Autorzy:

Tymoteusz Macewicz

Jakub Gucik

Projekt końcowy – Podstawy Uczenia Maszynowego

Badanie modeli i optymalizacji metod klasyfikacji

1. Wstęp:

1. Problem badawczy:

Postawionym problemem badawczym jest klasyfikacja plików audio, zawierających różne instrumenty, a także zbadanie oraz optymalizacja metod klasyfikacji. Celem jest poprawne rozpoznanie i sklasyfikowanie nagrania i głównego instrumentu przez wybrane modele oraz analiza ich działania, optymalizacji, a także przyczyn otrzymanych wyników.

2. Baza danych:

Pierwszą bazą danych, która została wybrana był to zestaw ze strony Kaggle – ‘Drum Kit Sound Samples’: https://www.kaggle.com/datasets/anubhavchhabra/drum-kit-sound-samples. Jest to baza zawierająca dźwięki perkusyjne z kategorii: kick, snare, tom i overhead. W bazie znajduje się 160 plików formatu .wav, po 40 z każdego rodzaju, a nagrania zostały pozyskane poprzez nagrania ‘live’ lub komputerowo. Jednak przez wzgląd na mały rozmiar bazy i osiągane niemalże maksymalne skuteczności na podstawowych modelach zdecydowano się na zmianę bazy danych. Ta jednak bardzo dobrze posłużyła do szybkiej weryfikacji i znalezienia błędów w kodzie.

Finalnie wybrano bazę – ‘IRMAS’, którą można znaleźć na stronie: https://www.upf.edu/web/mtg/irmas. Jest to zbiór danych w postaci plików .wav, przeznaczony do rozpoznawania instrumentów w sygnałach muzycznych audio – zawiera wiele sampli z oznaczonymi dominującymi instrumentami, czyli label’ami niezbędnymi do uczenia nadzorowanego. Jest więc to baza przeznaczona do zastosowań w automatycznym rozpoznawaniu i klasyfikacji. Zestaw danych został przygotowany przez Uniwersytet Pompeu Fabry w Barcelonie.

Same pliki zawierają adnotacje dotyczące głównego dominującego instrumentu grającego, zgodnie z oznaczeniami:

- cel – wiolonczela [ang. Cello]
- cla – klarnet [ang. Clarinet]
- flu – flet [ang. Flute]
- gac – gitara akustyczna [ang. Acoustic Guitar]
- gel – gitara elektryczna [ang. Electric Guitar]
- org – organy [ang. Organ]
- sax – saksofon [ang. Saxophone]
- tru – trąbka [ang. Trumpet]
- vio – skrzypce [ang. Violin]
- voi – wokale [ang. Voice]

Dodatkowo, część z plików oferuje dodatkowe opisy dotyczące gatunku i występowanie bębnów:

- dru – występowanie bębnów
- nod – brak bębnów
- cou_fol – country-folk
- cla – klasyczna
- pop-roc – pop-rock
- lat-sou – latin-soul

Nagrania to części muzyczne z aktualnych utworów z ubiegłego wieku. Co za tym idzie oferuje dużą różnorodność, a także jakość audio – utrudnia to zadanie klasyfikacji.

W kwestii plików – baza składa się z 6705 nagrań audio składających się na 11 klas w różnej liczebności. Są to pliki 16 bit stereo .wav o częstotliwości próbkowania 44100 Hz.

W początkowej fazie, ze względu na rozmiar bazy zawężono i wyrównano ilość plików .wav do 150 elementów (pierwszorzędnie usuwane były nagrania z oznaczeniami ‘dru’ i ‘nod’) na klasę, a co za tym idzie sumarycznie analizowana baza posiada 1650 elementów.

Wyselekcjonowana baza danych znajduje się na stronie pod adresem:

https://drive.google.com/drive/folders/1MegaYzPFYEbVGd5ica_SU_S_xoqSbcGF?usp=sharing

2. Przygotowanie i weryfikacja danych, preprocessing:

Przygotowanie danych polegało na ręcznym przeszukaniu nagrań i selekcji 150 elementów na zbiór i klasę. Zadanie było ułatwione, gdyż baza danych ma odpowiednio posortowane nagrania. Przygotowano odpowiednie biblioteki do użycia – skorzystano z paczek:

- scipy,
- os,
- librosa,
- optuna,
- pandas,
- numpy,
- collections,
- sklearn,
- pathlib.

Do analizy zdecydowano użyć się samego MFCC, a także związane z nim delty i parametry. Przygotowano również odpowiednie zestawienie folderów w kodzie, w celu łatwiejszego wyszukiwania nagrań oraz stworzono bazę danych z pomocą biblioteki pandas i DataFrame. Znajdują się w niej informacje, które zawierają:

- odczytane dane (nagranie),
- częstotliwość próbowania,
- typ instrumentu (zgodnie ze skrótami we wstępie),
- gatunek muzyczny,
- MFCC,

- MFCC-delta,
- MFCC-delta-delta,
- parametry wyliczone z MFCC:
    - wartość średnia,
    - odchylenie standardowe,
    - mediana,
    - I i III kwartyl,
    - rozrzut pomiędzy 10 i 90 percentylem,
    - kurtozę,
    - skośność,
    - wartość minimalną,
    - wartość maksymalną.

Do rozdziału danych użyto ‘train_test_split’ z biblioteki sklearn, a także ustandaryzowano dane za pomocą Standard Scaler’a.

3. Klasyfikacja:

W dalszej części raportu projektowego odpowiednie analizy i omówienie klasyfikatorów zostało odpowiednio przygotowane przez odpowiedzialne za dane modele osoby. Jednak całość, pomimo pracy indywidualnej, opierała się na konsultacjach rozwiązań i wyników, a także wprowadzonych zmian. Część wyboru i podsumowania została zakończona wspólnie.

Do zagadnienia klasyfikacji wybrano dwa modele:

1. SVM – wektory nośne – Tymoteusz Macewicz
2. ExtraTreesClassifier – Jakub Gucik

4. Klasyfikacja – omówienie modeli i optymalizacja parametrów:

1. SVM – wektory nośne:

SVM (Support Vector Machine) to nadzorowany algorytm uczenia maszynowego, służy do klasyfikacji obiektów za pomocą wyznaczania granic decyzyjnych w postaci hiperpowierzchni.

Optymalizowane parametry to:

1. „C Parameter” - maksymalna liczba błędnych klasyfikowań
2. „Kernel” - rodzaj jądra użytego w algorytmie
3. „Degree” - stopień funkcji jądra wielomianowego (Kernel musi być ‘poly’, inaczej ignorowany przez inne jądra)
4. „Gamma” - współczynnik jądra dla ‘rbf’, ‘poly’ i ‘sigmoid’
5. „Coef0” - niezależny parametr w funkcji jądra dla ‘poly’ i ‘sigmoid’
6. „Shrinking” - może zmniejszyć czas trenowania przy dużej liczbie iteracj

2. ExtraTreesClassifier:

Model jest algorytmem podobnym do klasycznego lasu losowego, który różni się kryterium podziału drzewa w danym węźle. Mamy więc podział losowy, a nie optymalny jak w lasach losowych, co powinno skutkować szybszym przetworzeniem danych, a co za tym idzie szybszym uczeniem nadzorowanym. Dodatkowo w ExtraTrees nie występuje losowanie ze zwracaniem, a co za tym idzie obserwacje podane do kolejnych drzew nie będą się powtarzać.

Parametrami, które zgłębiono i poddano pod optymalizację to:

1. „n_estimators” – liczba drzew w algorytmie
2. „max_depth” – maksymalna głębokość drzewa, podziału gałęzi
3. „min_samples_split” – minimalna liczba próbek potrzebnych do podziału węzła
4. „n_jobs" – liczba procesów do jednoczesnego wykonania („-1” – używa wszystkich procesorów)
5. „criterion” – kryterium pomiaru jakości podziału w drzewie. Do dyspozycji:
    a. Gini – model „Gini Impurity” – pomiar prawdopodobieństwa niepoprawnej klasyfikacji
    b. Log_loss – funkcja kosztu zdefiniowana ujemnym „log-likelihood’em” – dokładny opis i wzór dostępny w bibliotece sklearn
    c. Entropy

6. „max_features” – liczba cech do uwzględnienia w poszukiwaniu najlepszego splitu

7. „random_state” – kontroluje trzy źródła losowości opisane w dokumentacji

8. „warm_start” – użycie poprzedniego rozwiązania do sfitowania i dodania większej ilości estymatorów (nie tworzy lasu „od zera”)

5. Analiza wyników:

1. SVM – wektory nośne:

Wstępnie przeprowadzona została klasyfikacja na domyślnych wartościach parametrów modelu. Dane treningowe i testowe opierały się na wyliczonych współczynnikach MFCC sygnałów. Po wytrenowaniu modelu i przeprowadzeniu predykcji wyliczone zostały metryki oceniające model. Metryki miały następujące wartości:

Accuracy = 42.95%

F1 Score = 42.39%

Precision = 46.37%

Recall = 42.13%

Wstępne wyniki nie są zadowalające. Następnie przeprowadzono optymalizację parametrów w celu uzyskania lepszych wyników. Ocenianą metryką podczas optymalizacji było F1 Score. Szukano najlepszych wartości dla następujących parametrów: C, kernel, degree, gamma, coef0, shrinking przy 50 próbach. Uzyskane wyniki to:

Accuracy = 44.55%

F1 Score = 43.99%

Precision = 44.70%

Recall = 44.03%

Nie udało się uzyskać znaczącej poprawy w badanych metrykach (Wzrost F1 o 1.6 punktu procentowego). Następnie przeprowadzono próby optymalizacji dla innych zakresów parametrów. Przy żadnej próbie nie udało uzyskać się lepszych wyników, pomimo testowania licznych kombinacji.

Przeprowadzona została również próba optymalizacji parametrów modelu uczącego się na danych testowych opierających się na parametrach MFCC (wymienionych w przygotowaniu danych). Najwyższy uzyskany wynik F1 to 18%.

2. ExtraTreesClassifier:

Na początek prób i analizy wykonano test na podstawowym modelu z domyślnymi wartościami, a więc przeprowadzono trening oraz predykcję i wyliczono metryki. Należy zaznaczyć, iż testy zostały przeprowadzone na parametrach wyliczonych z MFCC. Macierz pomyłek zostanie pominięta w raporcie przez wzgląd na brak większych wybijających się obserwacji (jedynie pomyłki klasy 0 z 7 - wartość największa 5). Omówione zostaną natomiast metryki pod koniec tego punktu. W tej próbie uzyskano wyniki zestawione poniżej:

- Accuracy = 65.15%
- F1 Score = 64.58%
- Precision = 66.38%
- Recall = 65.15%

Wyniki jak na model niemodyfikowany i nie optymalizowany są zadowalające.

W kolejnej próbie dokonano już optymalizacji i w tej iteracji posłużono się parametrami: n_estimators, max_depth, min_samples_split i n_jobs. Wykorzystano 25 triali i otrzymano następujące wyniki:

- Accuracy = 58.78%
- F1 Score = 58.05%
- Precision = 61.62%
- Recall = 58.78%

Przeciw oczekiwaniom otrzymano jednak wyniki słabsze od przewidywanych, czyli niższe od podstawowego modelu. Przeanalizowano, więc powyższe triale i wyciągnięto wnioski:

- algorytm dobrze radzi sobie przy wysokiej liczbie n_estimators i max_depth – zakresy zwiększono,
- algorytm lepiej radzi sobie przy małej liczbie min_samples_split – zakres zawężono.

Przeprowadzono kolejną optymalizację, jednak w tym wypadku wraz ze zwiększeniem skomplikowania modelu, wydłużyły się czasy wykonania kolejnych triali. Całkowita optymalizacja trwała o 10 minut dłużej. Jednak zmiany przyniosły oczekiwane skutki - metryki wyniosły odpowiednio:

- Accuracy = 67.27%
- F1 Score =66.44%
- Precision = 69.95%

- Recall = 67.27%

Poczynione obserwacje pozytywnie wpłynęły na optymalizację i poprawiły wyniki wobec poprzedniej próby, jak i podstawowego modelu.

W związku z powyższym, by pchnąć wartości metryk jeszcze wyżej, zdecydowano się zbadać i przetestować pozostałe kryteria i parametry – dodano więc do analizy parametry – criterion, max_features, random_state i warm_start, a wyeliminowano ograniczenie max_depth (zgodnie z podejrzeniem, że brak limitu pomoże modelowi). Ponownie całość optymalizacji wydłużyła się o kolejne 10 minut, przez dodanie kolejnych parametrów oraz warunków. Był to jednak udany wysiłek, gdyż otrzymane wartości metryk, były najwyższe do tej pory, kolejno:

- Accuracy = 68.78%
- F1 Score = 68.40%
- Precision = 72.26%
- Recall = 68.78%

Kolejny raz przewidywania, wspomożone obserwacjami przyniosły skutki. Dodatkowo zaobserwowano, że dzięki ustawieniu warm_start = True – wartości dla F1 (który był naszą metryką optymalizującą) – nie wahały się w całym spektrum 0-70% na danych treningowych, a trzymały się w granicy 70 % - 5%. Z drugiej strony często jednak dochodziło do kolizji parametrów i problemów z tym ustawieniem, co powodowało zakończenie się niektórych trail'i niepowodzeniami.

Zdecydowano się również na zweryfikowanie hipotezy czy zmiana metryki optymalizacyjnej oraz "splitów" w crosswalidacji wpłynie na wyniki. Warto wspomnieć, iż ilość "splitów" w algorytmie została ustawiona na 11 ze względu na ilość klas. Jest to wybór podyktowany myślą, aby zadbać o odpowiedni podział klas i poprawienie uczenia modelu.

W przypadku zmiany "score'ra" na "accuracy" wartości spadły w dół:

- Accuracy = 62.42%
- F1 Score = 61.74%
- Precision = 65.33%
- Recall = 62.42%

Możemy, więc z łatwością zauważyć, iż w tym wypadku i tym modelu - F1 score jest dobrym wyborem.

W przypadku zmniejszenia "foldów" otrzymano nieznacznie lepsze wyniki optymalizacji, odpowiednio:

- Accuracy = 69.09%

- F1 Score = 68.65%
- Precision = 72.28%
- Recall = 69.09%

Co warto zaznaczyć, jest to model, który na zbiorze treningowym osiągnął najlepszy wynik - F1 score = 67.11%.

Kończąc - zbierając wszystkie obserwacje i dodatkowo wyciągając z piątej optymalizacji wniosek, iż model dobrze sprawuje się na - criterion = entropy oraz - max_features = log2/sqrt, a także łącząc to z poprzednimi iteracjami - ustawiono n_estimators na zakres 2-2500 i scoring = F1_macro. W tej optymalizacji osiągnięto kolejny mały sukces, gdyż na zbiorze treningowym osiągnięto maksimum równe - F1_score = 70.02%. Jednak na zbiorze testowym osiągnięto:

- Accuracy = 67.87%
- F1 Score = 67.19%
- Precision = 70.00%

Recall = 67.87%

Warto również tu zauważyć, iż przy odpowiednim manipulowaniu parametrami udało się w tej próbie uzyskać “efekt” warm_start, bez jego użycia - tj. wszystkie triale uzyskiwały wyniki zbliżone do 70%.

Podsumowując, otrzymano dwa najlepsze wyniki - na zbiorze treningowym lekko ponad 70%, a na zbiorze testowym 68.65% (w przypadku F1_score). Wartym zaobserwowania jest fakt, iż wyższa skuteczność na zbiorze treningowym nie gwarantuje wyższej skuteczności na zbiorze testowym. Same badania pokazały również, że znajomość parametrów, modelu i uważne obserwacje pomagają w dobrym dobrze narzędzi i algorytmów do zadań. Finalnie osiągnięto dzięki optymalizacji wzrost o około + 5% wszystkich parametrów (+ 10% względem pierwszej optymalizacji).

3. Wybór lepszego klasyfikatora:

Dla wyznaczonej bazy danych lepszym modelem do klasyfikacji obiektów okazał się ExtraTreesClassifier. Wyniki badanych metryk były znacząco wyższe dla zoptymalizowanego modelu ExtraTreesClassifier niż dla zoptymalizowanego modelu SVM (najwyższy wynik F1 na zbiorze testowym dla ExtraTreesClassifier wyniósł 68.65%, a dla SVM wyniósł 43.99%).

Warto zauważyć, że algorytm SVM, jako klasyfikator maksymalnego marginesu jest bardzo wrażliwy na wartości odstające w danych treningowych. Wybrana baza danych zawierała przykłady instrumentów użytych w wielu gatunkach muzycznych, co mogło negatywnie wpłynąć na odchylenia danych. Dodatkowo subiektywnie była to “trudna” baza danych, ponieważ

zawierała nagrania muzyczne instrumentów dominujących, a nie całkowicie odseparowanych od tła muzycznego.

6. Podsumowanie:

1. Wykonane czynności:

W ramach podjętych działań zaznajomiliśmy się z wybranymi modelami i podjęliśmy wysiłki w celu zoptymalizowania algorytmów i uzyskania najwyższych wyników. Przygotowaliśmy też odpowiedni preprocessing i przygotowanie analizy. Przeanalizowaliśmy również otrzymane dane i wyniki w celu wyciągnięcia wniosków w naszym badaniu.

2. Wyniki:

Wyniki zostały otrzymane jak w punkcie 5.3 i wyraźnie wskazują, iż na wybranej bazie danych IRMAS, lepiej funkcjonuje algorytm ExtraTrees, którego udało się lekko zoptymalizować. Wybraną metryką do porównania modeli było F1 score, gdyż jest odpowiednim do tego narzędziem, przez wzgląd na zawartość “precision” oraz “recall”, jako średniej harmonicznej.

3. Ocena wyników:

Mimo słabych wyników na jednym z algorytmów, wyniki są zadowalające, gdyż pokazało to dobrze, iż pewne modele mogą obsługiwać i nadawać się do pewnych zadań i danych lepiej lub gorzej od innych, bądź też mogą lepiej przetwarzać dane pewnego typu (przetestowaliśmy jedynie częściowo MFCC i parametry, nie wiadomo co byłoby z deltami). Sama optymalizacja modelu ExtraTrees była satysfakcjonująca i udała się chociaż w małym stopniu.

4. Propozycje i poprawki:

W celu uzyskania lepszych wyników klasyfikatora należy zwrócić uwagę na ilość danych w każdej klasie oraz zadbać o to, żeby była dostateczna, aby uniknąć przetrenowania modelu.

Warto również wziąć pod uwagę przetestowanie obu modeli na innym rodzaju danych tj. MFCC z deltami lub parametrami, które mogłyby pomóc w uzyskaniu lepszych wyników. Dodatkowo więcej czasu można by poświęcić na analizę parametrów i występujących zależności, jeżeli chcielibyśmy dostosować model pod nasz określony cel i dane.

Jako jedna z mniejszych rzeczy przydałoby się również więcej czasu lub mocy obliczeniowej, w celu przeprowadzenia dogłębniejszych analiz czy dłuższych optymalizacji z większą ilością parametrów i traili.

W przypadku ExtraTrees opcjonalnie można by sprawdzić podstawowy algorytm lasu losowego w celu porównania wyników.