**Obraz zawierający tekst, Czcionka, logo, wizytówka

Opis wygenerowany automatycznie**

**Mateusz Bocak i Gabriela Bieniek**

**Nr albumu studentów: 125104, 125101**

**Algorytmy klasyfikacyjne**

**Praca projektowa nr 2 z IMEBD**

Prowadzący: dr inż. Piotr Grochowalski

Rzeszów 2024

Spis treści

[Opis projektu 3](#_Toc186025376)

[Funkcje 3](#_Toc186025377)

[Użyte technologie 3](#_Toc186025378)

[Struktura projektu 4](#_Toc186025379)

[Zbiór danych 5](#_Toc186025380)

[Instalacja 5](#_Toc186025381)

[Obsługa aplikacji 6](#_Toc186025382)

[Repozytorium projektu](https://github.com/JustFiesta/Datasets-Classicifation.git)

## Opis projektu

Projekt implementuje aplikację opartą na Streamlit do klasyfikowania:

* wiadomości e-mail jako spam lub nie-spam
* grzyby jako trujące lub nie

Porównuje wydajność pięciu algorytmów klasyfikacji przy użyciu zestawu danych: [Enron Spam Dataset](https://github.com/MWiechmann/enron_spam_data.git) i [UCI Mushroom](https://archive.ics.uci.edu/dataset/73/mushroom) (dataset numer 73).

## Funkcje

1. Przyjazny interfejs:

* Interaktywny UI oparty na Streamlit

1. Metody klasyfikacji:

* Drzewo decyzyjne
* Naiwny Bayes
* K-najbliższych sąsiadów (KNN)
* Maszyna wektorów nośnych (SVM)
* Sieć neuronowa

1. Test porównawczy

* Porównuje metryki klasyfikacji w różnych modelach (dokładność, precyzja, odwołanie, F1-score)
* Wizualizacja macierzy pomyłek i wykresów wydajności

1. Obsługa danych

* Preprocesing: Łączy treść tematu i treści w jednym polu tekstowym
* Obsługuje brakujące i nieprawidłowe dane

1. Rozszerzalność

* Zaprojektowany w celu łatwego włączenia nowych metod klasyfikacji

## Użyte technologie

* Python: język programowania
* Streamlit: Interaktywny framework internetowy
* Scikit-learn: Biblioteka uczenia maszynowego
* Matplotlib & Seaborn: Narzędzia wizualizacji
* Pandas: Manipulacja danych

## Struktura projektu

1. Główna aplikacja: streamlit\_app.py

Obsługuje:

* Ładowanie i przetwarzanie zbioru danych
* Konfigurację interfejsu Streamlit
* Benchmarking oraz klasyfikację
* Wizualizację wyników

1. Implementacja algorytmów: classifier\_alghoritms.py

Zawiera implementację następujących algorytmów:

* Drzewo decyzyjne: wykorzystuje sklearn.tree.DecisionTreeClassifier
* Naive Bayes: wykorzystuje sklearn.naive\_bayes.MultinominalNB
* KNN: klasyfikacja oparta na najbliższych sąsiadach
* SVM: konfigurowalny dla różnych jąder (linear, rbf)
* Sieć neuronowa: klasyfikator oparty na perceptronie wielowarstwowym (MLP)

1. Przetwarzanie danych: *processing\_mushrooms.py* i *processing\_spam.py*

Funkcje:

* Ładowanie i oczyszczanie zbioru danych
* Łączenie tematu i treści wiadomości w jedno pole
* Kodowanie etykiet spam/nie-spam na wartości binarne (1 dla spamu, 0 dla nie-spamu)

1. Skrypt uruchamiający: run.py

Uruchamia aplikacje z głównego folderu projektu.

Automatyzuje:

* Instalację zależności z pliku requirements.txt
* Uruchamianie aplikacji Streamlit

1. Własnoręczna implementacja jednego algorytmu: manual\_knn.py

Implementacja KNN wykonana ręcznie.

## Zbiory danych

Projekt korzysta z dwóch zbiorów:

* Enron Spam Dataset, dostępnego pod adresem [Enron Spam Dataset](https://github.com/MWiechmann/enron_spam_data.git).
* UC Irvine Machine Learning Repository, zbiór nr 73 [UCI Mushroom](https://archive.ics.uci.edu/dataset/73/mushroom).

Przetwarzanie danych dla Datasetu Spam:

* Łączy kolumny Subject oraz Message w jedno pole: Combined\_Message
* Usuwa wiersze z brakującymi lub niepoprawnymi wartościami w kolumnie Spam/Ham
* Koduje etykiety Spam/Ham na wartości binarne (Spam)

Przetwarzanie danych dla Datasetu UCI Mushroom:

* Pobranie danych z repozytorium UCI.
* Konwersja danych na DataFrame z pandas.
* Połączenie danych cech i etykiet w jeden DataFrame.
* Uzupełnienie brakujących wartości metodą ffill.
* Enkodowanie kolumn typu object na wartości liczbowe za pomocą LabelEncoder.
* Zwrócenie przetworzonych danych jako DataFrame.

## Instalacja

Wymagania:

* Python 3.8 lub nowszy

Kroki instalacji:

1. Klonowanie repozytorium

>> git clone https://github.com/JustFiesta/Datasets-Classicifation.git

>> cd Datasets-Classification

1. Wypakowanie pliku zip ze zbiorem danych
2. Setup środowiska Python (w przypadku pierwszego użycia)

>>\* python3 -m ensurepip            # ensure pip is installed  
>> python -m venv ./.venv             # create venv  
>> source .venv/Scripts/activate     # activate virual environment (Linux)  
>> .venv/Scripts/activate             # activate virual environment (Windows)  
>> pip install -r .\reqiurements.txt # install dependencies

1. Uruchomienie aplikacji

>> python run.py

## Obsługa aplikacji

1. Uruchomienie aplikacji

Aplikacja Streamlit otworzy się w przeglądarce po wykonaniu run.py

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

1. Wybór funkcji
   * Indywidualna klasyfikacja: uruchamia i wizualizuje wyniki dla wybranej metody

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

* + Benchmark: porównuje wszystkie klasyfikatory jednocześnie

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

* + About: wyświetla informacje o projekcie

A screenshot of a computer

Description automatically generated

1. Wyniki

Obliczane metryki dla każdego klasyfikatora:

* Dokładność (accuracy): Ogólna poprawność predykcji
* Precyzja (precision): Ile przewidzianych spamów faktycznie było spamem
* Recall: Ile rzeczywistych spamów zostało poprawnie wykrytych
* F1-score: Średnia harmoniczna precyzji i recall