Projekt: Rozpoznawanie płci osoby mówiącej - opis wybranej koncepcji

1. Przygotowanie danych - main.py

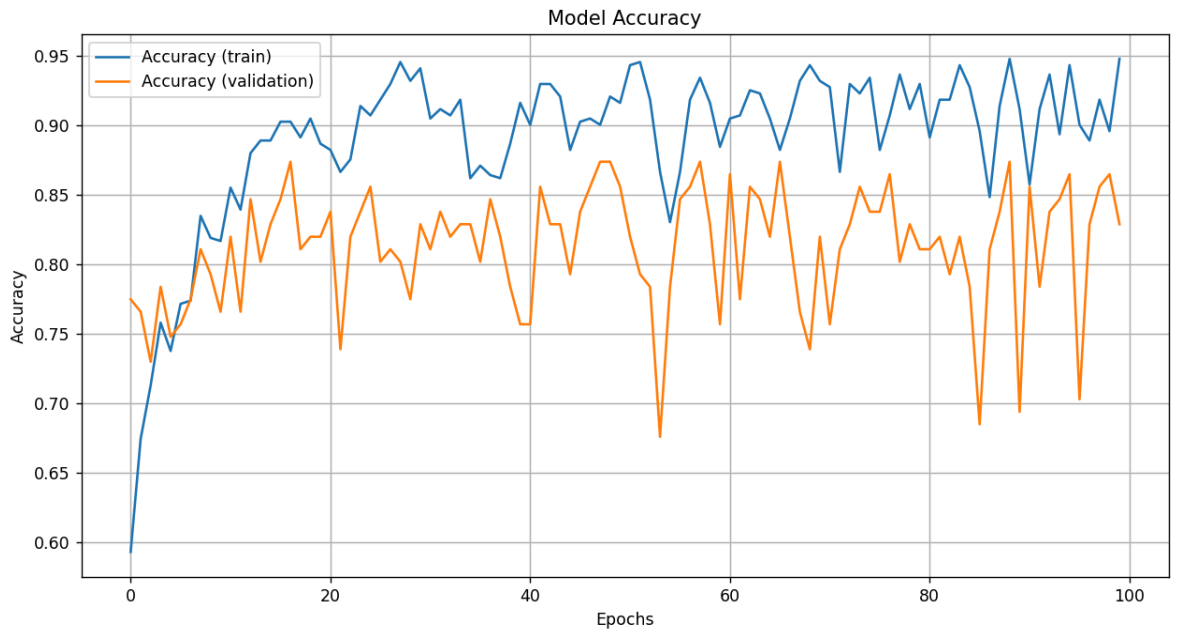
Do naszych testów korzystamy z zebranych 692 próbek dźwiękowych. Poddajemy je transformacie Fouriera, z której w razie potrzeby możemy wyświetlić wykresy. Następnie wykonujemy normalizację i zapis do plików .csv.
Częstotliwość w plikach .csv sięga powyżej 10000 Hz jednak dane te są niepotrzebne - więc skracamy ten plik, tak by obejmował częstotliwości tylko z interesującego nas zakresu - 0 Hz - 10000 Hz.
Na wyjściu algorytmu określamy stan - 0(kobieta) albo 1(mężczyzna). Na podstawie porównania stanu z predykcji oraz stanu rzeczywistego określamy i wyświetlamy dokładność metody.

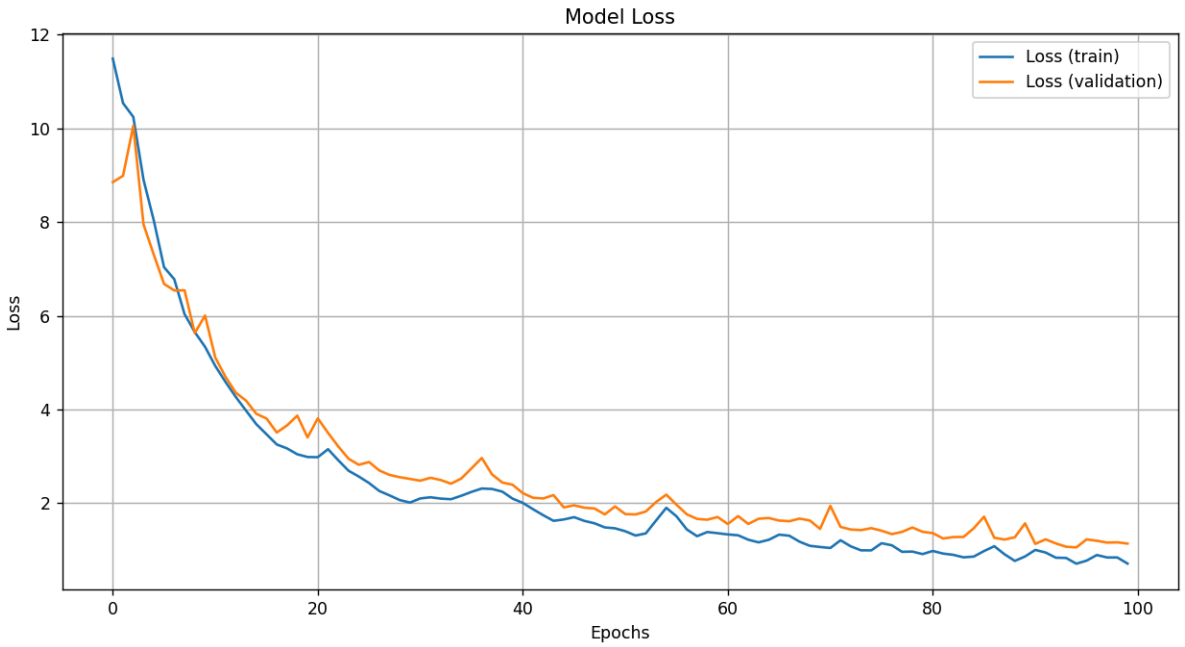
2. Koncepcje rozwiązania problemu - funkcje SVM, MLP, Autogluon

W naszym projekcie porównujemy wyniki z kilku koncepcji rozwiązania problemu.

Jako pierwsze skupiliśmy się na koncepcji związanej z SVM - skorzystaliśmy z modelu SVC: model = SVC(kernel = 'linear', C = 0.2). Wykonaliśmy walidację krzyżową. Następnie dla porównania podzieliliśmy zbiory na dane treningowe(80%) i testowe(20%). Zauważyliśmy, że oba rezultaty mają niską dokładność (70%).

W kolejnym podejściu zbadaliśmy koncepcję MLP - podzieliliśmy zbiory na dane treningowe(80%) i testowe(20%) i użyliśmy modelu Sequential. Użyliśmy sieci neuronowej typu feedforward, która wykorzystuje funkcję aktywacji softmax, z trzema ukrytymi warstwami w pełni połączonymi z funkcjami aktywacji LeakyReLU oraz warstwami Dropout . Model został skompilowany z optymalizatorem adam, funkcją straty categorical_crossentropy do wieloklasowej klasyfikacji oraz metryką accuracy do oceny dokładności. (Dokładność na poziomie 75-85%). Poniżej przykładowe wykresy dokładności i błędu:





Dodatkowo skorzystaliśmy z biblioteki Autogluon, która korzysta z technik automatycznego doboru modeli uczenia maszynowego. Używamy narzędzia TabularPredictor, który dobiera algorytmy, hiperparametry i strategie przetwarzania danych w celu uzyskania najlepszej wydajności na podanym zbiorze danych. Rezultaty w tej metodzie osiągały około 85-97% dokładności.

3. Przetwornik pliku dźwiękowego na kategorię - mężczyzna/kobieta. - super_model_ka.py
Na wejście modelu podajemy plik dźwiękowy od użytkownika. Następnie plik ten poddawany jest transformacie Fouriera. W kolejnym kroku plik ten analizowany jest przez najlepszy model - Autogluon. Na wyjściu w zależności od wyniku predykcji otrzymujemy informację: kobieta lub mężczyzna.
Zadanie to wykonujemy na 20 własnych plikach, a następnie z wyników na tym zbiorze wyliczamy dokładność.

4. Problemy projektowe
   a) Bardzo małe doświadczenie i wiedza.
   b) Zaszumione pliki dźwiękowe - Z początkowych zebranych 99 próbek usunęliśmy 16, które wydały nam się bezużyteczne. Poskutkowało to jednak pogorszeniem dokładności przewidywań w każdym modelu. - 7-krotne zwiększenie liczby próbek ustabilizowało poprawność wyników.
   c) Część plików .wav była dwukanałowa - co skutkowało dwuwymiarowym zapisem w pliku .csv. Udało nam się to rozwiązać wykorzystując tylko jeden kanał.
   d) Mała liczba próbek w porównaniu z ilością cech.
   e) Przy 700 próbkach uczenie modeli zajmuje bardzo dużo czasu, przez co rozwijanie projektu staje się trudne.
   f) Przy ustawieniu zmiennej target_length (odpowiedzialnej za ustawienie zakresu częstotliwości) na 15000 lub więcej, uczenie modeli w Autogloun.py jest utrudnione (a nawet pomijane), ponieważ przekraczana jest dostępna pamięć.
   g) Sprawdzanie poprawności modelu na własnych (całkiem nowych) plikach trwa krócej w przypadku korzystania z zapisanego wytrenowanego modelu.

h) Wymogiem używania modelu jest wgrywanie plików o nazwie “kobiecy…” lub “meski…” z rozszerzeniem .mp3 lub .wav.
i) Możliwym dopracowaniem projektu może być pogrupowanie częstotliwości w histogram np. co 10 Hz. Jednak z racji zadowalającej dokładności oraz zbliżającego się deadlinu przełożyliśmy wykonanie tego pomysłu (np. na pracę inżynierską :))
j) Problem ze wstawieniem plików na GitHub ze względu na limity wielkości przesyłanych plików.
k) Nieodpowiednio posortowane pliki po wczytaniu za pomocą glob.glob -> wczytywały się w kolejności 0,1,10,101 itd, zamiast 0,1,2,3,4. Sprawiło to, że następnie przydzielone labels były często błędne. Naprawiliśmy ten błąd korzystając z sortowania.

5. Wnioski
1. Model znacznie częściej myli się w prawidłowym zaklasyfikowaniu głosu męskiego niż kobiecego.
2. Dokładność modeli na zbiorze testowym z bazy wynosi:
    a) SVM - około 70%,
    b) MLP - 75-85%,
    c) Autogluon - 85-97%,

    dlatego skupiliśmy się na Autogluonie.
3. Dokładność na 20 własnych plikach dźwiękowych na podstawie wytrenowanego Autogluona wynosi 60-70%.

Netografia:
[1] Kod w pliku Fourier.py rozwijaliśmy na podstawie kodu ze strony
https://stackoverflow.com/questions/47982785/python-performing-fft-on-music-file
[2] Wiedzę o podstawach ML pozyskiwaliśmy z:
https://www.youtube.com/playlist?list=PLCC34OHNcOtpcgR9LEYSdi9r7XIbpkpK1
https://www.youtube.com/playlist?list=PLCC34OHNcOtqSz7Ke7kaYRf9CfviJgO55
[3] Autogluon
https://nbviewer.org/github/Innixma/autogluon-doc-utils/blob/main/docs/cheatsheets/stable/autogluon-cheat-sheet.pdf
[4] Nagrania głosu
https://commonvoice.mozilla.org/en/datasets
[5] Konwertowanie plików na mp3
https://convertio.co/pl/download/654c954a257c2e6fdcf33c70f69e920332f300/