**Dokumentacja aplikacji szacującej czas spędzony na siłowni**

*s25228 Łukasz ZIeliński, s24600 Kacper Bocian, s24629 James Herbert, s24682 Piotr Salamoński*

*Link do repozytorium -* [*https://github.com/s24600/Suml\_sumy*](https://github.com/s24600/Suml_sumy)

# 1. Cel aplikacji

Celem aplikacji jest predykcja liczby godzin, jaką osoba spędza na siłowni w zależności od rożnych czynników (takich jak wiek, płeć, waga, typ ćwiczeń itp.) na podstawie datasetu ze strony Kaggle, na którym został wytrenowany model.

# 2. Postać danych

Zestaw danych pochodzący ze strony Kaggle został wykorzystany do wytrenowania modelu. Oryginalny zestaw posiadał atrybuty:

* Wiek
* Płeć
* Waga (kg)
* Wzrost (m)
* Maksymalne tętno – maksymalna częstotliwość bicia serca
* Średnie tętno – średnia częstotliwość bicia serca
* Tętno spoczynku – częstotliwość bicia serca przed treningiem
* Długość pojedynczej sesji treningowej (godziny)
* Liczba spalonych kalorii
* Typ ćwiczeń (Cardio, Strength, Yoga, HIIT)
* Procent tkanki tłuszczowej
* Spożycie płynu – liczba litrów wody spożytej podczas treningu
* Częstotliwość treningów w tygodniu (liczba dni w tygodniu)
* Stopień doświadczenia (od początkującego (1) do profesjonalisty (4))
* BMI

W celu przeczyszczenia danych, kolumny „Liczba spalonych kalorii” oraz „Spożycie płynu” zostały usunięte, gdyż były to dane w okresie tygodnia, które sugerowały długość ćwiczeń.

Kolumny „Długość pojedynczej sesji treningowej” i „Częstotliwość treningów w tygodniu” zostały połączone w jedną kolumnę, którego wartości są szacowane.

Wszystkie dane nominalne zostały przekonwertowane na numeryczne oraz wiersze z pustymi wartościami zostały usunięte.

# 3. Sposób instalacji i zależności

W pliku README.md znajdują się różne sposoby odpalenia aplikacji, w tym:

* Sklonowanie repozytorium oraz uruchomienie aplikacji lokalnie poprzez komendy w terminalu.
* Uruchomienie aplikacji z wykorzystaniem Dockera poprzez zbudowanie obrazu oraz uruchomienie kontenera.
* **ZALECANE ROZWIĄZANIE** - Pobranie i uruchomienie obrazu z Docker Hub poprzez zalogowanie się do Docker Hub, uruchomienie pobranego obrazu i wejście na odpowiednią stronę.

W pliku requirements.txt znajdują się zależności, bez których aplikacja nie będzie w stanie poprawnie działać, w tym:

* **streamlit** – odpowiedzialny za wygląd aplikacji oraz obsługi danych podanych przez użytkownika.
* **pandas** – służy do manipulacji danymi.
* **numpy** – wymagany do poprawnego działania Autogluona.
* **autogluon** – odpowiedzialny za stworzenie i wytrenowanie modelu.
* **kagglehub** – służy do uzyskania dostępu do danych ze strony Kaggle.
* **scikit-learn** – dzieli zestaw danych na zestaw treningowy i testowy.

# 4. Struktura projektu

Aplikacja została napisana z rozdzieleniem logiki działania – przygotowania zestawu danych, obsługi modelu oraz reprezentacji aplikacji ze strony front-end.

### prepare\_csv.py – plik przygotowujący zestaw danych

W tym pliku pobierane są dane ze strony Kaggle, za pomocą kagglehub, w postaci pliku CSV, po czym jest ona zamieniana na DataFrame, przy okazji zamieniając wszystkie nominalne liczby na wartości numeryczne. Następnie wyliczany jest długość czasu spędzonego w tygodniu na siłowni i usuwane są dane przeszkadzające a nauczaniu.

### model\_train.py – plik do wytrenowania modelu

Tutaj wczytywane są dane z pliku CSV, po czym są one mieszane i podzielone na dwa sety: train i test. Później model jest uczony za pomocą TabularPredictor z Autogluona i zachowywany jest tylko najlepszy z nich, który następnie jest zapisywany do pliku.

### model\_prediction.py – plik do predykcji na podstawie podanych danych

To jest funkcja, która jest pod spodem Streamlita, która ma na celu przedstawienie predyktorowi odpowiedniego rodzaju danych, które zostały przekazane z front-endu i zwróceniu wartości, którą przewidział predyktor.

### app.py – plik front-end

W tym pliku znajduje się front-end napisany w bibliotece Streamlit, który jest zrobiony w miarę prosty sposób, aby każdy mógł z niego korzystać.

# 5. Model i jakość modelu

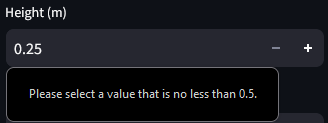
Autogluon wybrał kilka modeli, które pasowały do danego problemu, są to:

* LightGBMXT z wagą 0.733
* NeuralNetTorch z wagą 0.133
* NeuralNetFastAI z wagą 0.067
* XGBoost z wagą 0.067

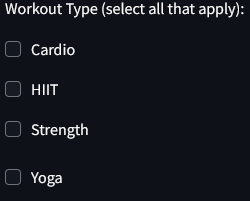
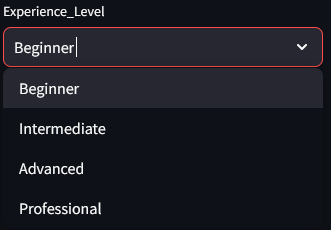
Dzięki temu, RMSE, które zostało osiągnięte to 0.776, które wydaje się wystarczająco dobre, ponieważ jest to pomyłka 45 minut na cały tydzień.

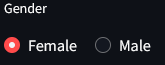
# 6. Obsługa aplikacji

Aplikacja umożliwia użytkownikowi podanie odpowiednich wartości do każdego atrybutu. Każda wartość numeryczna posiada minimalną i maksymalną wartość, której użytkownik nie może przekroczyć, w przeciwnym wypadku wyskoczy błąd.



Stopień doświadczenia, płeć oraz typ treningu można wybrać poprzez odpowiednie pole wyboru.





Gdy wszystkie pola są odpowiednio uzupełnione, użytkownik może kliknąć w przycisk „Predict”, który oszacuje liczbę godzin spędzonych na siłowni w tygodniu i wyświetli komunikat z wynikiem.

