**Dokumentacja do projektu**

Projekt dotyczy nauki wykorzystania narzędzi do przeprowadzenia analizy zbiorów danych.

Wybrany zbiór danych został pobrany z platformy Kaggle

* [Indicators of Heart Disease (2022 UPDATE)](https://www.kaggle.com/datasets/kamilpytlak/personal-key-indicators-of-heart-disease/data)

Zadanie zostało zrealizowane z wykorzystaniem VSCode, kaggle MLFlow oraz instalacji lokalnej na dysku komputera. (Argumentacją za tym był dostęp do laptopa o dobrych parametrach technicznych, procesor, pamięć RAM, Dyski SSD szybkie o dużej pojemności)

Na Github znajduje się [Pawel20240101/MLOps\_PZ](https://github.com/Pawel20240101/MLOps_PZ) w którym zawarte zostały wszystkie niezbędne informacje do odtworzenia i uruchomienia projektu.

Struktura projektu

* Data – katalog do przechowywania zbioru danych pobranego z Kaggle
* Doc – Dokumentacja projektu w postaci plików .docx oraz .pdf
* Eksploracja – Katalog w którym gromadzone są pliki .png powstałe w wyniku analizy danych wejściowych
* Models – modele klasyfikatorów pobrane lokalnie oraz plik wynikowy (najlepsze parametry ) z treningu zbioru testowego
* Notebooks – Niezbędne pliku jupyter notebook wymagane do realizacji projektu
* Reports – pliki w formacie .csv – wyniki dla poszczególnych klasyfikatorów
* Results – Graficzna prezentacja wyników – tj. confusion matrix, roc curve oraz dla modeli które obsługują SHAP – prezentacja wyniku SHAP.

Dodatkowe pliki

* settings.json – plik ze wskazaniem środowiska wykorzystywanego w projekcie. Ze względu na zdarzające się sytuacje kiedy środowiska były wybrane nieprawidłowo lub był problem z ich wskazaniem przygotowałem plik który rozwiązuje ten kłopot. Środowisko jest wskazane automatycznie po uruchomieniu projektu.
* requirements.txt – Plik z wymaganymi bibliotekami.
* README.md – ogólne informacje wyświetlające się na głównej stronie projektu w serwisie Github.
* Przydatne\_polecenia.txt – Plik z poleceniami niezbędnymi do realizacji projektu

——————————————————————————————————————————————

Projekt składa się z trzech plików znajdujących się w katalogu notebooks.

Odtworzenie projektu

1. **Pobierz projekt z Github**

W terminalu VScode wpisz: **git clone https://github.com/Pawel20240101/MLOps\_PZ**

1. **W projekcie na Twoim dysku utwórz odpowiednie środowisko**

Zajrzyj do pliku **Przydatne\_polecenia.txt**

1. **Aktywuj swoje środowisko**

Zajrzyj do pliku **Przydatne\_polecenia.txt**

1. **Zainstaluj biblioteki z pliku requirements.txt**

W terminalu VScode wpisz: **pip install -r requirements.txt**

1. **Uruchom serwer MLflow.**

W terminalu VScode wpisz: **MLflow UI**

1. **Z katalogu notebooks uruchom po kolei pliki** 
   1. 1\_Import\_i\_eksploracja\_danych\_ML.ipynb
   2. 2\_Przygotowanie\_danych\_ML.ipynb
   3. 3\_ML\_Workflow\_pipeline.ipynb
2. **Sprawdź wynik w MLflow**

Otwórz przeglądarkę i wpisz: **http://localhost:5000**

Opis plików .ipynb wykorzystanych w projekcie

1\_Import\_i\_eksploracja\_danych\_ML.ipynb – plik służy do pobrania danych z Kaggle oraz wykonuje eksplorację danych ze zbioru .csv

2\_Przygotowanie\_danych\_ML.ipynb – plik służy do wstępnego przygotowania danych do procesu kalsyfikacji

3\_ML\_Workflow\_pipeline.ipynb – plik służy do realizacji głównego zadania czyli trenowania modeli i zapisywania wyników na serwerze MLflow – Proces jest wykonywany całkowicie automatycznie. I składa się z następujących etapów

1: Instalacja i import niezbędnych bibliotek

2: Wczytywanie danych przygotowanych wcześniej

3: Przygotowanie danych – identyfikacja kolumn

4: Przygotowanie pipeline do przetwarzania danych

5: Trenowanie modeli i logowanie wyników do MLflow

6: Porównanie wyników modeli

7: Tuning hiperparametrów najlepszego modelu (Optuna)

8: Zapis najlepszego modelu do pliku

9: Predict pipeline – generowanie predykcji

10: Podsumowanie projektu

11: Zakończenie

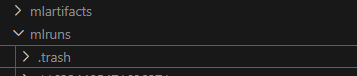
Oczywiście edytując plik można zmieniać parametry klasyfikatorów nazwy eksperymentów etc.

W punkcie 1 możemy zmieniać nazwy eksperymentu

W punkcie 5 umieszczone są klasyfikatory – możemy je wyłączać z uczenie wstawiając znak # przed klasyfikator lub dodać nowe – w takim przypadku trzeba zmodyfikować kod znajdujący się w dalszej części notebooka.

Uwaga.

Czasami w trakcie automatycznego tworzenia katalogu mlruns w środku niego nie jest tworzony katalog .trash. W takim przypadku należy ręcznie stworzyć taki katalog.



Wizualizacja wyników z wykorzystaniem MLflow (przykłady)

