Dokumentacja Techniczna Projektu AI: heart-attack-prediction

Wprowadzenie

Cel Projektu

Celem projektu **heart-attack-prediction** jest stworzenie aplikacji, która potrafi przewidywać ryzyko wystąpienia ataku serca na podstawie dostarczonych parametrów pacjenta. Dzięki wykorzystaniu zaawansowanych algorytmów uczenia maszynowego, aplikacja analizuje dane medyczne i dostarcza dokładne prognozy, które mogą wspierać decyzje medyczne i wczesną interwencję.

Znaczenie Przewidywania Ataku Serca

Ataki serca są jedną z głównych przyczyn zgonów na świecie. Wczesne wykrycie osób z wysokim ryzykiem może znacząco poprawić wyniki leczenia i zmniejszyć śmiertelność. Przewidywanie ryzyka ataku serca może pomóc w:

- Wczesnej identyfikacji pacjentów wymagających intensywniejszej opieki.
- Personalizacji planów leczenia.
- Zwiększeniu świadomości pacjentów na temat ich stanu zdrowia i konieczności zmiany stylu życia.

Opis Aplikacji

Aplikacja **heart-attack-prediction** przyjmuje zestaw parametrów medycznych pacjenta i na ich podstawie prognozuje ryzyko wystąpienia ataku serca. Interfejs użytkownika pozwala na łatwe wprowadzanie danych, a zaawansowane algorytmy w backendzie przetwarzają te dane, generując wyniki w czasie rzeczywistym.

Struktura Projektu

Projekt został zorganizowany w sposób modułowy, z podziałem na różne komponenty odpowiedzialne za poszczególne części procesu przewidywania ryzyka ataku serca. Struktura ta obejmuje:

1. **Frontend**: Zrealizowany za pomocą Streamlit, umożliwia interakcję z użytkownikiem.
2. **Backend**: Oparty na FastAPI, łączy warstwę frontendową z backendową i obsługuje logikę aplikacji. Wykorzystuje Kedro do zarządzania przepływem danych i logiką

przetwarzania. AutoGluon do trenowania ML oraz Weights & Biases do monitorowania procesu trenowania.
3. **Baza Danych**: do przechowywania danych wykorzystywane jest baza danych PostgreSQL.

Użyty zbiór danych

Dane są w formacie klucz: wartość.

- **Age**: Wiek pacjenta
- **Sex**: Płeć pacjenta
- **exang**: Dławica wywołana wysiłkiem (1 = tak; 0 = nie)
- **ca**: Liczba głównych naczyń (0-3)
- **cp**: Typ bólu w klatce piersiowej
    - Wartość 1: typowa dławica piersiowa
    - Wartość 2: nietypowa dławica piersiowa
    - Wartość 3: ból nie związany z sercem
    - Wartość 4: bezobjawowy
- **trtbps**: Ciśnienie krwi w spoczynku (w mm Hg)
- **chol**: Cholesterol w mg/dl pobierany za pomocą czujnika BMI
- **fbs**: Poziom cukru na czczo > 120 mg/dl (1 = prawda; 0 = fałsz)
- **rest_ecg**: Wyniki elektrokardiograficzne w spoczynku
    - Wartość 0: normalny
    - Wartość 1: mający nieprawidłowość fali ST-T (odwrócenie fali T i/lub uniesienie lub obniżenie ST o > 0,05 mV)
    - Wartość 2: wykazujący prawdopodobne lub pewne przerosty lewej komory serca według kryteriów Estes'a
- **thalach**: Maksymalne osiągnięte tętno
- **target**:
    - Wartość 0: mniejsze ryzyko ataku serca
    - Wartość 1: większe ryzyko ataku serca

Technologie Użyte do Tworzenia Projektu

Ogólna Architektura

Projekt został umieszczony w kontenerach Docker, z podziałem na frontend, backend oraz bazę danych.

Frontend

Frontend został wykonany przy użyciu frameworku Streamlit, co umożliwia interakcję użytkownika z aplikacją w przystępny i intuicyjny sposób.

Backend

Backend składa się z kilku kluczowych komponentów:

- **FASTAPI**: Służy do połączenia warstwy frontend z warstwą backend, obsługując żądania HTTP i dostarczając odpowiedzi do frontendu.
- **Kedro**: Odpowiada za zarządzanie pipeline'em danych i logiką aplikacji. Kedro zapewnia strukturyzowane podejście do tworzenia pipeline'ów przetwarzania danych, umożliwiając modularność i ponowne wykorzystanie kodu. Pipeliny odpowiadają za:
    - **Przetwarzania danych**: Odpowiada za walidację i przekształcanie danych wejściowych.
    - **Trenowania modeli**: Integruje AutoGluon do automatycznego trenowania modeli ML.
- **AutoGluon**: Wykorzystywany w pipeline Kedro do automatycznego trenowania modeli ML.
- **WanDB**: (Weights & Biases) Używane do śledzenia procesu trenowania modelu, monitorowania wyników i zarządzania eksperymentami.

Baza Danych

Dane są zapisywane w bazie danych PostgreSQL, która jest umieszczona w osobnym kontenerze.

Diagram przedstawiający uproszczoną architekturę aplikacji.