)
3. `cp` : chest pain type -- Value 1: typical angina -- Value 2: atypical angina -- Value 3: non-anginal pain -- Value 4: asymptomatic
4. `trestbps` : resting blood pressure (in mm Hg on admission to the hospital)
5. `chol` : serum cholestoral in mg/dl
6. `fbs` : (fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)
7. `restecg` : resting electrocardiographic results -- Value 0: normal -- Value 1: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV) -- Value 2: showing probable or definite left ventricular hypertrophy by Estes' criteria
8. `thalach` : maximum heart rate achieved
9. `exang` : exercise induced angina (1 = yes; 0 = no)
10. `oldpeak` : ST depression induced by exercise relative to rest
11. `slope` : the slope of the peak exercise ST segment -- Value 1: upsloping -- Value 2: flat -- Value 3: downsloping
12. `ca` : number of major vessels (0-3) colored by flourosopy
13. `thal` : 3 = normal; 6 = fixed defect; 7 = reversable defect

Figura 1: Variáveis utilizadas como features para a previsão de doença cardíaca em pacientes.

Quanto às métricas, propôs-se, em média, o mínimo de 75% de revocação e 70% de precisão para o modelo final. Espera-se, portanto, que o modelo seja principalmente capaz de evitar falsos negativos. No entanto, ele também deve evitar que muitos pacientes sejam erroneamente classificados como doentes.

2 Desenvolvimento

2.1 Repositório

Para o desenvolvimento do projeto, criou-se um repositório no GitHub, disponível em: <https://github.com/Samirnunes/heart-disease-prediction>.

2.2 Ferramentas Utilizadas

Utilizou-se a linguagem Python para o desenvolvimento do *software*. Para tratamento e manipulação de dados, utilizou-se a biblioteca Pandas. Já para criação de modelos de aprendizado de máquina, utilizou-se a biblioteca Scikit-Learn. Para tratar o balanceamento dos dados, valeu-se da biblioteca Imblearn. Por fim, realizou-se o deploy através da biblioteca Flask.

2.3 Análise dos Dados

Realizou-se uma análise inicial dos dados no notebook `data_analysis.ipynb`. Como resultados, foram obtidos, entre outros, o gráfico de barras da distribuição do target, por meio do qual se verificou leve desbalanceamento entre as classes, e o mapa de calor das correlações entre features e target.

2.4 Pipeline de Pré-processamento

Criou-se a classe `HdpDataPipeline` para funcionar como pipeline de pré-processamento dos dados do problema. Nela, encapsulou-se as seguintes operações:

- Imputação da média nas variáveis numéricas;
- Imputação da moda nas variáveis categóricas;
- Aplicação de `MinMaxScaler` para escalar os dados para o intervalo $[0, 1]$;

Quando o pipeline é aplicado em dados após o fit no treino, realiza-se, juntamente às operações anteriores, o clipping dos valores das features escaladas para o intervalo $[0, 1]$ e a verificação de se pelo menos 50% das variáveis foram supridas para o pipeline - se não forem, um erro ocorre, solicitando o preenchimento de mais valores. Finalmente, no momento do treinamento do modelo (classe `HdpModelTrainer`), realiza-se *oversampling* via SMOTE (Synthetic Minority Oversampling Technique), de forma a balancear as classes do target.

2.5 Escolha do Modelo: Validação

Realizou-se a validação de 6 modelos, os quais são listados a seguir juntamente de seus parâmetros (utilizou-se `random_state = 100`):

- `RandomForestClassifier(n_estimators=100, random_state=random_state, max_depth=3, max_leaf_nodes=10);`
- `DecisionTreeClassifier(random_state=random_state, max_depth=3, max_leaf_nodes=10);`
- `LogisticRegression();`
- `SVC(probability=True, random_state=random_state);`
- `XGBClassifier(random_state=random_state);`
- `KNeighborsClassifier(n_neighbors=5).`

Utilizou-se como processo de validação a verificação dos histogramas de revocação e de precisão gerados a partir de 10 validações cruzadas via 10-fold. A condição para validação é de que a média menos um desvio-padrão deve ser pelo menos maior que o limiar mínimo considerado (75% de revocação e 70% de precisão).

Verificou-se, com isso, que os dois modelos que passaram pela validação foram **RandomForest** e **LogisticRegression**. Seus histogramas podem ser vistos no repositório do projeto.

Ao fim, escolheu-se **LogisticRegression** como modelo para deploy, pelo fato ser menos propenso ao *overfitting*. Para a análise do *overfitting*, comparou-se as médias das métricas nos folds de treino com as médias das métricas no fold de teste.

2.6 Teste do Modelo

Feita a escolha da **LogisticRegression** como modelo, realizou-se a previsão no conjunto de teste e a obtenção das métricas, mostradas na Tabela 1.

Precision	0.85
Recall	0.81

Tabela 1: Precisão e revocação do modelo para deploy no conjunto de teste.

Os resultados corroboram com a validação realizada.

2.7 Treinamento Final e Deploy

Finalmente, realizou-se o treinamento do modelo em todas as 920 observações do conjunto de dados e seu deploy em uma página web criada via Flask, HTML e CSS. A previsão através do modelo é feita via um endpoint nomeado `predict_heart_disease`, que recebe os valores de cada uma das features e retorna 0 ou 1 para a previsão da doença. Além disso, foi adicionada a possibilidade de feedback por parte do médico - ele pode selecionar qual saída acha a correta, de forma que é possível comparar com o resultado do modelo e usar tal feedback futuramente para melhorá-lo.