Wrap-Up Report, RecSys06

Team 따봉 도치

Team 소개

👍 **따봉과 함께하는 즐거운 팀!**

> BoostCamp AI Tech 4기 추천 시스템 6조, **따봉도치!**
>
> **T4040_김성연**
> **T4056_김찬호**
> **T4079_박문순**
> **T4096_배성수**
> **T4171_이지훈**

우리팀의 목표

"첫 프로젝트인 만큼 경험과 성장에 집중하자!"

- **협업 역량**
- **데이터 분석**
- **모델 구조**
- **개발 역량**
- **추천 시스템 성능 향상**

목차

1-1. 프로젝트 개요

▼ 프로젝트 주제

📖 **Book Rating Prediction**

사용자의 책 평점 데이터를 바탕으로 사용자가 어떤 책을 더 선호할지 예측하는 태스크

▼ 데이터 개요

- 학습 데이터는 306,795건의 평점 데이터(`train_rating.csv`)
  - user_id, isbn, rating
- 149,570건의 책 정보(`books.csv`)
  - isbn, book_title, book_author, year_of_publication, publisher, img_url, language, category, summary, img_path
- 68,092명의 고객 정보(`users.csv`)
  - user_id, location, age

▼ 활용 장비 및 재료

개발환경 (AI Stage Server)

- OS: Ubuntu 18.04.5 LTS
- GPU: Tesla V100-SXM2-32GB

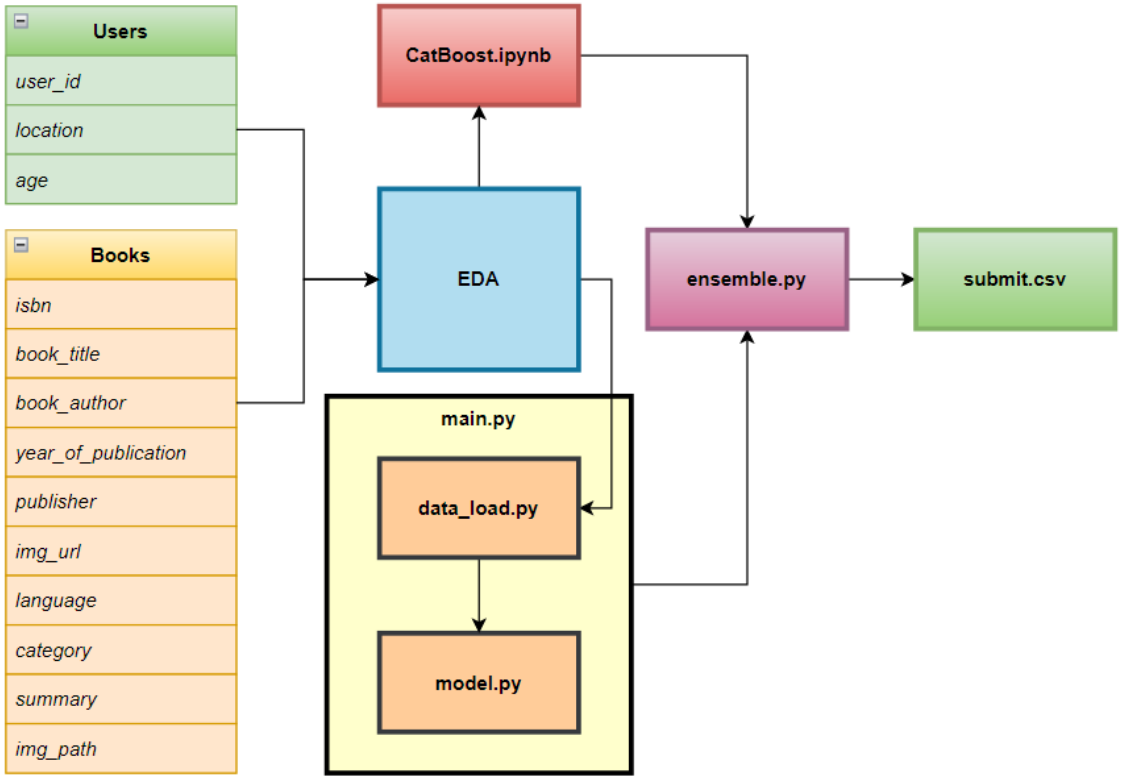
협업

- Github
- Slack
- Zoom
- Off-line

Tools

- Python
- Pytorch
- Weights & Biases + Sweep
- Optuna

▼ 프로젝트 구조 및 사용 데이터셋의 구조도



1-2. 프로젝트 팀 구성 및 역할

- **T4040_김성연(팀장):**
  - 팀의 방향성 설정 및 아이디어 제공.
  - 데이터 EDA 진행.
  - DeepCoNN 모델 실험해보기.
- **T4056_김찬호(팀원):**
  - 노션을 활용해 팀원의 진행과정 기록하기.
  - TabNet 모델 실험해보기.
- **T4079_박문순(팀원):**
  - 깃허브 전반적인 버전 관리.
  - FM, FFM 모델 실험해보기.
- **T4096_배성수(팀원):**

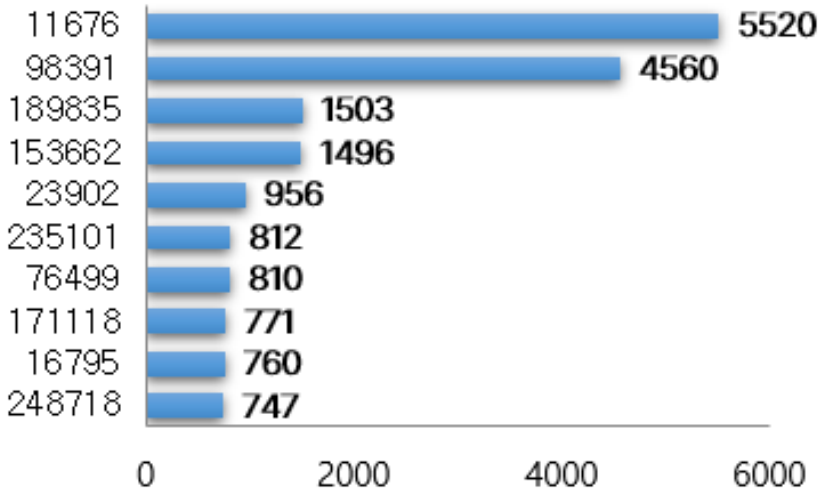
    - FFM+DCN 모델 구현하고 고도화 하기.
    - Cold-Start 관련 분석해보기.
- **T4171_이지훈(팀원):**
    - K-fold 및 Wandb, Optuna 등을 사용해 모델 고도화 하기.
    - 부스팅 기반 모델 실험해보기.

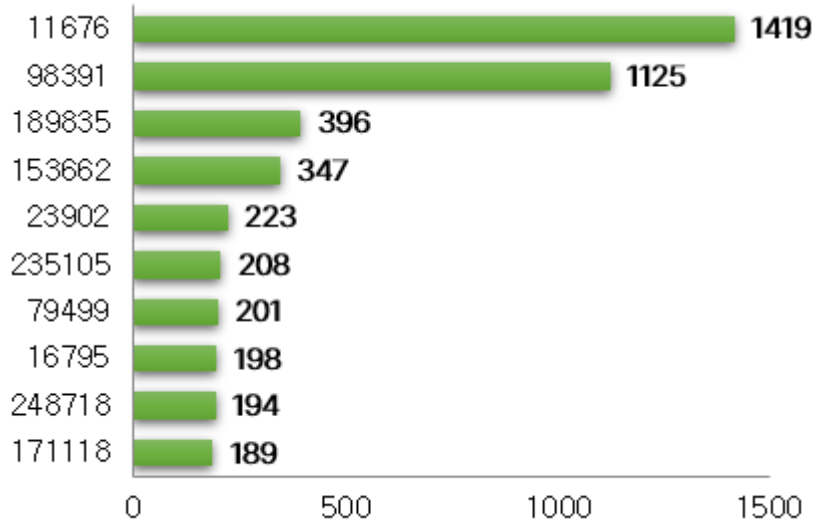
1-3. 프로젝트 수행 절차 및 방법

▼ EDA

평점 데이터 분석

- 유저 데이터는 68069개, 책 데이터는 149570개 존재하고 평점 기록은 306795개 존재 ⇒ **99.99699%만큼 희소행렬 데이터**
    - 유저와 책 데이터 수 대비 평점 기록 데이터가 상당히 많이 부족
    - MF 모델이나 딥러닝 모델 적용에 어려움이 있을 것으로 생각됨
- 최다 독서 유저 리스트를 보았을 때, **train과 test 데이터 구성이 거의 동일**한 것으로 생각됨
    - train 데이터의 일부를 valid 데이터로 사용한다면 test 데이터 환경과 유사할 것
    - train 데이터와 test 데이터 내 책을 많이 읽은 Top 10 User가 동일함





- **레이팅의 평균 값(약 7)**을 예측에 사용했을 때 RMSE 값은 **2.4332**
  - 추천 시스템 모델 평가의 기준점

유저 데이터 분석

- 전체 유저 데이터 68069개에서 **test 유저 데이터 26167개 중 train 데이터에 존재하지 않는 유저 데이터는 8266개**
  - **cold start 문제**가 예측됨
- **location 변수**를 간단히 전처리하여 **city, state, country**로 분할

```
user_id            68092
age                   91
location_city      11995
location_state      1324
```

```
location_country      269
dtype: int64
```

- **나이 그룹 별 평점 평균에 유의미한 차이** 존재

```
0 ~ 10 까지 출판된 책 623 개의 평균 평점은 6.7335 입니다.
10 ~ 20 까지 출판된 책 12034 개의 평균 평점은 7.1113 입니다.
20 ~ 30 까지 출판된 책 53141 개의 평균 평점은 7.2486 입니다.
30 ~ 40 까지 출판된 책 59784 개의 평균 평점은 7.0632 입니다.
40 ~ 50 까지 출판된 책 38489 개의 평균 평점은 7.2613 입니다.
50 ~ 100 까지 출판된 책 33299 개의 평균 평점은 7.3867 입니다.
NULL 나이 책은 92662개로 평균 평점은 6.7354 입니다.
```

책 데이터 분석

- **language, category, summary 변수에 결측값이 약 40%** 정도 존재

```
 0   isbn                 149570 non-null  object
 1   book_title           149570 non-null  object
 2   book_author          149570 non-null  object
 3   year_of_publication  149570 non-null  float64
 4   publisher            149570 non-null  object
 5   img_url              149570 non-null  object
 6   language             82343 non-null   object
 7   category             80719 non-null   object
 8   summary              82343 non-null   object
 9   img_path             149570 non-null  object
dtypes: float64(1), object(9)
```

- **책 출간 연도 그룹 별 평균 평점에 유의미한 차이** 존재

```
1900 ~ 1970 까지 출판된 책 2469 개의 평균 평점은 7.5055 입니다.
1970 ~ 1980 까지 출판된 책 7208 개의 평균 평점은 7.2278 입니다.
1980 ~ 1990 까지 출판된 책 36409 개의 평균 평점은 7.0839 입니다.
1990 ~ 2000 까지 출판된 책 140308 개의 평균 평점은 7.0004 입니다.
2000 ~ 2020 까지 출판된 책 87570 개의 평균 평점은 7.1678 입니다.
```

- **언어 그룹 펼 평균 평점**에 **유의미한 차이** 존재

```
'en'로 작성된 책 182282개의 평점 평균은 7.089 입니다.
'de'로 작성된 책 2226개의 평점 평균은 6.663 입니다.
'es'로 작성된 책 1486개의 평점 평균은 6.918 입니다.
'fr'로 작성된 책 1175의 평점 평균은 7.12 입니다.
'it'로 작성된 책 296개의 평점 평균은 7.399 입니다.
책 작성 언어가 NULL인 책 119084개의 평점 평균은 7.049 입니다.
```

데이터 분포 분석

- **변수 별로 평점이 기록된 책의 개수**를 나타낸 **그래프**
- **특별히 많이 추출된 일부 값**이 존재

▼ 모델 탐색

FM

- 유저와 책의 Field들을 활용하기 위해 실험해본 모델
- 데이터 전처리 이후 나쁘지 않은 성능을 보임
- DeepCoNN에서 사용하는 item summary vector를 추가해 실험해 봤으나 기본 FM모델보다 성능이 좋지 않음
- 성능이 더 좋아지지 않아 포기하게 됨

DeepCoNN, CNN_FM, TabNet

- **DeepCoNN**
  - 책의 Summary을 BERT 모델을 사용하여 전처리 한 뒤 값을 적용한 모델

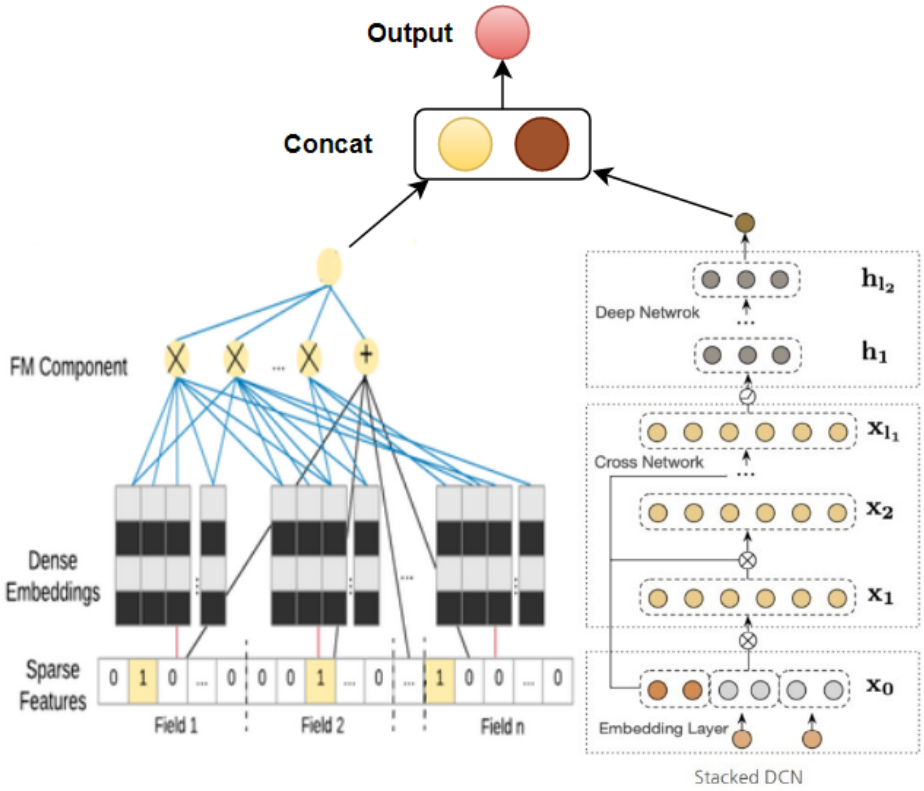
    - 유저의 Summary은 해당 유저가 읽은 책의 summary 길이 기준 상위 5개를 뽑아서 사용
    - Summary 자체에 결측값이 상당히 많고 유저의 Summary를 구하는 과정에서 Data Leakage가 일어나 OverFitting 가능성이 높음
    - 일반화 성능이 좋지 않아 사용하지 않음
- **CNN_FM**
    - 이미지 데이터를 활용하기 위해 실험해본 모델
    - 이미지 데이터 외에 context 데이터 등은 사용하지 않음
    - 성능이 좋지 않아 포기하게 됨
- **TabNet**
    - 트리 기반 부스팅 모델들이 주류인 정형 데이터 작업에서 딥러닝 모델의 성능을 끌어올린 모델
    - 트리 기반 부스팅 모델과 딥러닝 모델의 장점을 살린 모델
    - 여러 차례 실험을 진행했지만, 메인 모델인 CatBoost에 비해 낮은 성능을 보여주고 앙상블 결과도 좋지 않아 포기하게 됨

CatBoost & Boosting Models(XGBM,LGBM)

- **XGBM, LGBM**
    - 두 부스팅 모델의 성능이 CATB보다 좋지 않았음
    - Dataset이 명목형 데이터가 약 90% 차지했기 때문에, 인코딩 과정에서 효과를 보지 못했다고 판단
- **CATB**
    - Ordered Target Encoding
    - Categorical feature combination

FFM+DCN Hybrid Model

- 성능을 더욱 높이기 위해서 CatBoost와 앙상블 시너지를 내는 모델 탐색
- Tabular Data: Deep Learning is Not All You Need 논문으로부터 트리 기반의 Boosting모델과 딥러닝 모델을 앙상블한 결과가 더 좋은 성능을 낸다는 인사이트를 얻음



- 앞선 EDA를 통해 각 Field별 그룹에 따른 rating의 유의미한 차이가 존재하다는 사실을 알 수 있음
- 모든 Field 데이터를 활용할 수 있는 FFM 모델과 유저와 책의 Explicit 상호작용과 Implicit 상호작용을 학습하는 DCN 모델을 채택, 결합하여 Hybrid Model 설계
  - FFM Component에는 user_id와 isbn을 포함한 모든 Field들을, DCN Component에는 오직 user_id와 isbn만 입력으로 사용
  - FFM Component의 출력과 DCN component의 출력을 concatenate 한 뒤 Linear Layer를 통해 최종 예측평점을 출력
- Hyperparameter tuning 후 딥러닝 모델 중 가장 좋은 성능을 보여 채택하게 됨

▼ 모델 고도화

튜닝

- **Optuna**

    - 모델 생성 과정에서 적용하는 다양한 하이퍼 파라미터를 최적화 시켜주는 tool
    - Best 파라미터를 찾아서 자동으로 학습에 적용하는 부분이 있어 사람이 결과를 보고 하나하나 설정해주지 않아도 됨
    - 하이퍼 파라미터 최적화를 통해 모델이 성능이 향상되고, 학습도 원활하게 진행됨
    - 특히 하이퍼 파라미터 설정에 민감한 모델에서 좋은 성능을 보여줌
    - 교차 검증과 함께 사용하여 각 검증 시행마다 하이퍼 파라미터를 최적화 시켜 모델의 성능을 조금 더 고도화할 수 있음
- **Wandb(sweep)**
    - 머신 러닝 실험을 관찰하는 tool인 Wegihts & Biases
    - Sweep은 wandb의 부가 기능 중 하나로 실험을 통해 각 하이퍼 파라미터간의 관계와 모델 성능에 미치는 영향을 관찰할 수 있음
    - Optuna와 같은 Best 파라미터를 자동으로 적용해주는 기능은 없어 실험 결과를 보고 사람이 하이퍼 파라미터를 하나하나 조정해줘야 함

교차 검증

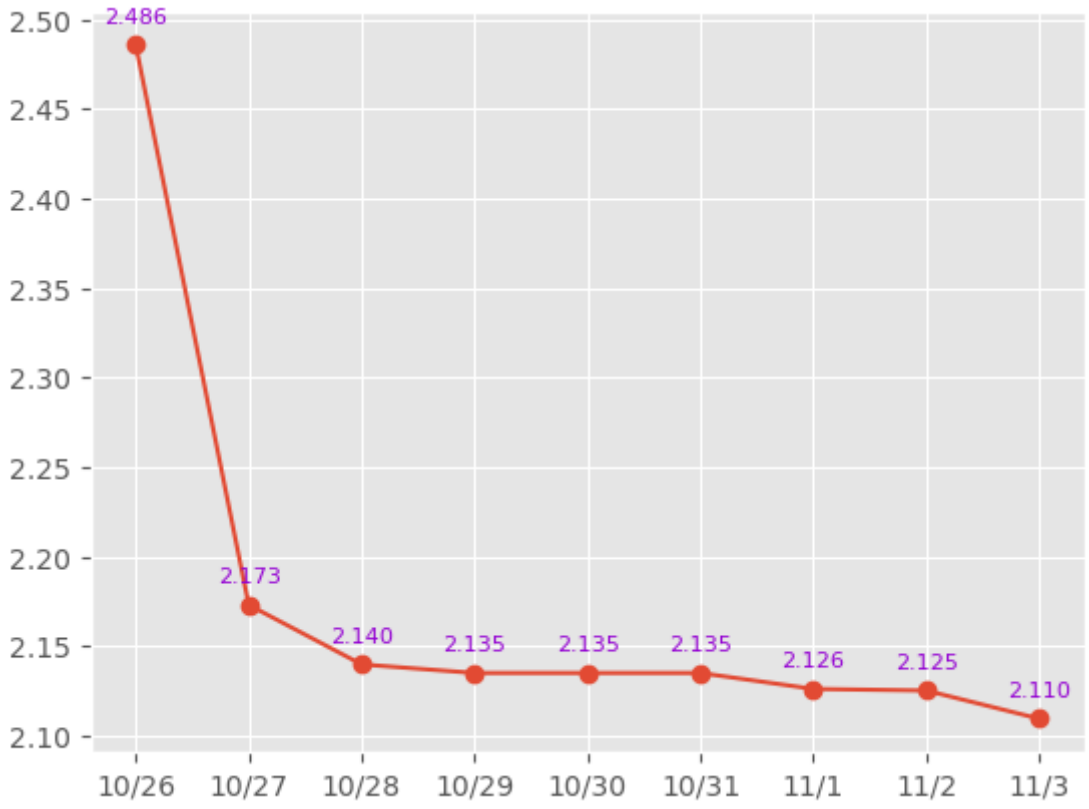
- **Stratified K-fold**
    - rating 변수의 분포가 불균형하기 때문에 fold 별로 동일한 rating 변수의 분포가 나뉘는 Stratified K-fold 사용.
    - 각 폴드에 Optuna를 적용해 K개의 모델을 튜닝하였고 배깅과 유사한 효과가 발휘되어 성능이 향상됨.

앙상블

- **Catboost & FFDCN**
    - Tabular Data: Deep Learning is Not All You Need 논문으로부터 트리 기반의 Boosting모델과 딥러닝 모델을 앙상블한 결과가 더 좋은 성능을 낸다는 사실을 알 수 있음
    - Tree계열의 Boosting 모델은 내삽, 안쪽으로 예측하려는 경향이 강함
    - 딥러닝 계열의 모델은 외삽, 바깥쪽으로 예측하려는 경향이 있음
    - 그래서 이 둘을 앙상블 하면 성능이 더 좋아질 수 있음

1-4. 프로젝트 수행 결과

🏆 **WINNER! WINNER! CHICKEN DINNER!**



- 10/26 - **DeepCoNN 기본 모델**을 적용하여 **2.4856**의 성능을 얻음
- 10/27 - **CatBoost 기본 모델**을 적용하여 **2.1732**의 성능을 얻음
- 10/28 - **CatBoost에 간단히 전처리한 데이터를 입력**하여 **2.14**의 성능을 얻음
- 10/29 - **CatBoost를 Optuna를 통해 하이퍼 파라미터 튜닝하고, 계층 교차 검증(Stratified K-fold)을 적용**하여 **2.1352**의 성능을 얻음 **(K=5)**
- 11/1 - **CatBoost에 각 교차 검증 시행(fold)마다 Optuna를 통해 하이퍼 파라미터를 튜닝**하여 **2.1263**의 성능을 얻음 **(K=5)**
- 11/1 - **FFM 모델과 DCN 모델의 구조를 합쳐 FFN+DCN 모델을 구현**하여 **2.4152**의 성능을 얻음

- 11/2 - **FFM+DCN 모델을 Optuna를 통해 하이퍼 파라미터를 튜닝**하여 **2.1783**의 성능을 얻음
- 11/3 - **CatBoost에 교차 검증 시행 횟수를 10으로 늘려** **2.1220**의 성능을 얻음 **(K=10)**
- 11/3 - **FFM+DCN 모델에 Optuna로 튜닝된 하이퍼 파라미터를 고정하고, 계층 교차 검증(Stratified K-fold)을 적용**하여 **2.1289**의 성능을 얻음 **(K=5)**
- 11/3 - **CatBoost와 FFM+DCN 모델의 결과를 앙상블**해서 **2.1095**의 성능을 얻음

1-5. 자체 평가 의견

▼ 잘한 점

- 데이터 EDA를 꼼꼼히 진행했습니다.
- 리더보드 스코어에 연연하지 않고 나름의 근거를 가진채 여러가지 모델을 실험했습니다.
- 실전 대회에서 활용도 높은 다양한 모델 고도화 전략을 사용했습니다.

▼ 시도했으나 잘되지 않았던 점

- DeepCoNN, CNN_FM, TabNet 등 규모가 큰 딥러닝 모델을 여러 각도로 살펴봤지만 좋은 성능을 거두지 못했습니다.
- 서로의 코드를 리뷰하는 시간을 가져보려 했으나 잘 하지 못했습니다.
- 이미지, 텍스트와 같은 비정형 데이터를 활용해보려 시도했지만, 잘되지 않았습니다.

▼ 아쉬운 점

- Github master 브랜치를 잘 활용하지 못해 협업 효과를 크게 보지 못했습니다.

▼ 프로젝트를 통해 배운점

- EDA를 통해 데이터에 직관을 얻는 것이 중요하다는 것을 다시 느꼈습니다.
- 성능이 잘 나오는 부스팅 모델이나 딥러닝 모델을 다룰 수 있는 능력을 키웠습니다.