

Documentația Proiectului:
Clasificator Naive Bayes Multinomial pentru
Predictia Genului Muzical

Bălăceanu Rafael Gabriel

16 noiembrie 2025

Cuprins

1	Modelul Matematic: Naive Bayes Multinomial	1
1.1	Ipoteza "Naivă"	1
1.2	Probabilități Logaritmice	1
1.3	Netezirea Laplace (Aditivă)	1
2	Structura Codului și Funcțiile Principale	2
2.1	data_processing.py	2
2.2	MultinomialNBayes.py	2
2.3	main.py	3
3	Instrucțiuni de Utilizare	3
4	Exemplu de Utilizare	3
4.1	Output în Terminal	4
4.2	Sesiune Interactivă	4
5	Referințe Bibliografice	5

1 Modelul Matematic: Naive Bayes Multinomial

Pentru clasificarea genurilor muzicale, formula este:

$$P(\text{Gen}|\text{Versuri}) = \frac{P(\text{Versuri}|\text{Gen}) \times P(\text{Gen})}{P(\text{Versuri})} \quad (1)$$

Unde:

- **P(Gen | Versuri)**: probabilitatea ca o piesă să aparțină unui anumit *Gen*, având în vedere *Versurile* sale. Acesta este rezultatul pe care dorim să-l calculăm.
- **P(Versuri | Gen)**: probabilitatea de a întâlni *Versurile* date într-o piesă dintr-un anumit *Gen*.
- **P(Gen)** este probabilitatea a priori: probabilitatea generală ca o piesă să aparțină unui anumit *Gen* în setul nostru de date.
- **P(Versuri)** este probabilitatea de apartenență a versurilor. Aceasta este constantă deci o putem elimina.

1.1 Ipoteza "Naivă"

Modelul face o presupunere "naivă" de independență condițională: consideră că prezența fiecărui cuvânt în versuri este independentă de prezența celorlalte cuvinte. Astfel probabilitatea unei piese să aparțină unui anumit gen muzical este probabilitatea fiecărui cuvânt să aparțină acestuia.

$$P(\text{Versuri}|\text{Gen}) = P(\text{Gen}) \prod_{i=1}^n P(\text{cuvânt}_i|\text{Gen}) \quad (2)$$

1.2 Probabilități Logaritmice

Înmulțirea multor probabilități mici (între 0 și 1) poate duce la erori de calcul (under-flow numeric). Pentru a evita acest lucru, implementarea calculează suma logaritmilor probabilităților:

$$\log(P(\text{Versuri}|\text{Gen})) = \log(P(\text{Gen}) \prod_{i=1}^n P(\text{cuvânt}_i|\text{Gen})) \quad (3)$$

$$\log(P(\text{Gen}|\text{Versuri})) = \log(P(\text{Gen})) + \sum_{i=1}^n \log(P(\text{cuvânt}_i|\text{Gen})) \quad (4)$$

Genul cu cea mai mare probabilitate logaritmică este ales ca predicție finală.

1.3 Netezirea Laplace (Aditivă)

Pentru a gestiona cuvintele care apar în setul de test dar limitate la anumite genuri (ceea ce ar duce la o probabilitate de zero pentru celelalte genuri), se aplică netezirea Laplace. O valoare mică, **alpha** (setată în cod la 1.0), se adaugă la numărătorul fiecărui cuvânt, prevenind probabilitățile nule.

2 Structura Codului și Funcțiile Principale

Proiectul este organizat în trei fișiere Python principale:

2.1 data_processing.py

Acest fișier conține toate funcțiile legate de încărcarea, curățarea și preprocesarea datelor.

- `read_csv(csv_path, cols, ...)`: Citește coloanele specificate ("Genre", "Lyrics") din fișierul CSV într-un DataFrame pandas și elimină rândurile cu date lipsă.
- `tokens_text(text, ...)`: Funcția centrală de procesare a textului. Primește textul brut (versurile) și realizează următoarele operațiuni:
 1. Elimină caracterele speciale și conținutul din parantezele drepte (ex: "[Chorus]").
 2. Converstește textul la litere mici.
 3. Elimină punctuația.
 4. Elimină cuvintele comune din limba engleză ("stop words").
 5. Împarte textul curățat într-o listă de cuvinte (token-uri).
- `preprocess_data(data, column, ...)`: Aplică funcția `tokens_text` pe coloana de versuri a DataFrame-ului și stochează rezultatul într-o nouă coloană, numită "Tokens".

2.2 MultinomialNBayes.py

Acest modul conține implementarea clasificatorului.

- **clasa MultinomialNaiveBayes:**
 - `__init__(self)`: Inițializează parametrii modelului: probabilitățile **a_priori**, vocabularul, probabilitățile **condiționate** și factorul de netezire **alpha**.
 - `train(self, x_train, y_train)`: Ordonează procesul de antrenare prin apelarea următoarelor două metode:
 - * `calc_a_priori(self, y_train)`: Calculează probabilitatea a priori $P(\text{Gen})$ pentru fiecare gen.
 - * `calc_cond_voc(self, x_train, y_train)`: Construiește vocabularul și calculează probabilitatea condiționată $P(\text{cuvânt}|\text{Gen})$.
 - `predict(self, x_test)`: Primește o listă de versuri noi și prezice genul pentru fiecare.
 - `evaluate(self, x_test, y_test)`: Măsoară performanța modelului comparând predicțiile sale cu etichetele reale.

2.3 main.py

Acesta este scriptul principal care leagă toate componentele.

- **Încărcarea și Preprocesarea Datelor:** Citește fișierul `Light_Music_Dataset.csv` / `Heavy_Music_Dataset1.csv` și preprocesează versurile.
- **Antrenarea și Evaluarea Modelului:**
 - **Optiunea1** (default): Împarte setul de date într-un set de antrenament (80%) și unul de test (20%) folosind `train_test_split`.
 - **Optiunea2:** Împarte setul de date într-un set de antrenament în care fiecare gen are o anumită proporție din setul original și unul de test, restul datelor rămase, folosind `create_false_imbalance(lyrics, genres, procent_per_genre)`.
 - Inițializează, antrenează și evaluează modelul `MultinomialNaiveBayes`.
- `plot_confusion_from_dict_proportions(...)`: Vizualizează rezultatele evaluării sub forma unei matrici de confuzie. (Cod generat cu ChatGPT)
- `start_testing()`: Inițiază o buclă interactivă în care utilizatorul poate introduce versuri pentru a obține o predicție.

3 Instrucțiuni de Utilizare

1. **Cerințe preliminare:** Asigurați-vă că aveți Python instalat.
2. **Setul de date:** Fișierul folosit `Light_Music_Dataset.csv` sau `Heavy_Music_Dataset1.csv` trebuie să se afle într-un folder numit `Music-Datasets` (sursele fișierelor de date pot fi găsite la începutul fișierului `main.py` pentru a putea fi descărcate).
3. **Instalarea Dependințelor:** Deschideți un terminal și rulați următoarea comandă:

```
1 pip install -r requirements.txt
2
```
4. **Rularea Proiectului:** Executați scriptul principal din terminal:

```
1 python main.py
2
```

Scriptul va antrena și evalua mai întâi modelul, afișând rezultatele. Ulterior, va aștepta ca utilizatorul să introducă text pentru clasificare.

4 Exemplu de Utilizare

După rularea comenzii `python main.py`, procesul de antrenare și evaluare va fi afișat în terminal. La final, va porni prompt-ul interactiv.

4.1 Output în Terminal

```
1 Details about Genre:
2 Metal : 100000
3 rock : 99997
4 rap : 99975
5 pop : 100000
6 country : 100000
7 Preprocessing the data...
8 -----
9 Training on 399977 datas
10 Calculating the a priori probability:
11 country: 0.19959397665365758
12 Metal: 0.19994399677981484
13 rock: 0.20009900569282735
14 pop: 0.2001865107243666
15 rap: 0.20017651014933358
16 -----
17 Calculating the conditioned probability and finding the vocabulary:
18 Size of vocabulary: 432670
19 -----
20 -----
21 Evaluate the model accuracy on 99995 datas:
22 60.44102205110256%
23 -----
```

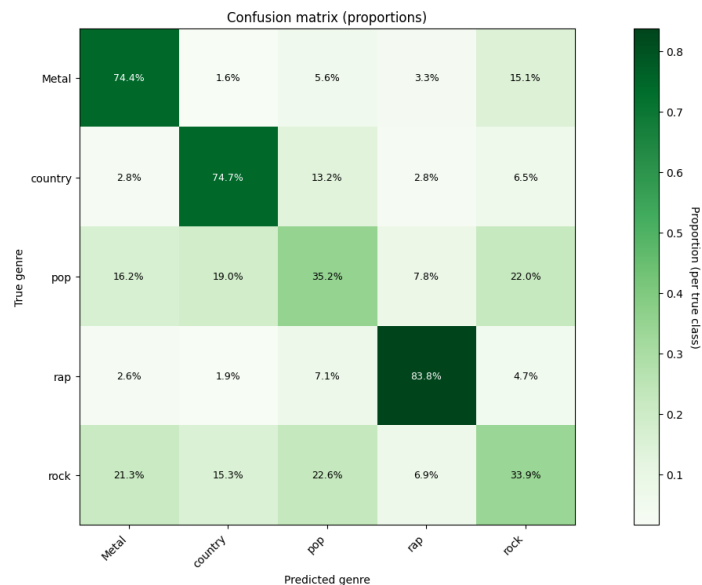


Figura 1: Graficul matricii de confuzie

4.2 Sesiune Interactivă

Programul va aștepta acum inputul dumneavoastră pe o singură linie (se poate folosi programul `oneLineLyrics.cpp`). Pentru a ieși, tastezi `EOF` și apăsăți `Enter`.

```
1 > Darkness, imprisoning me All that I see, absolute horror
2 metal
3
```

```
4 > Country roads, take me home To the place I belong  
5 country  
6  
7 > EOF
```

5 Referințe Bibliografice

- <https://www.geeksforgeeks.org/machine-learning/naive-bayes-scratch-implementation-using-python/>
- LLM-uri precum ChatGPT si Gemini
- Github