

Wrap-up Report

RecSys_3조 (7해쥬)

강성택, 김다빈, 김윤경, 김희수, 노근서, 박영균

1. 프로젝트 개요

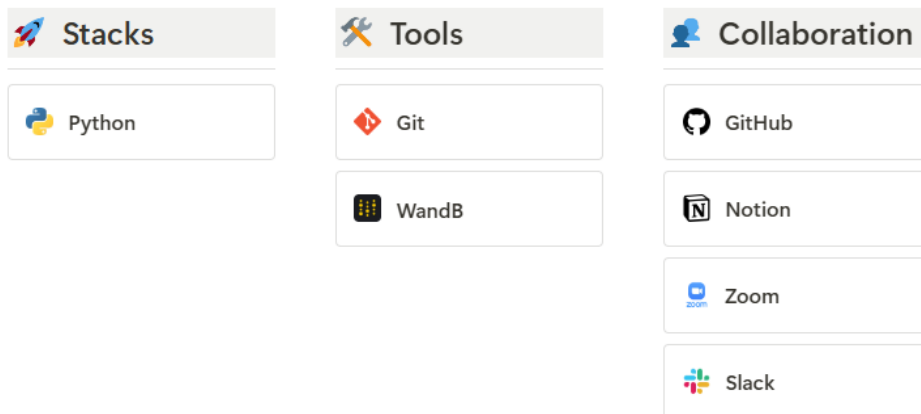
1.1 개요

책과 관련된 정보와 소비자의 정보, 그리고 소비자가 실제로 부여한 평점, 총 3가지의 데이터 셋을 활용하여 각 사용자가 주어진 책에 대해 얼마나 평점을 부여할지에 대해 예측한다.

이는 곧 소비자들의 책 구매 결정에 대한 도움을 주기 위한 개인화된 상품 추천과 같다.

1.2 환경

- (팀 구성) 6인 1팀
- (컴퓨팅 환경) V100 서버를 SSH로 연결하여 사용
- (협업 환경) Notion, Github, WandB
- (의사 소통) Zoom, Slack



2. 프로젝트 팀 구성 및 역할

데이터팀 / 모델팀으로 3:3 나눈 후, 각 팀마다 한 명씩 짝지어서 **페어 프로그래밍** 방식으로 진행

공통

서버 구축, EDA

Data 팀

- 공통 - 모듈화, 데이터 인코딩, text 정규화, 코드 리팩토링
- 노근서 - 여러 평균(산술, 조화, 베이지안, Steam rating formula)을 유저 활동 레벨에 따라 평균 평점 매핑
- 김다빈 - 결측치 처리, Age 범주화, CatBoost, Optuna, WandB 구성, 상위 카테고리 매핑
- 강성택 - Cold start 및 파레토 법칙, Stratified K-Fold, CatboostpruningCallback, CatBoost

Model 팀

- 공통 - 모듈화, 코드 리팩토링
- 박영균 - 트리 모델(CatBoost, XGBoost, LightGBM) 및 Optuna 모듈화, PCA 활용한 임베딩 벡터 변수 생성
- 김희수 - 트리모델 WandB 연결, Steam Rating Formula, book-based modeling
- 김윤경 - Stratified K-Fold, (Tree, text, image) Ensemble, user-based modeling

3. 프로젝트 수행 절차 및 방법

3.1 팀 목표 설정

진행 기간 : 10/28 (수) 10:00 ~ 11/7 (목) 19:00

- (1주차) 강의 전부 듣기, 데이터셋 이해, 각자 EDA 완성(구인구팀데이)
- (2주차) 데이터 전처리, 모델 모듈화, EDA별 실험, 모델 앙상블

3.2 프로젝트 계획

1. Project Rule, Github/Coding Convention(PEP8을 기초로 설정) 설정
2. 프로젝트 개발 환경 구축 - 서버 세팅, Github Template 적용
3. 데이터셋 사전 공부
4. EDA 진행 및 인사이트 도출
5. 데이터팀/모델팀 코드 개발 & 모듈화
6. EDA별 실험 및 성능 비교(WandB 사용) & 모델 튜닝 : 교차 검증, 하이퍼파라미터 튜닝
7. 앙상블
8. 코드 리팩토링 & 프로젝트 코드 최종 모듈화

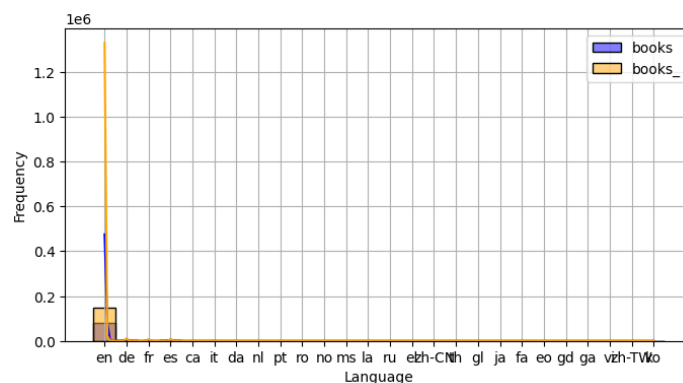
3.3 협업 방식

- 이슈 관리 - Github의 Issues + Notion 데이터베이스 보드로 진행
 - Github의 Issues : Issue Template(개요, TODO 작성)
 - 총괄 보드 : 전체 진행 상황 관리(일정, 제출 관리 등)
 - Data 보드 : 데이터 전처리, 피쳐 엔지니어링 함수 및 모듈화 진행상황 관리
 - Model 보드 : 모델 설계, 모듈화, 앙상블 개발 등 진행상황 관리
- Github Convention에 따라 Github 관리
- Coding Convention에 따라 개발
- WandB를 활용한 실험 관리

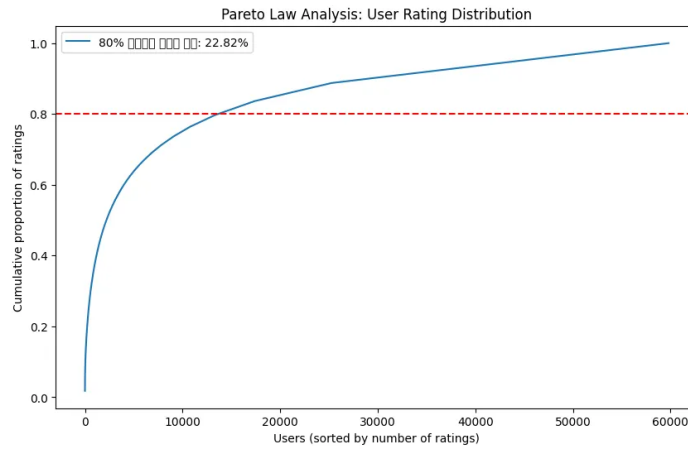
4. 프로젝트 수행 결과

4.1 탐색적 데이터 분석(EDA)

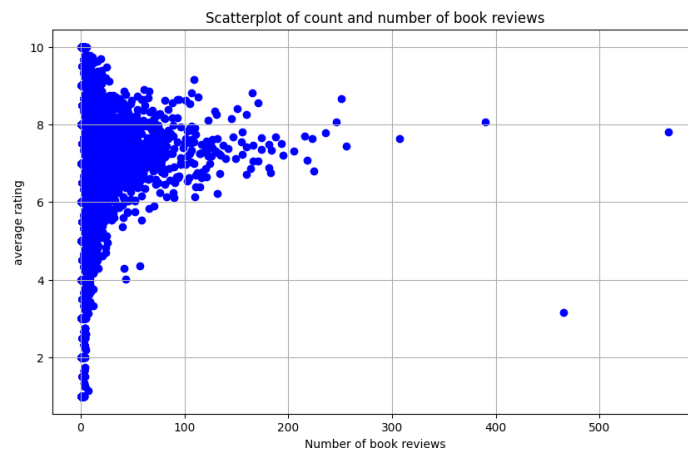
- language 변수는 결측치가 40% 이상이기 때문에 최빈값으로 대체할 경우 데이터의 분포가 왜곡될 수 있어 ISBN 활용해 결측치 대체



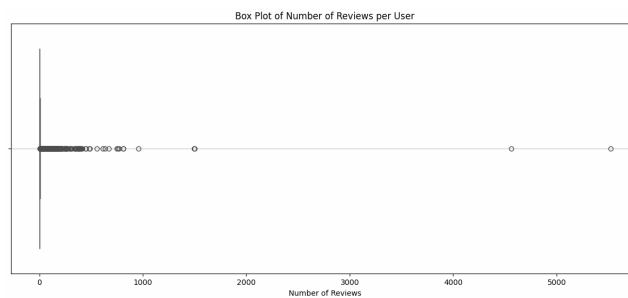
- 파레토 법칙에서 고안해낸 리뷰수가 많은 유저가 책 평점에 영향을 줄 것으로 보아 파생변수 생성



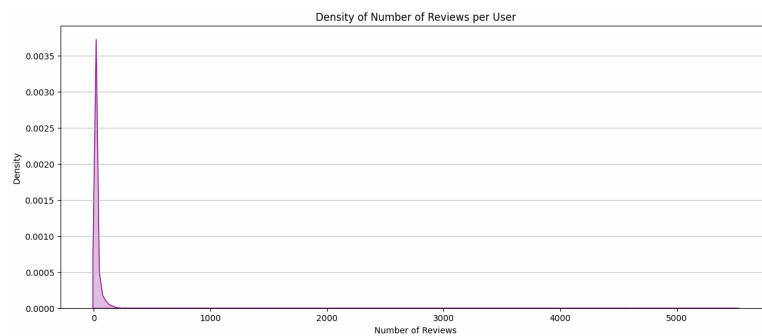
- 책에 대한 리뷰수가 많을수록 책 평점에 긍정적인 영향을 줄 것으로 예상해 파생변수 생성



- 유저별 평균 평점을 활용한 예측을 위해 파생변수 생성
 - 유저별 평점 수가 오른쪽으로 꼬리가 긴 분포인 점을 보완하기 위해 평균 평점 계산시 다양한 스코어 적용



유저별 평점 수의 Box Plot - 극단적으로 적은 값의 비율이 매우 높다.



유저별 평점 수의 분포 확인 - 극단적으로 적은 값의 비율이 높은 점을 감안해 KDE Plot 활용

1. 평점 수가 20개 이상 → 산술평균 또는 조화평균 적용
 2. 평점 수가 10~20개 → 베이지안 평균 적용
 3. 평점 수가 1~9개 → Steam Rating Formula 적용
- 콜드 스타트 문제를 해결하기 위해 유저 데모그래픽 정보를 활용, 공통 데모그래픽 정보를 갖는 평점을 매긴 유저의 리뷰 수에 따라 적절한 평균을 사용
 1. 평점 수가 10개 이상 → 산술평균 또는 조화평균 적용
 2. 평점 수가 10개 미만 → 베이지안 평균 또는 Steam Rating Formula 적용
 3. 기존에 존재하는 유저(Warm Start) → 이전에 계산한 스코어 유지
 - 스코어별 특징
 - **산술평균**: 주로 데이터가 고르게 분포했을 때 유용, 평점에 극단적인 값이 많을수록 영향을 크게 받음
 - **조화평균**: 작은 값에 더 민감하게 반응, 낮은 평점에 더 많은 가중치 부여
 - **Steam Rating Formula**: 평점 수가 적을수록 극단적인 값이 아닌 기준값으로 수렴하게 해 평가가 적은 리뷰가 전체 평점에 미치는 영향을 줄임

$$\text{average rating} = \frac{\text{sum}(\text{rating})}{\# \text{ of reviews}}$$

$$\text{score} = \text{average rating} - (\text{average rating} - 5.5) \cdot 2^{-\log_{10}(\# \text{ of reviews} + 1)}$$

- **베이지안 평균**: 평점 수가 적을수록 사전 확률로 정한 값(보통 전체 평점)을 가중치(보정값)를 더해 사용, 평점 수가 많아질수록 점차 실제 유저 평균으로 수렴

$$\text{Bayesian Average} = \frac{\sum x_i + m \cdot C}{n + m} \quad \text{where} \quad \begin{cases} C = \text{total average,} \\ n = \# \text{ of reviews,} \\ m = \text{constant} \end{cases}$$

4.2 데이터 전처리

- `text_preprocessing` 함수를 이용하여 `book_title`, `book_author`, `publisher`, `category` 변수 전처리
- `language` 결측치를 ISBN 국가 코드 기준 국가에 맞게 대체(예외 → 최빈값(en)으로 대체)
- `category` 변수를 리스트로 변환 후 첫 번째 값만 사용
- `location` 를 리스트로 변환 후 `location_country`, `location_city`, `location_state` 로 분리
결측치는 다른 변수 값으로 대체 후 나머지는 최빈값으로 대체
- `age` 결측치를 평균으로 대체

4.3 피처 엔지니어링

[파생변수 생성]

- 출판 연도(`publication_range`): 책의 출판 연도를 5년 단위로 범주화
- 사용자 나이(`age_range`): 이산형인 나이를 10년 단위로 범주화하여 기존 나이를 연령대별로 변환
- 상위 카테고리(`high_category`): 세부 도서 카테고리를 핵심 주요 주제로 범주화하여 복잡도를 줄임
- 리뷰 횟수(`user_review_counts`, `book_review_counts`): 사용자의 리뷰 횟수로 활동성을 측정하는 피처와 도서의 리뷰 횟수 책의 베스트 셀러 정도를 나타내는 피처를 추가
- 유저별 평균 평점(`average_rating`): Steam Rating Formula와 베이지안 평균을 활용해 평점을 예측하는 피처를 추가

[변수 선택]

- 범주형 변수: `user_id`, `age_range`, `location_country`, `isbn`, `book_title`, `book_author`, `publisher`, `language`, `high_category`, `publication_range`

- 수치형 변수: `user_review_counts`, `book_review_counts`

4.4 모델 선정 및 분석

[모델 비교]

다양한 모델(CatBoost, XGBoost, Image_DeepFM, Text_DeepFM + Ensemble)을 활용하여 `rating` 예측 성능을 비교. 모델 성능 비교에 사용한 평가지표로 `RMSE`를 중점적으로 활용

[양상블 기법 분석]

성능 개선을 위해 Weighted 앙상블을 활용하여 여러 모델의 예측을 결합. 각 모델로부터 얻은 서로 다른 예측값을 입력하여 앙상블의 성능을 극대화하는 전략을 적용했으며, 더 높은 성능을 보인 모델에 더 높은 가중치를 부여하여 예측 정확도를 최적화

[최종 모델 선정]

카테고리형 데이터와 구조화된 데이터를 처리하는 데 탁월한 성능을 보인 CatBoost 모델과, 여러 모델의 강점을 결합하여 더 나은 예측 성능을 보인 Weighted Ensemble 모델 각각을 최종 모델로 선정

[모델 튜닝]

- **Stratified k-fold** 교차 검증 적용
- **Optuna**를 활용한 하이퍼파라미터 튜닝
- **Weighted Ensemble**로 최종 결과 도출. (CatBoost 0.8 : Image_DeepFM 0.1 : Text_DeepFM 0.1)

4.5 모델 평가 및 개선 방법

[평가]


모델의 성능 평가는 `train RMSE`, `valid RMSE`, `MAE`, `MSE` 네 가지 평가 지표를 사용하여 수행.

[개선 방법]


- **Feature Engineering**: 도메인 지식, 각 데이터 특성을 고려해 결측값을 처리하고 파생변수를 생성하여 모델의 학습을 최적화
- **하이퍼파라미터 튜닝**: Optuna를 활용해 모델 각각의 하이퍼파라미터를 조정하여 최적의 성능 도출



4.6 모델 성능 및 결과

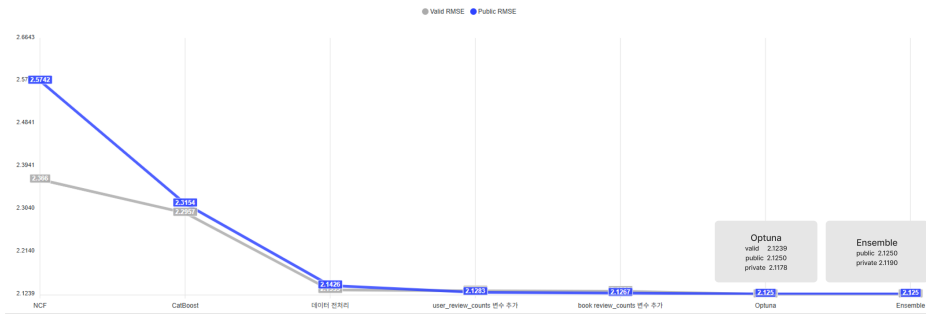
리더 보드[중간 순위]

1	RecSys_03조		2.1250	45	22h
---	------------	---	--------	----	-----

리더 보드[최종 순위]

1	RecSys_03조		2.1178	45	22h
---	------------	---	--------	----	-----

<input checked="" type="checkbox"/>	Catboost_stf...ptuna		2.1250 2.1178	2024.11.06 20:55	완료
제출1. CatBoost + Stratified K-fold + Optuna					
<input checked="" type="checkbox"/>	Ensemble_Wei...d_0.8		2.1250 2.1190	2024.11.07 00:39	완료
제출2. CatBoost + Image DeepFM + Text DeepFM 8:1:1 앙상블					



	Valid RMSE	Public RMSE	Private RSME	Model	전처리	변수
NCF	2.3660	2.5742	2.5762	NCF	베이스라인	Set 1
CatBoost	2.2957 ▼	2.3154 ▼	2.3150 ▼	CatBoost	베이스라인	Set 1
데이터 전처리	2.1333 ▼	2.1426 ▼	2.1375 ▼	CatBoost	4.2 데이터 전처리	Set 2
user_review_counts 변수 추가	2.1314 ▼	2.1283 ▼	2.1222 ▼	CatBoost	4.2 데이터 전처리	Set 2 + user_review_counts
book_review_counts 변수 추가	2.1306 ▼	2.1267 ▼	2.1203 ▼	CatBoost	4.2 데이터 전처리	Set 2 + user_review_counts, book_review_counts
Optuna (최종 제출)	2.1239 ▼	2.1250 ▼	2.1178 ▼	CatBoost	4.2 데이터 전처리	Set 2 + user_review_counts, book_review_counts
Ensemble (최종 제출)	2.1239 (-)	2.1250 (-)	2.1190 ▲	CatBoost	4.2 데이터 전처리	Set 2 + user_review_counts, book_review_counts

(참고) 변수 선택

- Set 1

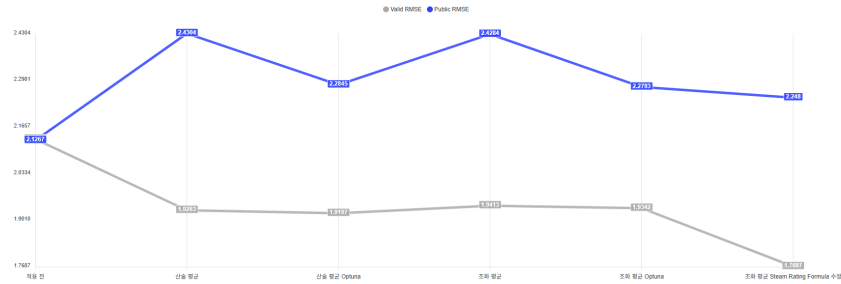
user_id, age_range, location_country, location_state, location_city, isbn, book_title, book_author, publisher, language, publication_range

- Set 2

user_id, age_range, location_country, isbn, book_title, book_author, publisher, language, high_category, publication_range

4.7 시행 착오

- 유저별 평점(average_rating)이라는 파생변수를 만들기 위해 다음과 같은 과정을 거쳤다.
 - 유저별 평점 분포가 다르다는 점, 유저별 평점 수가 극단적으로 적은 경우가 대부분인 점(평점 수 상위 10%가 4개 이상)을 고려해 다양한 평균(산술 · 조화 · 베이지안 평균, Steam Rating Formula)을 도입했고, 이를 평점 수의 구간마다 각 평균들의 특징을 살려 각각 매핑시키게끔 파생변수를 생성했다.
 - 한편 콜드 스타트 문제를 해결하기 위해 다음 방식을 사용했다.
 - test data에 처음 등장하는 유저의 데모그래픽 정보와 똑같은 정보를 갖는 유저를 train data에서 찾는다.
 - train data에서 미리 계산한 다양한 평균 평점(산술 · 조화 · 베이지안 평균, Steam Rating Formula)을 활용할 수 있게 평균 평점의 평균으로 사용했다. (마치 표본평균의 평균처럼)
 - 그리고 같은 데모그래픽 정보를 갖는 유저의 평점 수에 따라 다른 평점을 매기는 방식을 test data에서도 적용했다.
 - 평점 수가 10 미만이면 베이지안 평균 또는 Steam Rating Formula
 - 평점 수가 10 이상이면 산술평균 또는 조화평균
 - 하지만 책별 평점 수(book_review_counts), 유저별 평점 수(user_review_counts), 유저별 평균 평점(average_rating)를 추가로 사용했을 때 valid RMSE에 비해 public RMSE가 높게 측정됐다.



- valid 대비 public에서 **RMSE**가 오른 이유는 다음과 같이 예상했다.
 - 특정 평점 수를 기준(현재는 10 또는 20으로 적용)으로 일괄적으로 평균 계산 방식을 묶어버려 유저별 평점 분포에 각자 다른 평균을 적용시키지 못했다. 그로 인해 다양하게 나타나는 평점 분포를 전부 커버할 수 없었을 것으로 예상된다.
 - 콜드 스타트 문제를 극복하기 위해 **user feature**를 활용했지만 **item feature**를 활용하지 않았기 때문에 적절하지 않은 방법일 수 있다.
- 개선 방법으로 Steam Rating Formula의 **log term에 +1을 적용**시켜 평점을 하나만 매긴 상황에서 선택한 평점과 중앙값 사이에서 유저가 매긴 평점으로 더 치우쳐지게끔 조절해보았고, 그 결과 0.03 만큼의 미세한 **public RMSE** 성능 향상을 확인할 수 있었다.

	Valid RMSE	Public RMSE	Model
적용 전	2.1305	2.1267	CatBoost
조화 평균 사용	1.9413 ▼	2.4284 ▲	CatBoost
조화 평균 사용 + Optuna 적용	1.9342 ▼	2.2783 ▼	CatBoost
조화 평균 사용 + Steam Rating Formula 수정	1.7687 ▼	2.2480 ▼	CatBoost

summary 임베딩 벡터 활용

아무래도 중요해 보이는 텍스트 데이터인 **summary**를 전혀 활용하지 않은 것이 마음에 걸려 이를 이용해보기로 했다. **text_data.py**에 의해 생성된 **summary** 변수의 유저, 책 요약 임베딩 벡터를 각각 PCA로 5차원 벡터로 변환한 뒤, 이를 열 개의 새로운 열로 추가하여 CatBoost 학습에 이용해 보았다.

	Valid RMSE	Public RMSE	Model
적용 전	2.1306	2.1267	CatBoost
PCA를 활용한 임베딩 벡터 변수 추가	2.1288 ▼	2.4313 ▲	CatBoost

결과적으로는 **public RMSE**가 상승하여 오히려 좋지 않은 결과를 가져왔다. 이에 일반적인 트리 모델 기법에서는 텍스트 데이터의 특징적인 부분을 잘 잡아내지 못하는 것으로 보인다는 사실을 안 것으로 만족했다.

5. 자체 평가 의견

잘한 점

- EDA 기반 구현**: 각자의 EDA를 종합하여 데이터의 다양한 인사이트를 도출했다. 특히 변수 간의 상관관계나 상호작용을 고려한 새로운 특성을 생성하거나, 기존 변수들을 재구성함으로써 더 정교한 예측이 가능해졌고, 이는 성능 향상으로 이어졌다.
- 적극적인 피드백 반영**: 지난 프로젝트의 피드백을 바탕으로 협업, 기록, 코드 구현 등 다양한 방식에서 발전했다. 특히 WandB를 활용해 실험 관리와 시각화를 체계적으로 수행했고, 각자의 역할을 세분화해 소통하여 효율적으로 진행할 수 있었다. 결과적으로 협업 방식이 더 원활해졌다.
- 도메인 지식 활용**: 파레토 법칙(80/20)을 활용하여 중요한 변수나 데이터를 선별하였다. 전체 데이터에서 성능에 큰 영향을 미치는 상위 20%의 중요한 요소를 파악하고, 이를 중심으로 모델을 최적화하는 접근을 취한 것이 좋은 결과를 이끌어냈다.

아쉬운 점

- 시간관리**: 이번 프로젝트는 스프린트처럼 4일 만에 진행되었기 때문에, 빠르게 목표를 달성하기 위해 집중할 필요가 있었다. 그러나 짧은 시간 안에 모든 작업을 완료하려다 보니 Github 및 문서 관리를 제대로 진행하지 못하였으며 충분한 검토나 테스트를 하지 못한 점이 아쉬웠다.

- **단일 평가지표 사용** : 이번 프로젝트에서 RMSE, MAE, MSE 세 가지 평가지표를 사용했지만, 실질적으로 모델 평가 시 가장 주로 확인한 평가지표는 대회 평가지표였던 RMSE였다. 여러 평가지표를 동시에 확인하였다면 모델을 좀 더 균형 있게 평가하고, 분석을 할 수 있었을 것 같다는 점에서 아쉬웠다.
- **프로젝트 접근법** : 평가 지표가 RMSE인 것에서도 그렇고, 프로젝트 진행 중에는 회귀 문제로만 생각했는데, 프로젝트가 끝나고 보니 분류 문제로서의 접근도 가능했을 거라는 생각이 들었다. 결국 평점 데이터는 정수형으로 주어졌다는 점과 로지스틱 회귀 등의 전통적인 기법이 추천 분야에서 여전히 많이 쓰인다는 점을 고려했을 때, 지표 문제 정도만 잘 처리하면 분류 기법을 활용하는 것도 충분히 좋을 것이라는 아쉬움이 남는다.

이번 프로젝트를 통해 배운 점

→ 여전히 3 GUNA!

1. 모듈화를 잘 하면 이렇게 좋구나!
2. WandB와 노션 데이터베이스를 활용했더니 이렇게 좋구나! (실험 및 제출 관리의 필요성)
3. 그리고 언제나 도메인 지식은 알면 알수록 중요하구나!

다음 프로젝트에 적용해볼 점

→ 3 GI!

1. 이번 프로젝트 구조를 참고하여 다음 프로젝트 진행하기!
2. 가장 성능이 뛰어난 모델을 기준으로만 실험을 진행하는 것이 아닌 다양한 모델과 함께 실험을 진행하면서 기록하기!
3. 모델을 단순히 점수 기준으로만 사용하지 말고, 모델 구조에 대한 이해를 한 후, 모델을 선정하기!