Dokumentacja techniczna

Health Mate

Małgorzata Biziewska s22275

Krzysztof kaleta s20601

Łukasz Rybak s15339

SUML, Warszawa, 2024

Spis treści

Opis aplikacji i uzasadnienie funkcjonalności

Health Mate to aplikacja webowa, która zaprojektowana jest jako osobisty asystent zdrowia, mający na celu zwiększenie świadomości na temat otyłości i chorób sercowo-naczyniowych oraz ich prewencji. Aplikacja analizuje dane dotyczące stylu życia, diety, aktywności fizycznej oraz podstawowych parametrów fizjologicznych użytkownika. Dzięki temu może wskazać potencjalne ryzyko zdrowotne oraz zaproponować zmiany w nawykach, aby przeciwdziałać tym zagrożeniom. Jest to szczególnie ważne, ponieważ liczba przypadków otyłości i chorób sercowo-naczyniowych stale rośnie na całym świecie. Aplikacja ma na celu wspomóc użytkowników w prowadzeniu zdrowszego trybu życia poprzez dostarczanie spersonalizowanych zaleceń. Poprzez śledzenie codziennych aktywności oraz monitorowanie postępów, Health Mate pomaga użytkownikom w osiąganiu ich celów zdrowotnych i utrzymaniu dobrego samopoczucia. Wykorzystuje zaawansowane algorytmy do analizy danych i przewidywania ryzyka, co czyni ją nieocenionym narzędziem w walce z chorobami cywilizacyjnymi.

Aplikacja działa na wszystkich przeglądarkach internetowych, co zapewnia jej kompatybilność z aktualnymi urządzeniami. Aplikacja zawiera demonstracyjny program, który korzysta z hostingu Streamlit oraz kontenerów Docker. Dzięki zastosowaniu Dockera, aplikację można uruchomić bez potrzeby instalowania dodatkowych pakietów, co upraszcza proces wdrożenia i zapewnia spójność środowiska. Docker umożliwia izolację środowiska, automatyzację procesów i łatwość skalowania, co sprawia, że aplikacja jest bardziej niezawodna i łatwiejsza w utrzymaniu.

Dane:

Źródło: https://www.kaggle.com/datasets/aravindpcoder/obesity-or-cvd-risk-classifyregressorcluster/code

Poniżej zostały opisane zmienne użyte w procesie treningu.

Zmienna	**Opis**
Gender	Płeć
Age	Wiek
Height	Wzrost
Weight	Waga
Family_history_with_overweight	Historia nadwagi w rodzinie
FAVC	Częste spożywanie wysokokalorycznych potraw
FCVC	Częstotliwość spożywania warzyw
NCP	Liczba głównych posiłków
CAEC	Spożywanie jedzenia między posiłkami
SMOKE:	Palenie papierosów
CH2O	Dzienne spożycie wody
SCC	Monitorowanie spożycia kalorii
FAF	Częstotliwość aktywności fizycznej
TUE	Czas spędzony na korzystaniu z urządzeń technologicznych
CALC	Spożycie alkoholu
MTRANS	Środek transportu jaki najczęściej wybiera osoba

Zmienna predykcyjna:

Nobesity - kategoria otyłości
- Niedowaga: mniej niż 18.5
- Normalna waga: od 18.5 do 24.9
- Nadwaga: od 25.0 do 29.9
- Otyłość I stopnia: od 30.0 do 34.9
- Otyłość II stopnia: od 35.0 do 39.9
- Otyłość III stopnia: powyżej 40

Przetworzenie danych

Zbiór danych połączono z repozytorium. Po wczytaniu datasetu wykonano następujące kroki, takie jak czyszczenie danych, usunięto duplikaty, zmienne kategoryczne zostały zamienione na zmienne numeryczne, dane zostały znormalizowane. Z datasetu została usunięta zmienna predykcyjna a dataset został podzielny na zbiór treningowy i testowy.

Wykorzystane technologie:

Python

Python to interpretowany język programowania o otwartym kodzie źródłowym, który jest powszechnie używany w analizie danych, uczeniu maszynowym, oraz automatyzacji. Jego czytelna składnia i rozbudowany ekosystem bibliotek sprawiają, że jest idealnym wyborem do tworzenia prototypów i wdrażania rozwiązań machine learning.

Scikit-learn

Scikit-learn to biblioteka w Pythonie, która dostarcza prostych i efektywnych narzędzi do analizy danych oraz modelowania predykcyjnego. Zawiera implementacje algorytmów machine learning, w tym klasyfikacji, regresji, klasteryzacji i redukcji wymiarów. Jest łatwa w użyciu i dobrze zintegrowana z innymi bibliotekami, takimi jak numpy i scipy.

Pandas

Pandas to biblioteka w Pythonie, która dostarcza struktury danych i narzędzia do pracy z danymi tabelarycznymi. Umożliwia łatwe manipulowanie danymi, analizę i czyszczenie danych. Jej główne struktury danych, DataFrame i Series, są fundamentalne dla pracy z dużymi zestawami danych, umożliwiając efektywne zarządzanie i przekształcanie danych.

Streamlit

Streamlit to biblioteka w Pythonie, która umożliwia szybkie tworzenie interaktywnych aplikacji webowych do prezentacji wyników analiz i modeli machine learning. Jest zaprojektowana z myślą o prostocie i szybkim prototypowaniu, co pozwala na przekształcenie skryptów Pythona w interaktywne aplikacje bez potrzeby posiadania zaawansowanej wiedzy na temat front-endu.

Docker

Docker to platforma służąca do tworzenia, zarządzania i uruchamiania aplikacji w kontenerach. Kontenery to lekkie, przenośne jednostki, które zawierają wszystko, co jest potrzebne do uruchomienia aplikacji: kod, runtime, system operacyjny, biblioteki i inne zależności. Docker ułatwia proces tworzenia aplikacji, testowania ich w różnych środowiskach oraz wdrażania na serwery produkcyjne, zapewniając, że aplikacja będzie działała tak samo niezależnie od otoczenia.

Jupyter Notebook

Jupyter Notebook to interaktywne środowisko do tworzenia i udostępniania dokumentów, które zawierają kod, wizualizacje, narratywny tekst i równania matematyczne. Jest szczególnie popularny wśród naukowców danych, badaczy i deweloperów, ponieważ umożliwia wykonywanie kodu w czasie rzeczywistym, analizę danych i tworzenie dynamicznych wykresów.

Działanie aplikacji

Główna funkcjonalność aplikacji polega na przewidywaniu stopnia otyłości na podstawie danych wprowadzonych przez użytkownika i składa się z etapów

- Wprowadzanie danych: Użytkownik korzysta z interfejsu aplikacji, aby wprowadzić swoje informacje.
- Przetwarzanie: Wprowadzone dane są przetwarzane przez aplikację.
- Analiza i obliczenia: Model aplikacji analizuje dane użytkownika i oblicza przewidywany poziom otyłości.
- Prezentacja wyników: Wyniki analizy są przedstawiane użytkownikowi.

Konfiguracja środowiska i uruchomienie aplikacji

1. **Pobierz i zainstaluj Python**
    - Upewnij się, że masz zainstalowaną najnowszą wersję Pythona. Możesz ją pobrać z python.org.

2. **Skonfiguruj środowisko wirtualne**
    - Otwórz terminal lub wiersz poleceń i przejdź do katalogu, w którym znajduje się projekt:

      cd /path/to/your/directory

    - Utwórz środowisko wirtualne:

      python -m venv env

    - Aktywuj środowisko wirtualne:
        - Na Windows:

          env\Scripts\activate

        - Na macOS/Linux:

source env/bin/activate

- Zainstaluj wymagane pakiety:

  pip install -r requirements.txt

3. **Wytrenuj model**
   - Jeśli katalog ml_models jest pusty, przed uruchomieniem aplikacji wykonaj następującą komendę aby uruchomić model:

   python .\model_training\model_train.py

4. **Uruchom aplikację**
   - Aby uruchomić aplikację, użyj poniższej komendy:

     streamlit run app.py

Docker (opcjonalnie)

1. **Budowanie obrazu Dockera**
   - Aby zbudować obraz Dockera, uruchom:

     docker build -t healthmate_app .

2. **Uruchomienie kontenera Dockera**
   - Aby uruchomić aplikację w kontenerze Docker, wykonaj:

     docker run -p 8501:8501 healthmate_app

Struktura projektu

SUML_PROJECT/

├ data/ # Folder zawierający dane wejściowe do analizy

└── ObesityDataSet.csv # Plik CSV z danymi dotyczącymi otyłości

├data_preparation/ # Folder z plikami do przygotowania danych

└──__init__.py # Plik inicjalizujący moduł data_preparation

└── data_prep.py # Skrypt do przygotowania i przetworzenia danych wejściowych

├env/ # Folder zawierający środowisko wirtualne

├ ml_models/ # Folder z wytrenowanymi modelami machine learning

└── random_forest_model.pkl # Wytrenowany model Random Forest zapisany w formacie .pkl

├ model_training/ # Folder z plikami do trenowania modeli

└── __init__.py # Plik inicjalizujący moduł model_training

└──model_train.py # Skrypt do trenowania modeli machine learning

├.gitignore # Plik konfiguracyjny Git ignorujący niepotrzebne pliki i foldery

├app.py # Główny plik aplikacji Streamlit

├avocado.png # Przykładowy plik graficzny używany w aplikacji

├Dockerfile # Plik konfiguracyjny Docker do budowania obrazu kontenera

├README.md # Plik README zawierający opis projektu

├requirements.txt # Plik z listą zależności Pythona potrzebnych do uruchomienia aplikacji