**Projekt zaliczeniowy ZUM**

**Opis:** Wybrać zadanie z zakresu Natural Language Processing, Computer Vision albo Audio i stworzyć system z wykorzystaniem technologii prezentowanych na zajęciach. Końcowy projekt powinien być zaprezentowany na ostatnich zajęciach w semestrze.

**Proponowana struktura projektu:**

1. Znalezienie datasetu do wybranego zadania (proponowane źródła: HuggingFace, PapersWithCode)
2. Zebranie i przygotowanie danych:
   1. Zebrać dane (najlepiej wybrać zbiór z HF), oczyścić i przetworzyć w zależności od wybranego zadania
   2. Preprocessing danych w zależności od dziedziny i zadania (nie wszystkie wypisane będą potrzebne!):
      1. NLP: tokenizacja, stop słowa, lemanizacja/stemming, outliery, duplikaty etc.
      2. CV: zmiana rozmiaru, normalizacja, augmentacja (obracanie obrazu, przycinanie, etc.)
      3. Audio: standaryzacja częstotliwości próbkowania, usunięcie ciszy, normalizacja poziomu głośności, wyciągnięcie cech
3. Inżynieria cech:
   1. Wyciągnąć najważniejsze cechy do modeli ML
   2. Potencjalne procesy w zależności od zadania:
      1. NLP: reprezentacja tekstowa (Tf-Idf, word embeddings)
      2. CV: wykrywanie krawędzi, histogramy kolorów, składowa główna
      3. Audio: MFCC, Chroma, Spektogram
   3. Warto rozważyć redukcję wymiarów
4. Klasyczne ML:
   1. Wybrać i zbudować model z zastosowaniem klasycznego ML jako baseline
   2. Pamiętać o etapie: treningu, walidacji i testowania
   3. Ewaluacja modelu w oparciu o wybrane metryki
   4. Zachować wyniki do porównania z modelami z pkt. 5
5. Model oparty o architekturę Transformera:
   1. Wybrać model oparty o transformer w zależności od zadania
   2. Proponowane architektury (warto poszukać innych na HF):
      1. NLP: BERT, RoBERTa, DistiBERT
      2. CV: Vision Transformer (ViT), Swin Transformer
      3. Audio: Wav2Vec2, HuBERT
   3. Wykonać finetuning modelu do wybranego zadania
   4. Można pomyśleć nad augmentacją danych czy transfer learningiem
   5. Ocenić wyniki modelu i porównać z modelem bazowym
6. Ewaluacja:
   1. Porównać modele i wybrać najlepszy
   2. Zastosować metryki ewaluacji do modeli klasycznego ML i opartych o Transformer
   3. Dodać wizualizacje (np. macierz pomyłek, ROC curve) dla porównania modeli
   4. Przeanalizować błędy i zidentyfikować potencjalne obszary do poprawy
   5. Podsumować wyniki, które zdecydowały o wyborze modelu
7. Tablica interaktywna:
   1. Wdrożyć wybrany model i utworzyć tablicę interaktywną do nadzoru
   2. Stworzyć tablicę w Streamlit albo Plotly Dash
   3. Tablica ma umożliwić:
      1. przetestowania modelu (predykcje) na danych testowych i nowych danych
      2. wizualizację wyników, metryki ewaluacji, analizę błędów

**Wymagania:**

* Projekt napisany w Pythonie
* Proponowane narzędzia i biblioteki:
  + Dane: numpy, pandas
  + Obróbka danych:
    - NLP: spacy, nltk
    - CV: OpenCV, Pillow
    - Audio: librosa, pydub
  + Klasyczne uczenie maszynowe: scikit-learn, xgboost
  + Uczenie głębokie: Tensorflow, PyTorch
  + Transformer: HuggingFace (transformers, datasets etc.)
  + Ewaluacja: matplotlib, plotly, seaborn, scikit-learn
  + Dashboard: Streamlit, Plotly Dash

**Forma oddania projektu:**

1. Repo na githubie
   1. Udostępnione dla [dwnuk@pjwstk.edu.pl](mailto:dwnuk@pjwstk.edu.pl) w przypadku repo prywatnego (collaborator)
2. Skrypty w notatniku Jupyter albo pliki .py
3. Dokumentacja
   1. W formie readme na githubie:
      1. Opis wybranego zadania i celu projektu
      2. Opis danych (pochodzenie, liczba etc.)
      3. Etapy projektu
      4. Krótkie podsumowanie wyników i wnioski – co sprawdziło się najlepiej, co można by zrobić inaczej (to sugestie, proszę pomyśleć nad tym w grupie☺)
4. Dashboard do zaprezentowania na ostatnich zajęciach
   1. Interaktywna tablica pokazująca działanie wytrenowanego modelu, nadzór (np. learning curve) oraz wyniki

**Przykładową strukturę projektu proszę potraktować jako propozycję. Zamiast klasycznego ML można jako model bazowy stworzyć sieć neuronową.**