**Documentação da Aplicação de Previsão de Ataque Cardíaco**

**Visão Geral**

Esta aplicação é composta por um back-end em Flask e um front-end em HTML, CSS e JavaScript. O objetivo é prever o risco de ataque cardíaco de um usuário com base em entradas de dados.

**Estrutura do Projeto**

heart\_attack\_prediction/

│

├── app.py

├── evaluate\_model.py

├── expected\_columns.txt

├── heart\_attack\_prediction\_dataset.csv

├── index.html

├── knn\_heart\_attack\_model.pkl

├── label\_encoders.pkl

├── preprocess.py

├── processed\_heart\_attack\_prediction\_dataset.csv

├── scaler.pkl

├── script.js

├── style.css

└── README.md (este arquivo)

**Arquivos e Funções**

1. **app.py**:
   * **Descrição**: Arquivo principal que define a API Flask.
   * **Função**: Contém o endpoint /predict que recebe dados em formato JSON, processa-os e retorna uma previsão de risco de ataque cardíaco.
   * **Como Rodar**: python app.py
2. **evaluate\_model.py**:
   * **Descrição**: Script para avaliar o desempenho do modelo.
   * **Função**: Calcula métricas de desempenho como precisão, recall e f1-score.
   * **Como Rodar**: python evaluate\_model.py
3. **expected\_columns.txt**:
   * **Descrição**: Contém a lista de colunas esperadas pelo modelo.
   * **Função**: Garante que os dados de entrada estejam no formato correto.
4. **heart\_attack\_prediction\_dataset.csv**:
   * **Descrição**: Conjunto de dados original utilizado para treinamento e avaliação do modelo.
   * **Função**: Fonte de dados para o treinamento do modelo.
5. **index.html**:
   * **Descrição**: Página HTML que contém o formulário para entrada de dados do usuário.
   * **Função**: Interface do usuário para coletar dados.
6. **knn\_heart\_attack\_model.pkl**:
   * **Descrição**: Arquivo contendo o modelo KNN treinado.
   * **Função**: Usado para fazer previsões de risco de ataque cardíaco.
7. **label\_encoders.pkl**:
   * **Descrição**: Contém os label encoders para variáveis categóricas.
   * **Função**: Converte variáveis categóricas em numéricas.
8. **preprocess.py**:
   * **Descrição**: Script de pré-processamento dos dados.
   * **Função**: Prepara os dados antes de treinar o modelo ou fazer previsões.
   * **Como Rodar**: python preprocess.py
9. **processed\_heart\_attack\_prediction\_dataset.csv**:
   * **Descrição**: Conjunto de dados processado.
   * **Função**: Dados após pré-processamento.
10. **scaler.pkl**:
    * **Descrição**: Contém o scaler para normalização dos dados.
    * **Função**: Normaliza os dados.
11. **script.js**:
    * **Descrição**: JavaScript que captura os dados do formulário e envia para a API.
    * **Função**: Envia dados do formulário para a API e exibe o resultado.
12. **style.css**:
    * **Descrição**: Estilos CSS para a página HTML.
    * **Função**: Estiliza a página HTML.

**Configuração e Execução**

**1. Pré-requisitos**

* Python 3.x
* Pip (gerenciador de pacotes do Python)

**2. Instalação de Dependências**

pip install -r requirements.txt

**3. Treinamento e Avaliação do Modelo**

1. **Pré-processamento dos Dados**:

python preprocess.py

1. **Avaliação do Modelo**:

python evaluate\_model.py

**4. Execução da API Flask**

1. Execute a API Flask:

python app.py

1. A API estará disponível em http://127.0.0.1:5000/predict.

**5. Front-end**

1. Abra o arquivo index.html
2. Preencha o formulário com os dados do usuário e clique em "Prever" para obter a previsão de risco de ataque cardíaco.
3. Use o botão "Testar" para preencher automaticamente o formulário com valores de exemplo e testar a API rapidamente.

**Exemplo de JSON para Teste**

json

{

"Age": 60,

"Sex": "Male",

"Cholesterol": 250,

"Heart Rate": 80,

"Diabetes": 1,

"Family History": 1,

"Smoking": 0,

"Obesity": 1,

"Alcohol Consumption": 0,

"Exercise Hours Per Week": 3,

"Diet": "Healthy",

"Stress Level": 5,

"Systolic Blood Pressure": 130,

"Diastolic Blood Pressure": 85,

"Income": 60000,

"BMI": 27.5,

"Triglycerides": 190,

"Physical Activity Days Per Week": 3,

"Sleep Hours Per Day": 7,

"Country": "USA",

"Continent": "North America",

"Hemisphere": "Northern",

"Medication Use": 1,

"Previous Heart Problems": 1,

"Sedentary Hours Per Day": 6

}

**Fluxo de Dados**

1. **Entrada de Dados**: O usuário preenche o formulário na página index.html.
2. **Envio para a API**: Os dados são capturados pelo JavaScript (script.js) e enviados para a API Flask no endpoint /predict.
3. **Processamento pela API**: A API recebe os dados, os pré-processa (normaliza e codifica variáveis categóricas) e faz a previsão usando o modelo KNN.
4. **Resposta da API**: A API retorna a previsão (risco de ataque cardíaco) para o front-end.
5. **Exibição do Resultado**: O JavaScript exibe o resultado da previsão na página HTML.