Clasificarea sunetelor de chitară în minor sau major

Tema 1

Tehnici de învățare automată

Proiect realizat de:

Sătmar Elena – Elisabeta

Grupa 332AA

2023-2024

Nume: Sătmar Elena-Elisabeta
Grupă: 332AA

Descrierea temei alese:

Tema aleasă este clasificarea notelor muzicale produse de corzile unei chitări în două categorii: majore sau minore. Pentru urechea umană neantrenată, cele două categorii sunt foarte asemănătoare, acestea nediferețiând la înălțimea sau durata sunetului. Foarte mulți clasifică cele 2 tipuri ca fiind ”vesele” sau ”triste”. Astfel, cu ajutorul unui algoritm de clasificare și a unui set de date, am încercat să clasific tonalitățile corzilor în majore sau minore.

Set de date ales:

Pentru o acuratețe cât mai bună, am ales un set de date cu un total de 960 de de fișiere audio în format .wav. Acesta este o combinare de două seturi de date. Setul de date este împărțit în cele două categorii: major (care va fi notat cu 1) și minor (care va fi notat cu 0). Cele 960 de eșantioane sunt împărțite în mod egal în fiecare categorie, 480 de fișiere audio făcând parte din tonalitatea majoră și 480 din tonalitatea minoră. Pentru o acuratețe cât mai bună, toate fișierele au aceeași durată și frecvență.

Setul de date este împărțit în 80% date de training și validare și 20% date de testare. Acest lucru se va realiza direct din program cu o funcție de split între cele două subseturi.

Algoritm ales:

Pentru această temă am ales să folosesc SVM (Support Vector Machine) deoarece are o capacitate de a gestiona seturi de date cu mai multe caracteristici. Acest lucru ajută în prelucrarea unor date auditive, fiind formate din mai multe caracteristici precum lungime de undă, tonalitate, înălțime, durată etc.

Parametrii utilizați:

Ca și parametrii am folosit C=10000000, kernel="poly", gamma="scale".

Pentru C am ales o valoare mare întrucât setul de date ales este unul destul de mare. Acest lucru ajută algoritmul să clasifice cât mai corect setul de date.

Nume: Sătmar Elena-Elisabeta
Grupă: 332AA

Întrucât caracteristicile sunt mai complexe, am ales o functie polinomială pentru o evidențiere cât mai exacta.

Alegerea parametrului gamma a fost influențat, de asemenea, de setul de date ales. Parametrul ”scale” are în vedere ajustarea scălii de calcul în functie se numărul de date introduse.

Împărțirea setului de date se face aleatoriu, cu ajutorul unui parametru ”random_state” care are în vedere folosirea aceluiași set de date pentru mai multe rulări. Acest lucru ajută la verificarea și ajustarea codului. Acest parametru este optional și poate fi eliminat.

Rezultate:

În urma rulării programului s-au obținut mai multe rezultate:

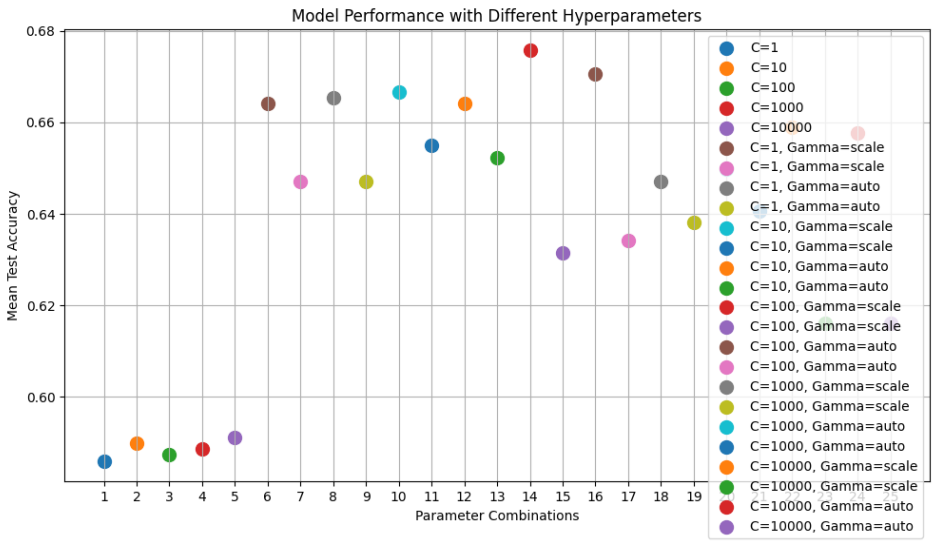

Figure 1 Performanta programului pe diferiti parametrii

Pentru o verificare dacă parametrii aleși au fost optimi, am realizat un grafic care să evidențieze cea mai bună alegere pentru o acuratețe cât mai mare:

```
Best Hyperparameters: {'C': 100, 'gamma': 'scale', 'kernel': 'rbf'}
Corresponding Accuracy: 0.6758254817078346
```

Astfel, se poate observa că parametrii aleși nu sunt cei mai buni, fapt ce poate duce la o acuratețe mai scăzută.

În figura de mai jos se poate vedea distribuirea setului de date:

Primul grafic indică că cele două clase (minor - 0 și major – 1) au un număr de date aproximativ egale pentru o precizie cât mai corectă. Cel de-al doilea grafic evidențiază modul de împărțire al setului de date în set de antrenare și validare (aproximativ 80% din setul de date) și set de testare (aproximativ 20% din setul de date)

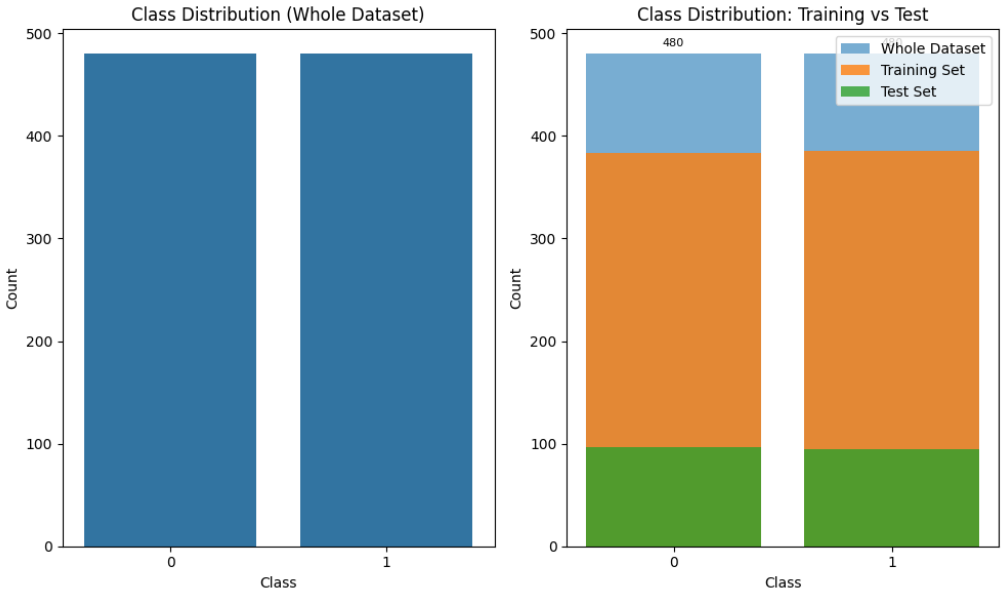

Figure 2 Distribuirea setului de date

Pentru parametrii aleși (C=10000000, kernel="poly", gamma="scale", random_state = 6) rezultatele sunt următoarele:

```
Cross-Validation Scores: [0.57792208 0.64935065 0.51948052 0.63398693 0.69934641]
Mean Cross-Validation Score: 0.616017316017316
Accuracy on Test Set: 0.671875
Classification Report on Test Set:
              precision    recall  f1-score   support

           0       0.66      0.73      0.69        97
           1       0.69      0.61      0.65        95

    accuracy                           0.67       192
   macro avg       0.67      0.67      0.67       192
weighted avg       0.67      0.67      0.67       192
```

```
Misclassified values:
Sample: 0, Predicted Label: 1, True Label: 0
Sample: 5, Predicted Label: 0, True Label: 1
Sample: 7, Predicted Label: 1, True Label: 0
Sample: 12, Predicted Label: 1, True Label: 0
Sample: 14, Predicted Label: 0, True Label: 1

Corectly classified values:
Sample: 1, Predicted Label: 0, True Label: 0
Sample: 2, Predicted Label: 1, True Label: 1
Sample: 3, Predicted Label: 0, True Label: 0
Sample: 4, Predicted Label: 1, True Label: 1
Sample: 6, Predicted Label: 1, True Label: 1
```

Se poate observa că acuratețea este de 0.67 pe setul de date și pe parametrii aleși, observându-se o precizie mai mare asupra categoriei de sunete majore. De asemenea, se pot observa și câteva exemple de instanțe clasificate greșit și câteva instanțe clasificate corect. Se poate observa că instanțele din categoria sunetelor majore se pot confunda mai ușor cu sunete din categoria minoră. Acest lucru este evidențiat și în matricea de confuzie a setului de test:

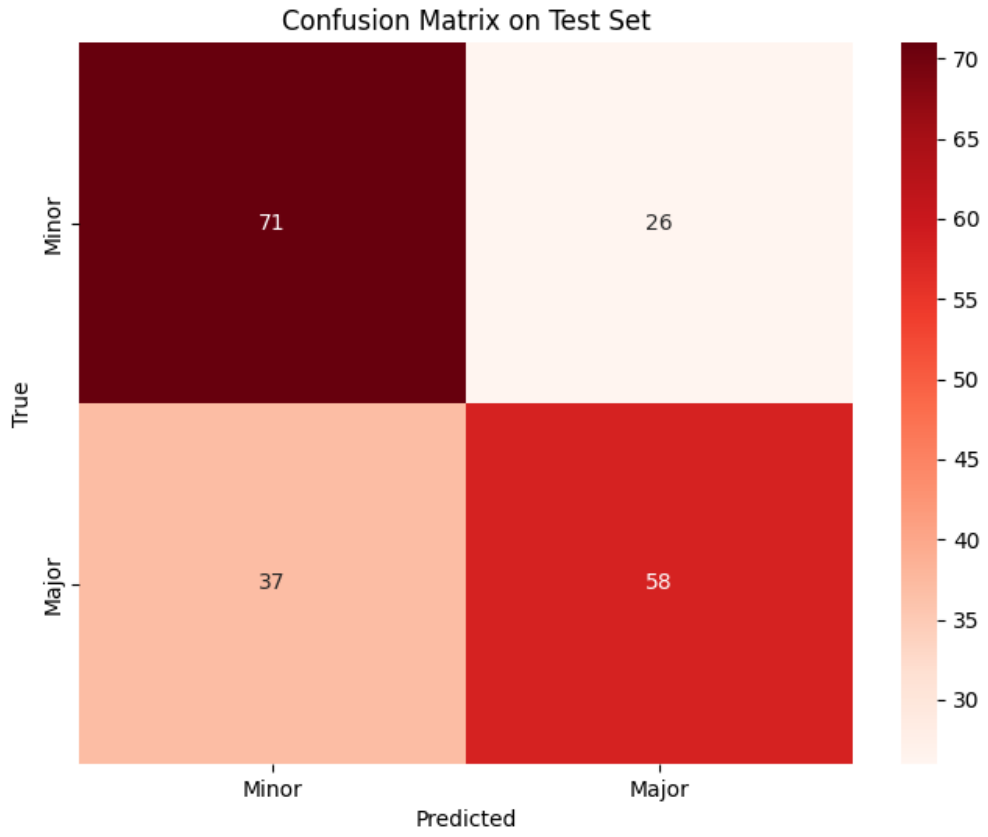

Figure 3 Matricea de confuzie pentru setul de test

Acest lucru se datorează și diferenței mici de sunet dintre cele două categorii, nefiind detectabil nici de o ureche umană neantrenată. Astfel, pentru o evidențiere și mai exactă a programului, am creat două grafice în care să evidențiez performanța modelului de clasificare:

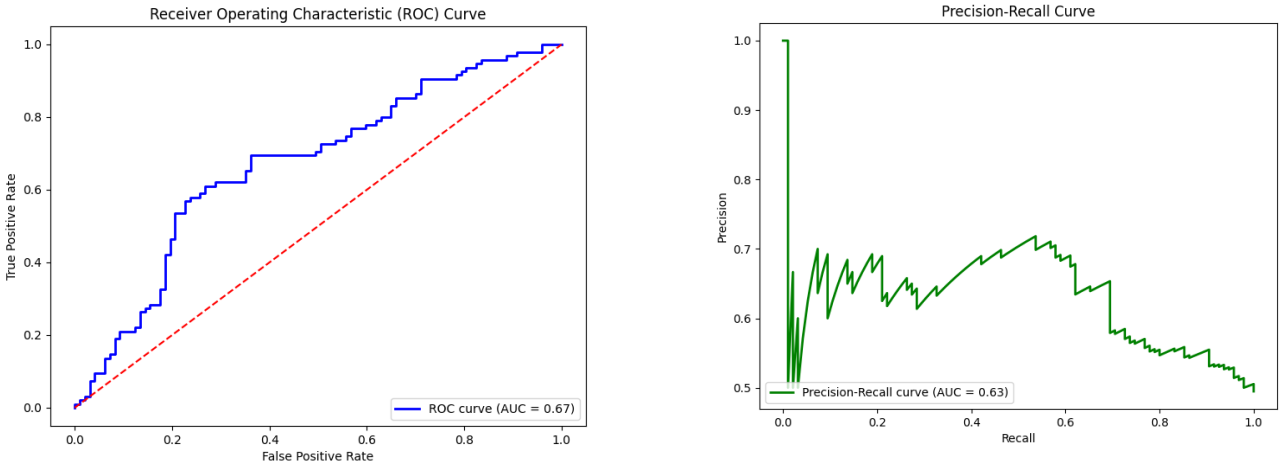

Figure 4

Curba ROC evidențiază faptul că algoritmul ales are o acuratețe de 67%, așa cum indică valoarea AUC. Astfel, deși mai bun decât alți algoritmi, plasâmdu-se deasupra diagonalei, modelul nu este perfect. Acest lucru este evidențiat în cel de-al doilea grafic al curbei de Precision-Recall.

Astfel, pentru a se face distingerea dintre tonurile majore și minore, algoritmul are șanse destul de mari să facă o presupunere greșită, datorită acurateții scăzute al acestuia. Deși acuratețea este peste 50% (șansele de a categorisi eșantionul în mod corect sunt mai mari decât de a îl clasifica greșit), nu este nici pe departe un algoritm eficient. Acest lucru se poate datora din mai multe cauze: setul de date este dificil de clasificat/ distins chiar și de mintea umană sau alegerea parametrilor nepotriviți.

Concluzii:

Setul de date joacă un rol important în antrenarea, validarea și testarea algorimului. Cu cât acesta este mai bine definit, cu atat algorimtul va avea o acuratețe mai mare. Astfel, este greu de antrenat un algoritm de clasificare a sunetelor create de o chitară în major sau minor întrucât acestea se aseamănă și pot fi destul de ușor confundate și de către persoane.

Configurația parametrilor este de asemenea importantă pentru o acuratețe cât mai mare asupra setului de date ales. Dacă unul sau mai mulți parametrii nu sunt aleși corespunzător, acuratețea algoritmului poate scădea semnificativ.

Datorită unui set mare de date, timpul de execuție al programului crește semnificativ, fapt ce poate crea un dezavantaj.

Bibliografie

Biblioteci Python utilizate:

- Os – pentru citirea setului de date din sistemul de operare
- Librosa – pentru analizarea si prelucrarea semnalelor audio
- Numpy
- Sklearn
- Matplotlib
- Seaborn – pentru creare de grafice

Setul de date este obținut prin îmbinarea a două seturi de date:

- https://www.kaggle.com/datasets/mehanat96/major-vs-minor-guitar-chords/data
- https://www.kaggle.com/datasets/deepcontractor/musical-instrument-chord-classification/data

Setul de date folosit și codul:

- https://drive.google.com/drive/folders/1uP7f2fjmcEAaF6bHLCRAq8G4jT_rr_4s?usp=drive_link