Sistem de recomandare folosind factorizare de matrice (NMF) și similaritate pe taguri

Gatej Stefan-Alexandru, Potop Horia-Ioan

May 26, 2025

1 Descrierea aplicației

Scopul acestei componente este de a recomanda melodii similare pe baza tagurilor asociate fiecărei piese muzicale. Problema se modelează ca o problemă de optimizare cu constrângeri: factorizarea non-negativă a unei matrice binare (melodie × tag).

2 Formularea matematică a problemei

Fie $V \in \mathbb{R}^{m \times n}$ matricea binară ce indică asocierea dintre m melodii și n taguri. Dorim să găsim două matrice $W \in \mathbb{R}^{m \times r}$ și $H \in \mathbb{R}^{r \times n}$, cu $W \geq 0$, $H \geq 0$, astfel încât:

$$\min_{W \geq 0, H \geq 0} \|V - WH\|_F^2 \tag{1}$$

Constrângeri:

- $W_{ij} \geq 0$
- $H_{ij} \geq 0$

Concepte cheie: Profil latent și Cluster latent

Profil latent Un profil latent este o reprezentare numerică, invizibilă direct, a unei entități (de exemplu, o melodie sau un utilizator) într-un spațiu de factori ascunși (latenti). Acești factori nu sunt direct observați în datele inițiale (cum ar fi tagurile sau genurile), ci sunt descoperiți automat de algoritmul de factorizare (NMF). Fiecare melodie va avea un vector de factori latenti (profil latent), de exemplu: $[0.2, 1.5, 0.0, 0.7, ...]$. Acești factori pot corespunde unor teme, stiluri sau combinații de caracteristici pe care algoritmul le-a identificat ca fiind relevante, dar care nu au neapărat o interpretare clară.

Cluster latent Un cluster latent este un grup de entități (melodii, utilizatori etc.) care au profiluri latente similare, adică sunt apropiate în spațiul factorilor latenti. Acest cluster poate corespunde, de exemplu, unui anumit gen muzical, unui anumit artist sau unui tipar de preferințe pe care algoritmul l-a descoperit, chiar dacă nu există un tag explicit pentru acel grup.

Pe scurt: Profilul latent este vectorul de factori ascunși care descrie o melodie/utilizator, iar un cluster latent este un grup de melodii/utilizatori cu profiluri latente similare.

3 Soluția implementată

Am implementat două metode pentru rezolvarea problemei:

3.1 Algoritmul multiplicativ pentru NMF

- Algoritm iterativ ce actualizează alternativ W și H folosind reguli multiplicative.
- Rapid și eficient pentru matrice mari.

3.2 NMF din scikit-learn (`sklearn.decomposition.NMF`)

- Implementare optimizată ce folosește coordinate descent.
- Oferă rezultate robuste și convergență rapidă.

Fragment de cod

```
def nmf_multiplicative(V, r, max_iter=100, tol=1e-4):
m, n = V.shape
np.random.seed(42)
W = np.abs(np.random.randn(m, r))
H = np.abs(np.random.randn(r, n))
for it in range(max_iter):
H *= (W.T @ V) / (W.T @ (W @ H) + 1e-10)
W *= (V @ H.T) / ((W @ H) @ H.T + 1e-10)
return W, H

from sklearn.decomposition import NMF
model = NMF(n_components=10, init='random', random_state=42, max_iter=100)
W_sklearn = model.fit_transform(track_tag_matrix)
H_sklearn = model.components_
```

4 Rezultate numerice și concluzii

4.1 Erori de reconstrucție

- **Multiplicative NMF:** 337.03
- **sklearn NMF:** 336.42

4.2 Recomandări generate

- **Tag similarity:** Recomandă melodii cu taguri identice/similare.
- **NMF multiplicativă:** Recomandă melodii cu profil latent apropiat, nu neapărat cu aceleași taguri.
- **NMF (sklearn):** Recomandă melodii din același cluster latent (ex: toate piesele unui artist).

4.3 Timp de execuție

Algoritmul multiplicativ și sklearn NMF rulează rapid (câteva secunde pentru setul de date testat).

4.4 Concluzii

- Ambele metode de NMF oferă rezultate similare ca eroare de reconstrucție.
- NMF descoperă similarități latente între melodii, nu doar pe baza tagurilor explicite.
- Recomandările pe factori latenti pot fi mai relevante pentru utilizator decât cele bazate strict pe taguri.

5 Comparare cu alte funcții Python

Pentru această problemă, funcțiile `scipy.optimize.minimize` nu sunt potrivite direct, deoarece nu pot gestiona eficient constrângeri de non-negativitate pe matrice mari (ar fi fost extrem de lent). NMF din `sklearn` este optimizată pentru această clasă de probleme și oferă rezultate competitive cu algoritmul multiplicativ clasic.

6 Bibliografie

- Lee, D. D., & Seung, H. S. (2001). Algorithms for non-negative matrix factorization. Advances in neural information processing systems, 13.
- `https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.NMF.html`
- `https://en.wikipedia.org/wiki/Non-negative_matrix_factorization`