Movie Recommender System

Almaz Dautov, a.dautov@innopolis.university

Introduction

Recommender systems play a crucial role in various domains, offering personalized suggestions to users based on their preferences, interests, and past behaviors. The goal is to develop a machine learning model that suggests movies to users, taking into account their metadata details and favorite movies. The MovieLens 100K dataset, comprising 100,000 ratings from 943 users on 1682 movies, will serve as the foundation for this task. The dataset provides user ratings, demographic information, and details about each movie.

Data analysis

Items

drop columns that don't have crucial information: `'movie_title'`,`'release_date'`,`'video_release_date'`, `'IMDb_URL'` and aggregate genres for future simplicity to pass to the model.

	item id	genre
0	1	Animation, Children's, Comedy
1	2	Action, Adventure, Thriller
2	3	Thriller
3	4	Action, Comedy, Drama
4	5	Crime, Drama, Thriller
...	...	...
1677	1678	Drama
1678	1679	Romance, Thriller
1679	1680	Drama, Romance
1680	1681	Comedy
1681	1682	Drama

1682 rows × 2 columns

Users

drop `'zipcode'`, it can be useful for something like regional recommendations, but the number of users is low in our dataset.

Divide `'age'` into groups:

age	group
<13	child
13 - 18	teenage
19 - 30	adults

31 - 45	middle age
>45	old adults

it can be useful, for example childs are more likely would watch cartoons(in our dataset genres:[Animation, Children's])

What has been removed
Feature engineering:
History of ratings for each timestamp (how many films of each category user_id rated before rating item_id)

Model Implementation

For the model implementation, the LightFM library was chosen as the primary tool. LightFM is a hybrid recommendation model that integrates collaborative filtering and content-based filtering for personalized recommendations. The model employs matrix factorization to decompose the user-item interaction matrix, learning latent representations for users and items. Users and items are represented as embeddings in a shared latent space, capturing their underlying characteristics. The model calculates interaction scores through a weighted sum of these embeddings, generating predictions by computing the dot product of user and item embeddings.

LightFM supports various loss functions, including logistic loss, Bayesian personalized ranking (BPR) loss, and WARP (Weighted Approximate-Rank Pairwise) loss, allowing flexibility based on the recommendation task. Hyperparameters like learning rate, regularization terms, and the number of components can be tuned for optimization. The model handles sparse data effectively and provides evaluation metrics such as precision at k, recall at k, and AUC for assessing recommendation quality. LightFM is implemented in Python, offering a user-friendly approach for researchers and practitioners, making it versatile for a broad spectrum of recommendation scenarios.

Model Advantages and Disadvantages

Advantages

Hybrid Recommendation Support
LightFM supports both collaborative and content-based filtering, allowing for a hybrid approach.

Scalability
LightFM is designed for scalability, making it suitable for handling large datasets efficiently.

Optimization Techniques
The library incorporates optimization techniques that contribute to faster model training and convergence.

Evaluation Metrics
LightFM provides built-in tools for evaluating model performance using metrics such as precision, recall, and AUC.

Disadvantages

Limited Support for Numeric Values
One notable limitation of LightFM is its inherent inability to directly handle numeric values in the user-item interactions.

Limited to Python
LightFM is implemented in Python, which may be a limitation if a project requires integration with systems or platforms that primarily use other programming languages.

Dependency on Feature Quality
The effectiveness of LightFM can be influenced by the quality and relevance of the features used in the model. If the side information or additional features provided are not meaningful, it may impact the overall recommendation accuracy.

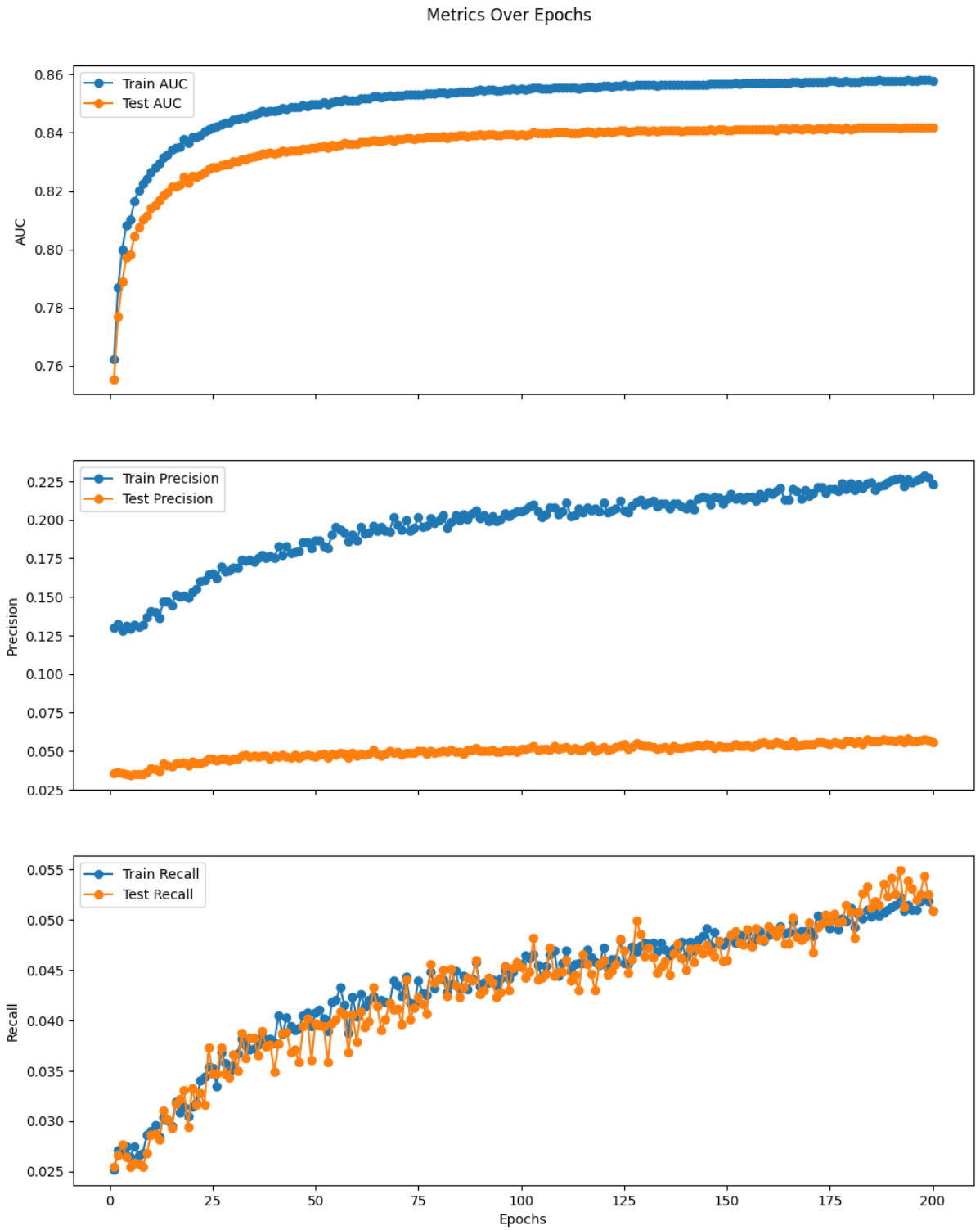
Documentation and Community Support
While LightFM has documentation and community support, it may not be as extensive as some other popular machine learning libraries.

Training Process

training params:

```
LEARNING_RATE = 0.01
NO_COMPONENTS = 200
NO_EPOCHS = 200
ITEM_ALPHA = 0
USER_ALPHA = 0
```

all other parameters are default


Metrics Over Epochs
Train AUC
Test AUC
AUC
0.86
0.84
0.82
0.80
0.78
0.76
Train Precision
Test Precision
Precision
0.225
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
Train Recall
Test Recall
Recall
0.055
0.050
0.045
0.040
0.035
0.030
0.025
0
25
50
75
100
125
150
175
200
Epochs

Evaluation

Model	ROC AUC	Precision@10	Recall@10
lightFM with FE*	0.7907	0.0364	0.0266
lightFM CF	0.8351	0.1083	0.0775
lightFM hybrid	0.8417	0.0560	0.0509
lightFM hybrid tuned	0.8474	0.0609	0.0639

*FE: History of ratings for each timestamp(this feature only adds complexity, which is why the model's performance only decreases)

Results

Example of inference:

```
Films watched by user: 2
```

	item id	movie_title	genre
180	181	Return of the Jedi (1983)	Action, Adventure, Romance, Sci-Fi, War
259	260	Event Horizon (1997)	Action, Mystery, Sci-Fi, Thriller
319	320	Paradise Lost: The Child Murders at Robin Hood Hills (1996)	Documentary
320	321	Mother (1996)	Comedy
326	327	Cop Land (1997)	Crime, Drama, Mystery
330	331	Edge, The (1997)	Adventure, Thriller
339	340	Boogie Nights (1997)	Drama
341	342	Man Who Knew Too Little, The (1997)	Comedy, Mystery
345	346	Jackie Brown (1997)	Crime, Drama
346	347	Wag the Dog (1997)	Comedy, Drama

```
Recommended:
```

	item id	movie_title	genre
11	12	Usual Suspects, The (1995)	Crime, Thriller
194	195	Terminator, The (1984)	Action, Sci-Fi, Thriller
203	204	Back to the Future (1985)	Comedy, Sci-Fi
233	234	Jaws (1975)	Action, Horror
257	258	Contact (1997)	Drama, Sci-Fi