

# Book Rating Prediction

recsys-02 추천답갈비 팀

# 목차

01

소개

팀 소개  
프로젝트 소개  
진행 방식

02

프로젝트 진행

진행 일정  
EDA  
다양한 시도  
모델링

03

최종 결과물

최종 모델 선정  
양상블

04

회고

새롭게 배운 점  
아쉬웠던 점

# 01. 소개

팀 소개  
프로젝트 소개  
진행 방식

# 팀소개

추천 닭갈비 팀을 소개합니다!



강현구

EDA  
DeepFM, ResNet  
DeepFM



서동준

TextFM, TextDeepFM,  
LGBM



이도걸

WDN, DCN, NCF,  
Boosting



이수미

Add features,  
FM, FFM



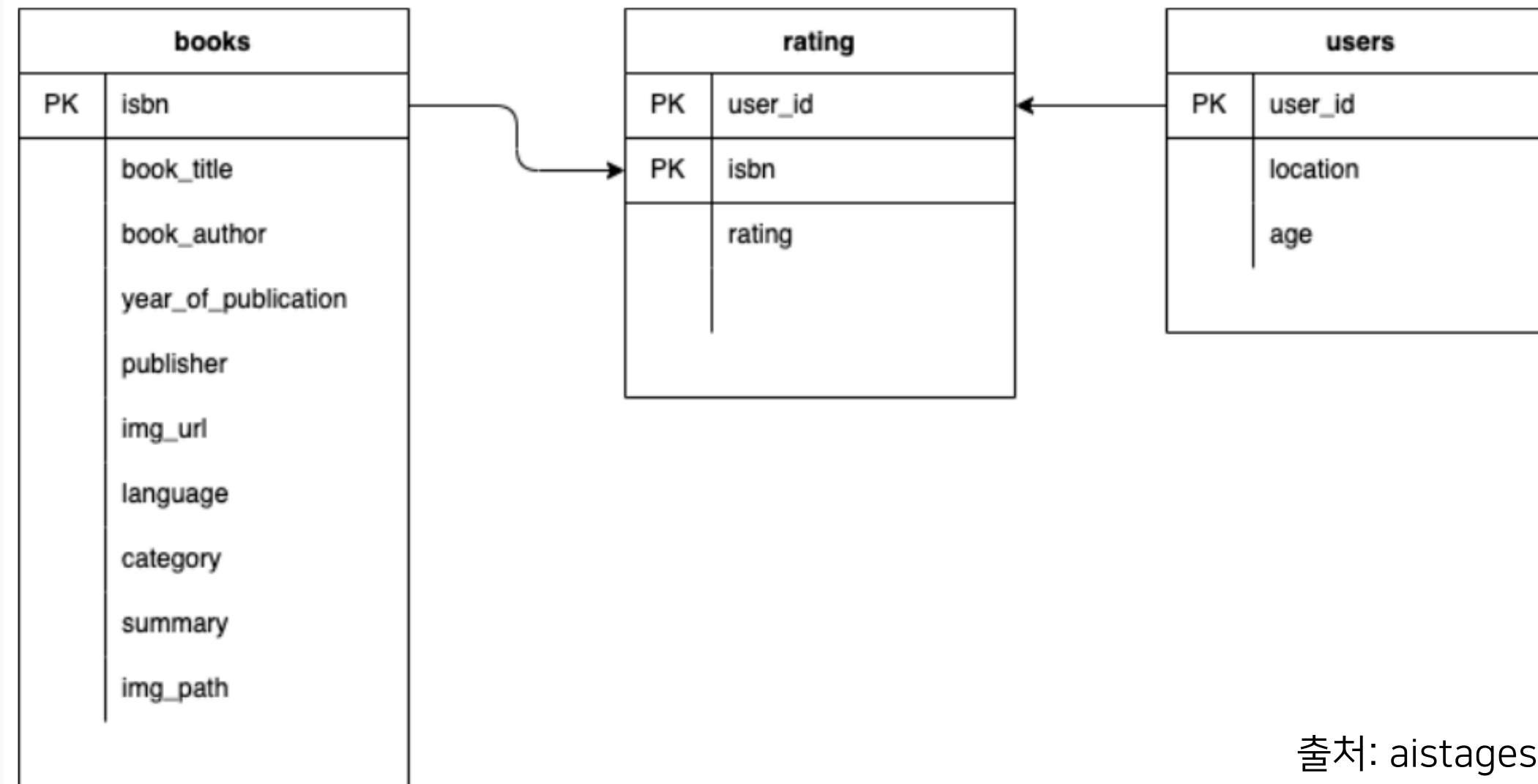
최윤희

Data Preprocessing,  
ImageFM, Image  
DeepFM, Boosting

# 프로젝트 소개

책, 유저, 평점 데이터를 가지고 소비자의 책 선호도(평점)을 예측하는 프로젝트

목표 : 강의에서 배운 추천 시스템 모델을 이해하고 활용해보자!



출처: aistages



## 협업 방식

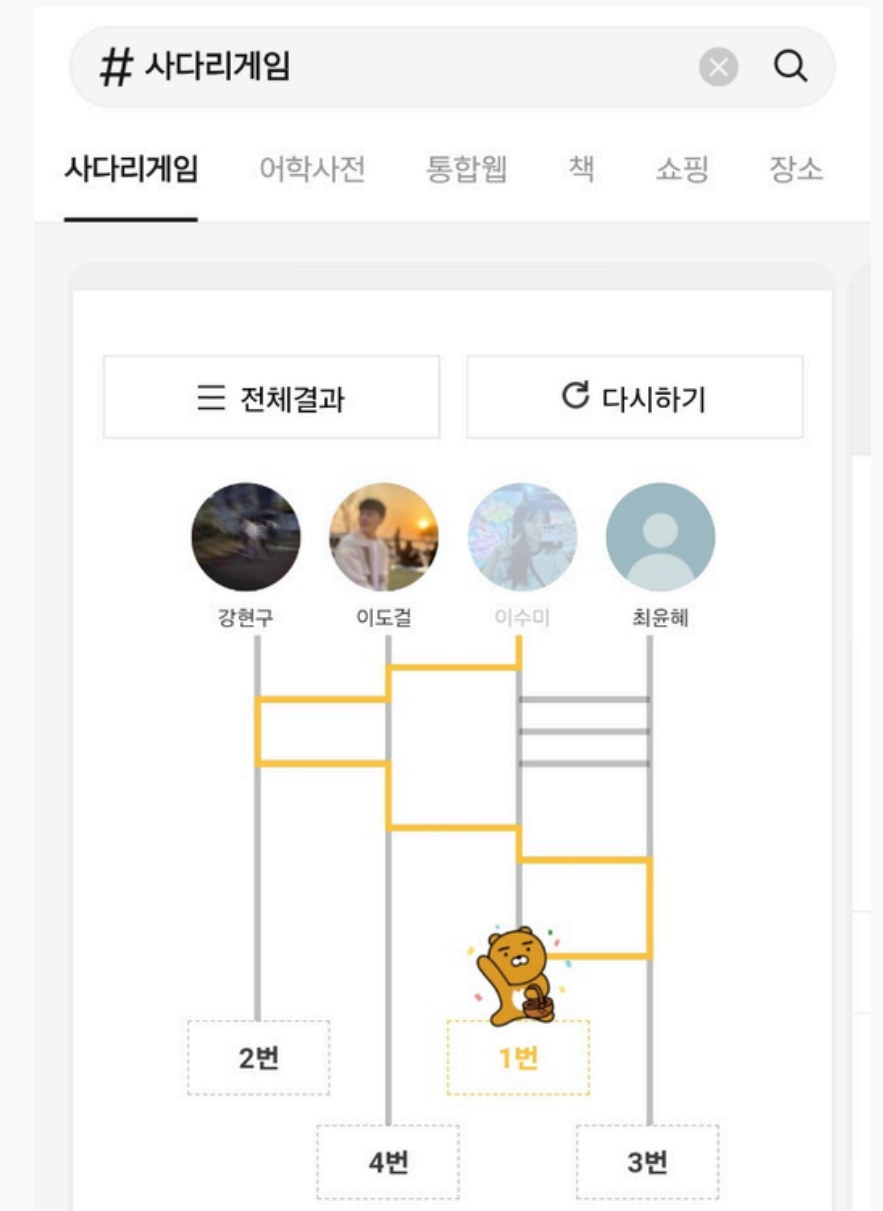
baseline 에 제공된 코드를 모두 이해하기에 시간이 부족하다고 생각

-> 모델을 나누자! -> 사다리 타기



모델	20 epochs 시간	baseline test RMSE
FM	1분 30초	4.4859
FFM	3분 33초	5.1079
DeepFM	1분 34초	2.9706
WDN	1분 34초	3.5141
DCN	1분 35초	2.8090
NCF	1분 34초	2.7160
Image FM	14분 13초	2.8592
Image DeepFM	14분 8초	2.7083
Text FM	5분 39초	3.5082
Text DeepFM	6분 00초	2.9607
ResNet DeepFM	25분 29초	2.7238

출처: aistages





# 협업 방식

깃허브, Wandb와 노션도 사용

## 1. Wandb



## 2. Notion









1월 5일	이수미(자연과학)	20241105_093739_FM	X	FM	3.63469	1.93156		dropout 적용하고 epoch (100)
1월 5일	seodongjoon	Text_FM	O	Text_FM	2.199	1.664	2.4312	
1월 6일	seodongjoon	Text_DeepFM	O	Text_DeepFM	2.228	1.714	2.3848	
1월 6일	Dogeol Lee	20241106_150003_WDN	O	WDN	1.7817	1.3062	2.2847	전처리 + 복잡도 낮춰서 0.2 + dropout 0.5 + lr:
1월 6일	최윤희	catboost_context_data	O	CatBoost	1.548336345		2.3896	수미 코드로 바꾸고 con 에 catboost 대강
1월 6일	Dogeol Lee	20241106_161055_DCN	O	DCN	1.97337	1.38389	2.3724	lr: 1e-3, embed_dim: 3 cross_layer_num: 3, m [32, 64, 32], dropout: 0
1월 6일	seodongjoon	lgbm with text_vector_pca	O	LightGBM	2.1959		2.1944	

## 3. Github

<input type="checkbox"/>	<input checked="" type="checkbox"/>	[020] 토픽 모델링	Question	Test	#20 by SooMiiii was closed 1 hour ago	1 of 3 tasks
<input type="checkbox"/>	<input checked="" type="checkbox"/>	[019] books 데이터 결측치 처리	EDA	Feature	#19 by yunhye Choi was closed 1 hour ago	3 tasks done
<input type="checkbox"/>	<input checked="" type="checkbox"/>	[017] Adversarial Validation 실험	Test		#17 by yunhye Choi was closed 1 hour ago	1 task
<input type="checkbox"/>	<input checked="" type="checkbox"/>	[016] 데이터 전처리 & 피쳐 추가 - context_data.py	Feature	Refactor	#16 by SooMiiii was closed 8 hours ago	1 of 2 tasks
<input type="checkbox"/>	<input checked="" type="checkbox"/>	[012] BERTopic / BERT 임베딩	Feature	Test	#12 by SooMiiii was closed 5 hours ago	2 tasks done
<input type="checkbox"/>	<input checked="" type="checkbox"/>	[010] FM / FFM 모델 고도화	Feature	Test	#10 by SooMiiii was closed 5 hours ago	4 of 5 tasks
<input type="checkbox"/>	<input checked="" type="checkbox"/>	[009] Image FM 고도화	Feature	Test	#9 by yunhye Choi was closed 1 hour ago	3 tasks
<input type="checkbox"/>	<input checked="" type="checkbox"/>	[008] Text FM 고도화	Feature	Test	#8 by seoo2001 was closed 1 hour ago	3 of 4 tasks
<input type="checkbox"/>	<input checked="" type="checkbox"/>	[005] 데이터 살펴보기	EDA		#5 by SooMiiii was closed 6 hours ago	4 tasks
<input type="checkbox"/>	<input checked="" type="checkbox"/>	[003] GitHub Issue, PR 레이블, 템플릿 설정	Setting		#3 by SooMiiii was closed 3 days ago	2 tasks done

# 협업 방식

## 소소한 자랑: 이슈 템플릿

9 labels	
 bug	버그 이슈
 docs	문서 작성 이슈
 EDA	탐색적 데이터 분석
 Feature	기능 구현 이슈
 Help	도와줘잉
 Question	질문
 Refactor	코드 리팩토링
 Setting	개발 환경 & GitHub 세팅
 Test	가설 검증 테스트

### Issue: EDA

탐색적 데이터 분석 If this doesn't look right, [choose a different type](#).



Add a title

#### 설명

하시려는 EDA에 대해 설명해주세요.

자세히 적을수록 좋습니다!

#### 작업할 내용

할 일을 체크박스 형태로 작성해주세요.

최대한 세분화 해서 적어주세요!

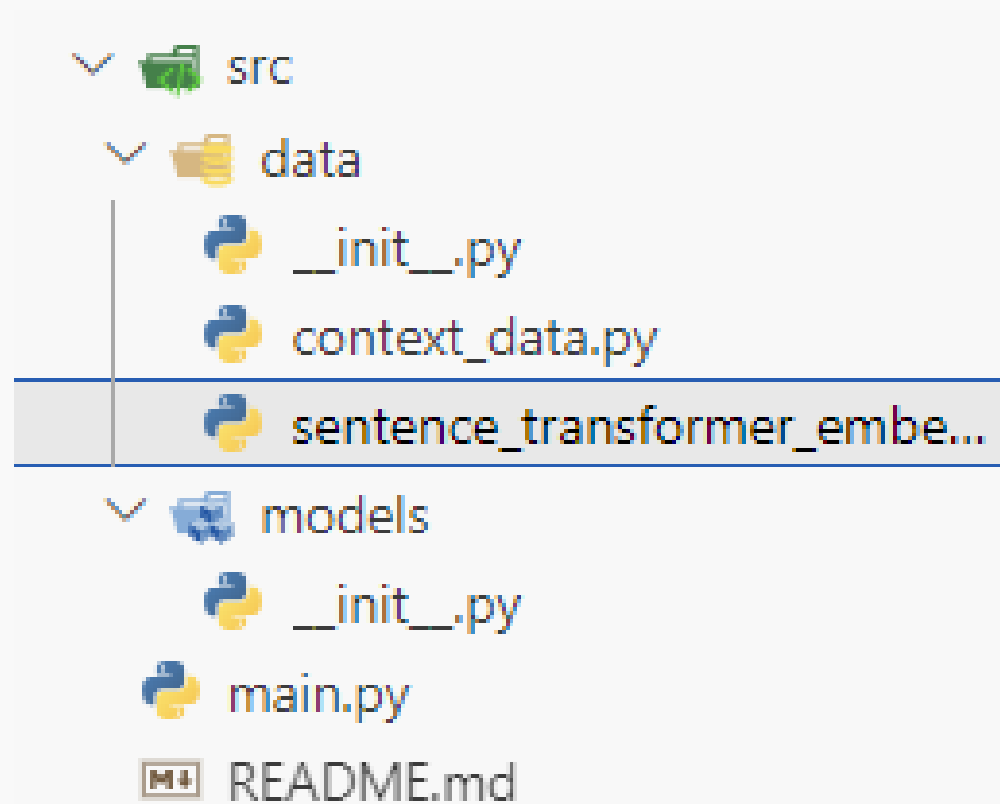
#### 참고 자료

참고 자료가 있다면 작성해 주세요.



# 협업 방식

필요한 기능은 모듈화하여 코드 추가



```
def main(books: pd.DataFrame, target_dim: int, output_path: str) -> torch.Tensor:
    """
    책 정보에서 문장 임베딩을 생성한 후 MLP 모델로 차원을 축소하여 `.numpy` 파일로 저장

    Args:
        books (pd.DataFrame): 책 정보를 포함한 데이터프레임
        target_dim (int): 축소하고자 하는 임베딩 차원
        output_path (str): 결과를 저장할 `.numpy` 파일 경로

    Returns:
        torch.Tensor: 축소된 임베딩 결과
    """
    # 제목, 카테고리, 요약은 하나의 문자열로 합쳐 임베딩 입력 준비
    docs = (books['book_title'] + " " + books['category'] + " " + books['summary']).tolist()

    # SentenceTransformer 모델을 사용해 문장 임베딩 생성
    sentence_model = SentenceTransformer('all-MiniLM-L6-v2')
    embeddings = sentence_model.encode(docs)

    # 차원 축소를 위한 MLP 모델 정의
    class DimensionReducer(nn.Module):
        """
        MLP 모델을 통해 차원을 축소하는 클래스

        Attributes:
            fc1 (nn.Linear): 첫 번째 완전 연결층
            relu (nn.ReLU): ReLU 활성화 함수 "relu": Unknown word.
            fc2 (nn.Linear): 두 번째 완전 연결층
        """
        def __init__(self):
            super(DimensionReducer, self).__init__()
            self.fc1 = nn.Linear(384, 128)
            self.relu = nn.ReLU() "relu": Unknown word.
            self.fc2 = nn.Linear(128, target_dim)
```

## 02. 프로젝트 진행

진행 일정

EDA

다양한 시도

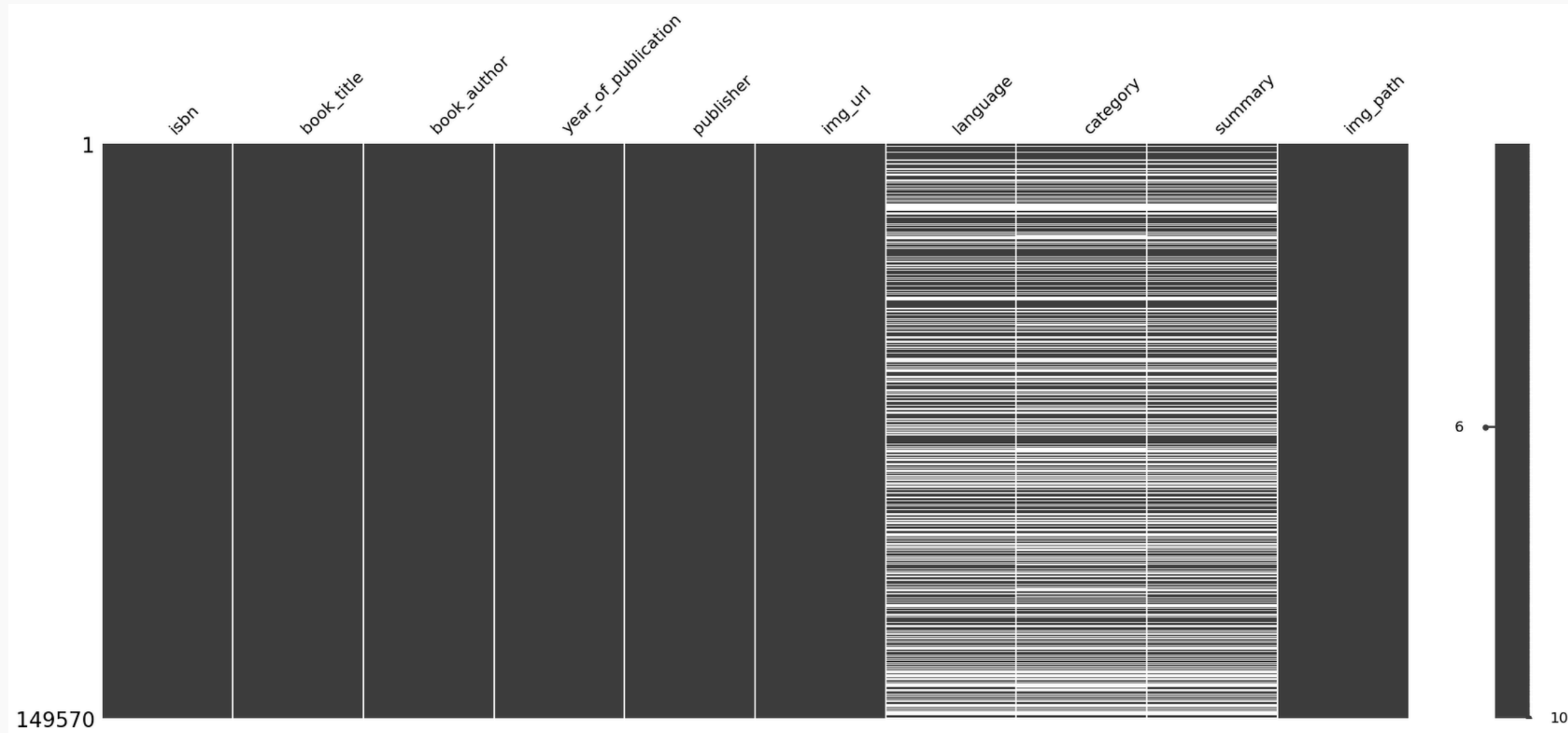
모델링

# 진행 일정

날짜	10/30	11/02	11/04	11/07
데이터분석	<div>EDA &amp; Data Exploration</div> <div>Preprocessing</div> <div>Missing value imputation</div>	<div>Feature Engineering</div>	<div>Additional Data Processing</div>	
모델링		<div>TextFM &amp; Text DeepFM</div> <div>FM &amp; FFM</div> <div>DeepFM &amp; ResNet DeepFM</div> <div>WDN &amp; DCN &amp; NCF</div> <div>Image FM &amp; Image DeepFM</div>	<div>Boosting</div>	<div>Hyperparameter Tuning</div> <div>Ensemble Modeling</div>

# EDA: books

language, category, summary에 결측치가 아주 많음.



# EDA: books

language

```
print(books['language'].isna().sum()/len(books))  
## 44%가 nan
```

0.449468476298723

44%가 nan

```
print(len(books[books['language']=='en'])/len(books['language'].dropna()))  
## nan 제외 95%가 영어다.
```

0.957251982560752

영어가 약 96% 차지 -> 그렇다고 결측치를 다 영어로 채우는게 맞을까?

# EDA: books

language -> isbn 이용

## 2. ISBN 원리

- 모든 책은 세상에 단 한가지만 존재해야 한다.
- 저자가 똑같은 내용의 책을 다시 출간한다 해도 바뀌지 않는다.
- 번호는 10개와 13개로 구성된다. (2007년 1월 1일 이후로 13개만 출간됨)

### 2.1 ISBN 10자리

ISBN-10

8	9	-	5	6	7	4	-	1	8	9	-	1
---	---	---	---	---	---	---	---	---	---	---	---	---

- 1) 89 -> 출판 국가 또는 언어 번호
- 2) 5674 -> 출판사 고유번호
- 3) 189 -> 출판사에서 책에 붙인 번호(임의로 붙임)
- 4) 1 ->

isbn의 앞 두자리가 출판 국가 또는 언어를 나타내는 기호

- 따라서 앞 두자리 - language가 1:1로 매핑되는 코드북 제작
- 코드북 기준으로 결측치 대체
- 결측치 : 67227개 -> 105개

출처 : <https://securityspecialist.tistory.com/86>



# EDA: books

## 카테고리

```
print(books['category'].isna().sum()/len(books))  
# 46%가 nan
```

✓ 0.0s

0.4603262686367587

## 46%가 nan

```
len(set(cleaned_books))
```

✓ 0.0s

4292

총 카테고리 4292개로 세분화 되어있음

```
# 'computer'가 포함된 요소 필터링  
print(len(set(cleaned_books[cleaned_books.str.contains('computer', case=False, na=False)])))  
set(cleaned_books[cleaned_books.str.contains('computer', case=False, na=False)])  
# 컴퓨터 관련 카테고리만 51개
```

✓ 0.0s

51

```
{'ASP (Computer network protocol)',  
'Assembler language (Computer program language)',  
'Atari 400 (Computer)',  
'BASIC (Computer program language)',  
'Business Computer programs',  
'C (Computer program language)',  
'C (Computer program language).',  
'C# (Computer program language)',  
'Computer Communication Networks',  
'Computer Graphics',  
'Computer animation',  
'Computer capacity',  
'Computer crimes',  
'Computer engineering',  
'Computer graphics',  
'Computer graphics.',  
'Computer industry',  
'Computer input-output equipment',  
'Computer network architectures',  
'Computer network resources',  
'Computer networks',  
'Computer programming',  
'Computer science',  
'Computer security',  
'Computer software',
```

“컴퓨터”가 들어가는 카테고리만 51개임.

## EDA: books

카테고리 -> 유사한 카테고리는 하나로 통합해서 줄여보자!

- Sentence Bert 임베딩
- AgglomerativeClustering (계층적 클러스터링)
- 결합되지 않은 카테고리는 'etc' 카테고리로 통합
- 카테고리: 4292개 -> 654개

```
books[books.category_clustered == 'fiction']['category'].unique()
```

✓ 0.0s

```
array(['fiction', 'fantasy fiction', 'fantastic fiction.',  
      'historical fiction', 'suspense fiction', 'yiddish fiction',  
      'suspence fiction', 'didactic fiction', 'fantastic fiction',  
      'fantasy literature', 'experimental fiction', 'samoan fiction',  
      '20th century general fiction', 'mystery fiction',  
      'childrens fantasy fiction', 'duitse fiksie', 'galician fiction'],  
      dtype=object)
```

유사한 카테고리가 잘 묶인 것 확인!

# EDA: books

## 카테고리

동일 저자, 제목이면서 출판사만 다른 데이터 존재

	book_title	book_author	publisher	category
55931	The Secret Garden	Frances Hodgson Burnett	Dell Publishing	children with disabilities
92417	The Secret Garden	Frances Hodgson Burnett	Landoll	children with disabilities
80446	The Secret Garden	Frances Hodgson Burnett	Scholastic	courage
5098	The Secret Garden	Frances Hodgson Burnett	Tor Books	fiction
59876	The Secret Garden	Frances Hodgson Burnett	Signet Classics	gardens
13601	The Secret Garden	Frances Hodgson Burnett	HarperTrophy	juvenile fiction
20350	The Secret Garden	Frances Hodgson Burnett	Laure Leaf	juvenile fiction
22930	The Secret Garden	Frances Hodgson Burnett	David R. Godine Publisher	juvenile fiction
51414	The Secret Garden	Frances Hodgson Burnett	Scholastic	juvenile fiction
88361	The Secret Garden	Frances Hodgson Burnett	Landoll	juvenile fiction
96003	The Secret Garden	Frances Hodgson Burnett	HarperCollins	juvenile fiction
59944	The Secret Garden	Frances Hodgson Burnett	Tor Books (Mm)	NaN
114426	The Secret Garden	Frances Hodgson Burnett	Signet Classics	NaN
122896	The Secret Garden	Frances Hodgson Burnett	Signet Book	NaN

- 동일 저자, 제목 중 최빈 카테고리로 결측치 대체
- 나머지 na는 동일 저자의 최빈 카테고리로 대체
- 결측치: 68,851개 -> 27,203개

## EDA: users

location: 주소 자체가 잘못 입력된 경우 다수 존재

1. 동일 country, city인데 state가 다른 경우

	location_country	location_city	location_state	count
0	united kingdom	london	england	415
1	united kingdom	london	london	53
2	united kingdom	london	uk	6
3	united kingdom	london	middlesex	4
4	united kingdom	london	greater london	3
5	united kingdom	london	lazio	2
6	united kingdom	london	brixton	1
7	united kingdom	london	battersea	1
8	united kingdom	london	alaska	1
9	united kingdom	london	alabama	1
10	united kingdom	london	bayern	1
11	united kingdom	london	foreign	1
12	united kingdom	london	city	1
13	united kingdom	london	forest gate	1

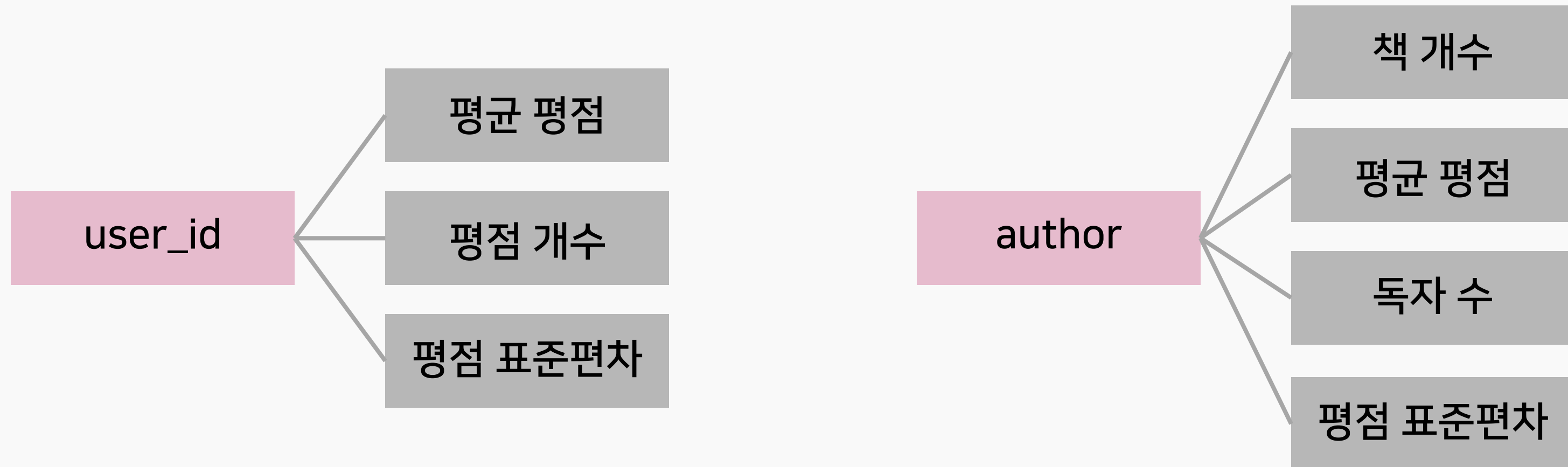
2. 동일 state, city인데 country가 다른 경우

	location_country	location_city	location_state	count
0	usa	phoenix	arizona	122
1	maracopa	phoenix	arizona	1
2	maricopa	phoenix	arizona	1
3	ysa	phoenix	arizona	1

->가장 높은 빈도의 조합으로 수정

## 추가 변수

데이터의 특성을 더 잘 반영하고자 `user_id`와 `author`를 기준으로 총 7개의 변수를 추가



변수의 특성에 따라 없는 값이면 평균, nan, 0으로 대체

## 추가 변수

간단한 부스팅 모델 적합 시 변수 중요도

- user\_avg\_rating, author\_avg\_rating이 유독 높음
- rating을 평균 낸 변수라서 validation에서 data leakage 발생한 것으로 보임
- 모델이 해당 변수에 너무 의존하도록 학습되므로 리더보드에서의 성능이 낮게 나옴  
-> 변수 제외

	score
user_avg_rating	45.107889
author_avg_rating	29.416860
author_unique_readers	5.417257
user_id	4.977736
book_author	4.605251
rating_count	4.063092
author_book_count	1.617444
book_title	0.899547
publisher	0.777371
location_city	0.659243
location_state	0.588469
category	0.406428
publication_range	0.358757
age_range	0.287544
location_country	0.281867
language	0.276576
isbn	0.258670



# 사용 변수 목록

user

'user\_id', 'age\_range',  
'location\_country', 'location\_state', 'location\_city'

books

'isbn', 'book\_title', 'book\_title\_len', 'book\_author',  
'language', 'publication\_range', 'publisher', 'category'

target

'rating'

추가

"rating\_count", 'user\_rating\_variance', 'author\_book\_count',  
'author\_unique\_readers', 'author\_rating\_variance'

※ 모델마다 다를 수 있음.

## 추가) Adversarial Validation 실험

train 데이터와 valid/test 간의 피쳐 분포 차이가 존재하는지 확인

각 변수별로 모델 학습하여 train, valid, test 예측

1.train, test셋을 각각 0,1로 target

2.각 변수별로 lgbm 모델 적합

3.target 정확도 계산

4.0.75 이상인 변수 제외

-> 0.5 이상의 정확도를 보이는 변수가 있긴 하지만  
모두 0.6 이하로 엄청 높진 않음

1.train, test 예측

	adv_score	feature
0	0.590183	rating_count
1	0.584625	user_avg_rating
2	0.578559	author_unique_readers
3	0.578528	author_book_count
4	0.576913	author_avg_rating
5	0.499834	language
6	0.497827	location_country
7	0.497684	age_range
8	0.497627	location_state
9	0.497359	publication_range
10	0.495376	user_id
11	0.495371	category
12	0.495369	location_city
13	0.491749	book_title
14	0.491441	book_author
15	0.491188	publisher
16	0.488370	isbn

2. train, valid 예측

	adv_score	feature
0	0.587997	rating_count
1	0.582553	user_avg_rating
2	0.578643	author_unique_readers
3	0.578248	author_book_count
4	0.575700	author_avg_rating
5	0.500522	language
6	0.499649	age_range
7	0.497840	location_country
8	0.497676	publication_range
9	0.497412	location_city
10	0.496612	category
11	0.496554	location_state
12	0.494765	user_id
13	0.492156	book_title
14	0.491323	publisher
15	0.490719	book_author
16	0.489356	isbn

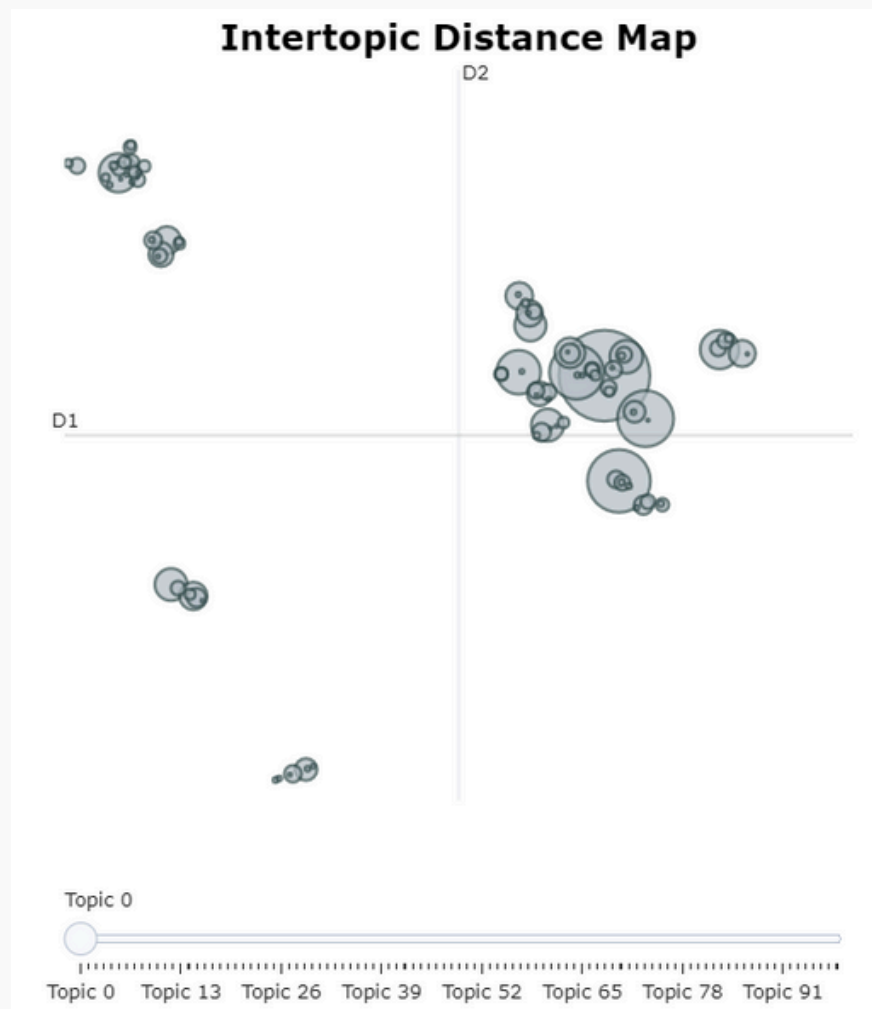
## 추가) 카테고리 통일 노력

BERTopic: BERT 기반의 토픽 모델링 구현체

category와 summary에 nan이 많음

-> 제목 + 카테고리 + 요약은 하나의 list로 만들어서 토픽으로 임베딩

-> 결과: -1(이상치) 가 너무 많이 나와서 사용하지 않음



45% 가 -1에 해당

100개 토픽 모델링  
시각화 결과

```
model.get_topic_info()
```

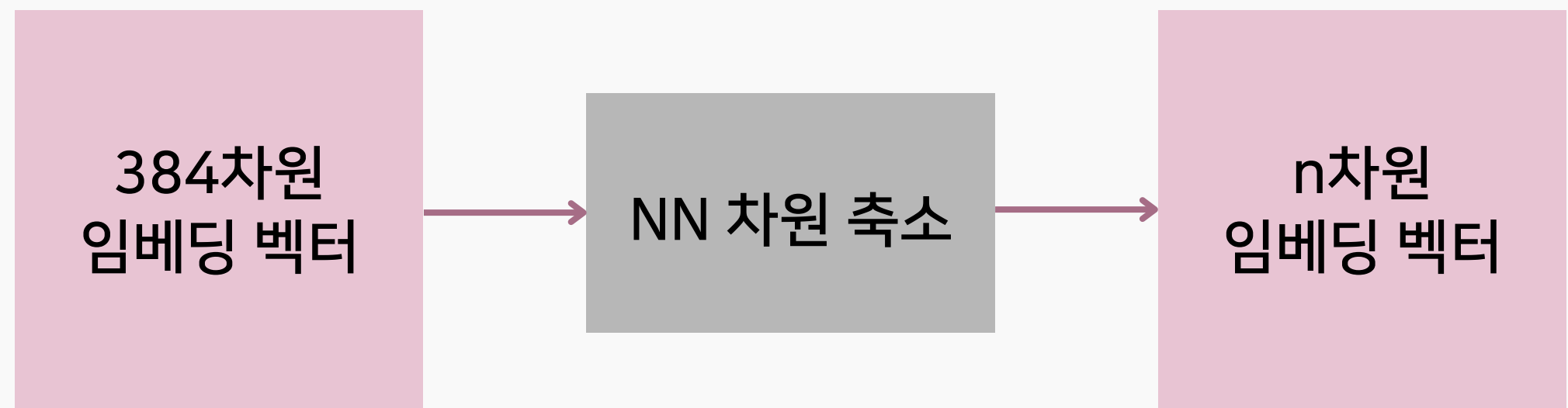
	Topic	Count	
	0	-1	68211
	1	0	15627
	2	1	7452
	3	2	5945
	4	3	5542
	...	...	...
	95	94	13
	96	95	11

## 추가) 카테고리 통일 노력

SentenceTransformer embedding

1. Sentence-BERT의 all-MiniLM-L6-v2 모델을 활용해서 title 혹은 title+category+summary 를 수치벡터로 변환
2. 384 차원의 임베딩 벡터를 NN 모듈을 통해 n차원으로 축소
3. book 데이터에 추가

유의미한 성능 향상을 보이지는 않음.



# Modeling

# FM, FFM, NCF, DCN, WND, DeepFM

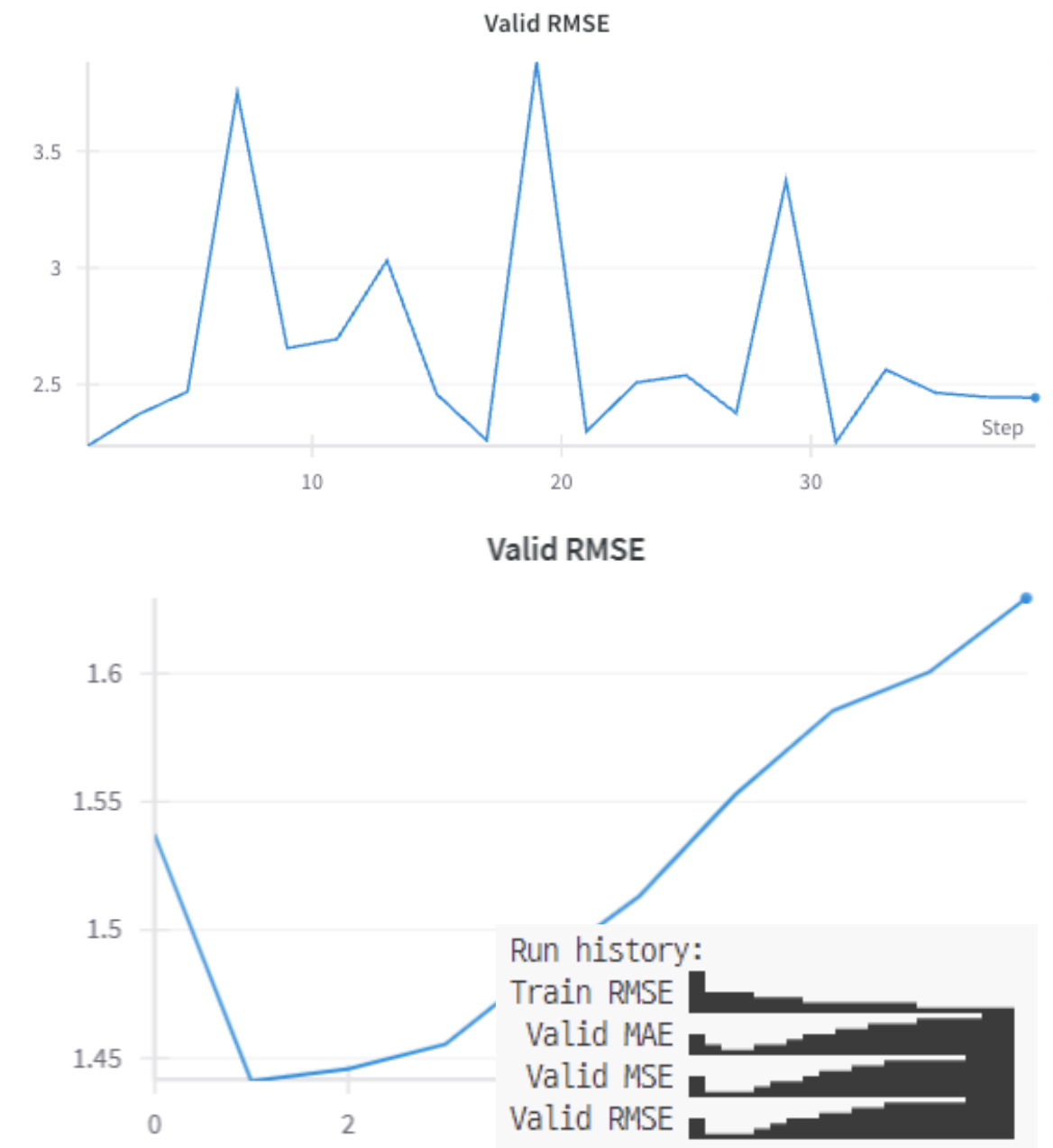
FM, FFM, NCF, DCN, WDN, DeepFM 등 다양한 추천 시스템 구현 시도  
각 모델의 특성과 학습 패턴을 심층적으로 분석

## 대부분의 모델에서 불안정한 학습 곡선 관찰

Valid RMSE의 지속적 증가와 명확한 수렴점을 찾지 못함

과적합 경향이 두드러져 일반화 성능 저하 확인

상대적으로 안정적인 성능을 보인 FM과 DeepFM을 앙상블용 모델로 채택



# Modeling

## Image FM, Image Deep FM, ResNet DeepFM

도서 커버 이미지의 시각적 특성을 추천 시스템에 접목하고자 시도

데이터 전처리 & 모델 학습에 긴 시간 소요

성능 향상폭이 투입 자원 대비 드라마틱하지 않음

프로젝트 시간 제약을 고려하여 비용 효율적인 접근법 모색

초기 실험 결과를 바탕으로 텍스트 기반 모델과 기존 협업 필터링 모델 최적화에 역량 집중

## LGBM, XGB

일반적인 Boosting 모델에서 어떤 성능을 보이는지, 앙상블에 사용하기 위해 실험

준수한 성능을 보였으나 최종 모델에서 사용한 Catboost의 성능보다 좋지 못했음

최근 업데이트로 사용 가능해진 category 특성을 적용해서 학습했으나 이 또한 Catboost보다 성능 하락



# Text\_DeepFM

## 1. Text Vector

AutoTokenizer의 pretrained 모델 변경

'bert-base-uncased' 모델로 text vector를 생성하는 것은 책의 제목과 요약 문장의 의미를 효과적으로 담아내기 어려움  
sentence-transformer(SBERT) 기반 'sentence-transformers/all-MiniLM-L6-v2' 모델로 변경

## 2. 모델 구조

user, item 임베딩 파라미터 분리

user\_text\_vector와 item\_text\_vector의 임베딩 파라미터를 분리하여 개별적으로 학습 가능

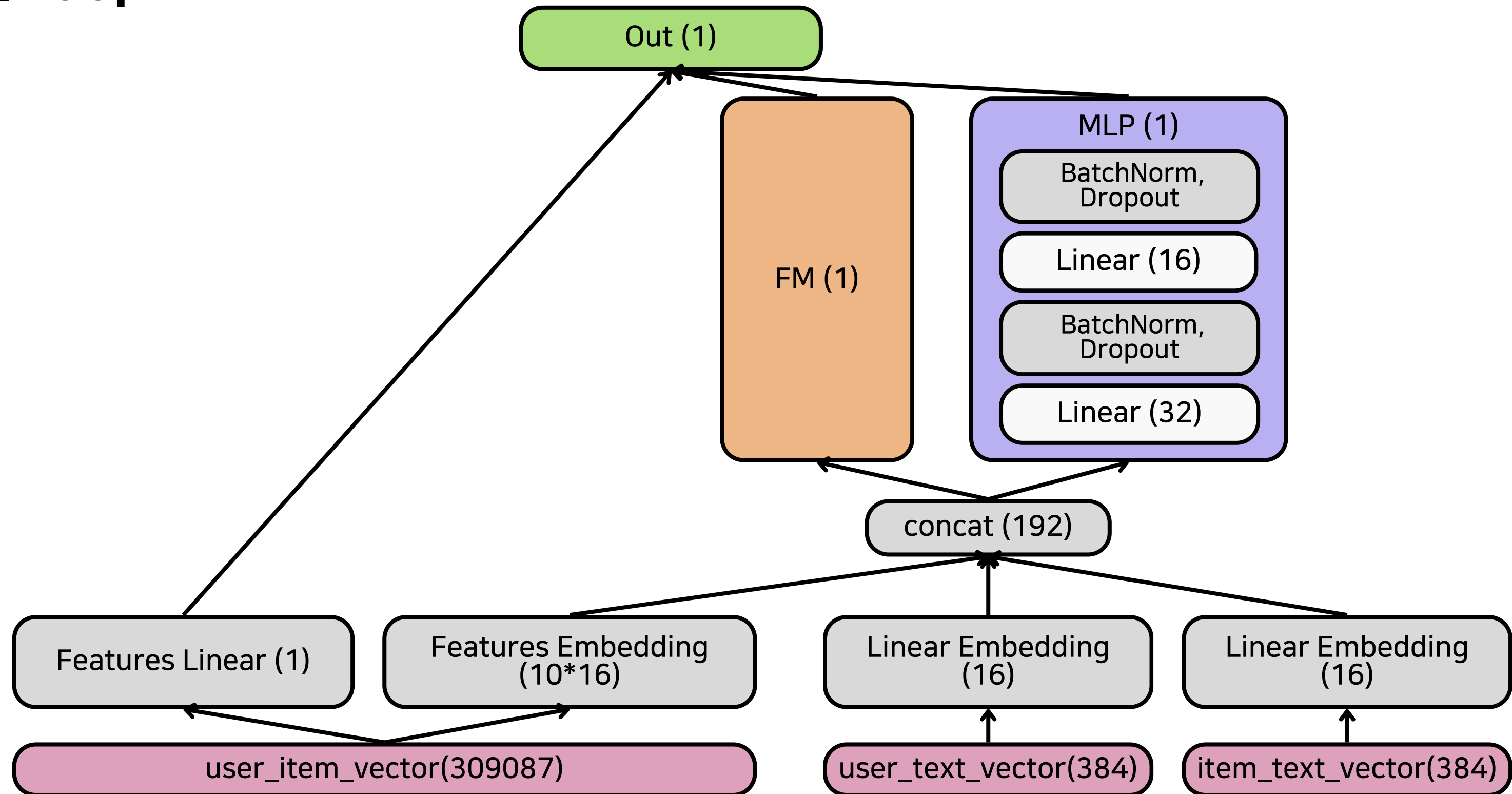
MF Layer에 적용되는 임베딩을 동일하게 MLP에도 적용

SBERT를 통해 생성한 text vector는 384차원으로, MLP의 Layer의 input으로 적용하기에 크기가 크다고 판단  
FM input의 임베딩 레이어와 동일한 파라미터로 임베딩 적용 후, MLP Layer 통과

## 3. 데이터셋 추가

context 데이터를 user\_book\_vector에 concat하여 전체 학습 과정에 추가함

# Text\_DeepFM



# CatBoost

## 1. Categorical feature

Category의 특성을 활용할 수 있다.

'cat\_features' 매개변수를 사용하여 카테고리 형 데이터를 학습할 수 있다.  
카테고리 형 변수는 "순위 기반 특성 임베딩" 기법을 통해 모델에 반영된다

## 2. 모델 구조

### 결정 트리 (Decision Trees)

CatBoost는 여러 개의 결정 트리를 순차적으로 학습하여 예측을 개선한다. 각 트리는 이전 트리의 오차를 보완하는 방향으로 학습된다.

### Gradient Boosting

각 트리는 이전 트리의 예측 오류를 줄이기 위해 학습되며, 이를 반복하여 모델의 성능을 향상시킨다.

## 3. cat\_feature

```
['user_id', 'age_range', 'location_country', 'location_state', 'location_city', 'isbn', 'book_title', 'book_author', 'publisher', 'language', 'category', 'publication_range']
```

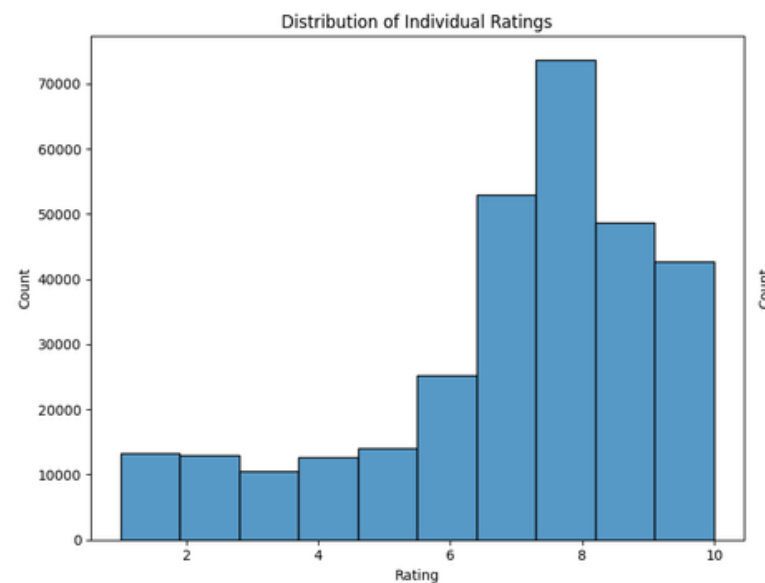
# CatBoost Categorization

## 1. Idea

Classifier로는 결과가 어떨까? -> 10개 label을 분류하기는 힘들 것 같으니 rating을 범주화하자.

-> 범주화 한 김에 이대로 Regressor로 학습해볼까? -> Catboost 결과가 좋으니 CatBoostRegressor를 사용하자.

## 2. 범주화 기준



구간	값
1~3	2
4~6	5
7~7	7
8~8	8
9~9	9
10~10	10

```
def map_rating(rating):  
    for (low, high), new_value in rating_mapping.items():  
        if low <= rating <= high:  
            return new_value  
    return rating  
  
y_train = y_train.apply(map_rating)
```

## 3. result

Valid Score : 2.1382 / Private Score : 2.1321 / Public Score : 2.1252

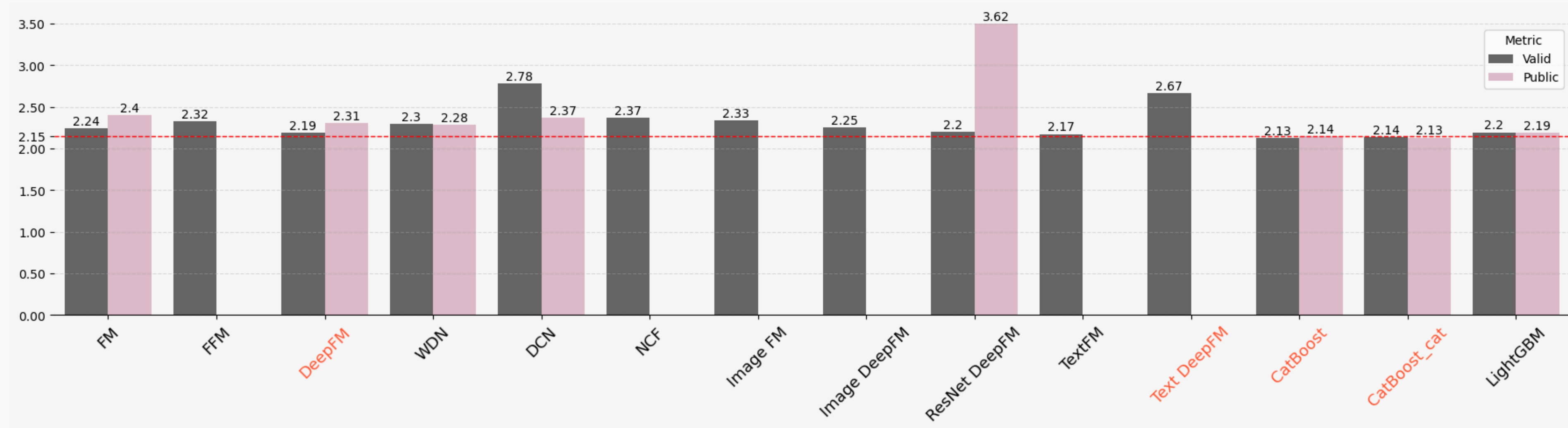
## 03. 최종 결과물

최종 모델 선정  
양상블

# Modeling result

## 최종 모델 선정 기준

1. Boosting Model: Validation Score, Public Score < 2.15  
CatBoost, CatBoost\_범주화
2. DL Model: Validation Score < 2.2  
DeepFM, Text\_DeepFM





# Hyper parameter

## Catboost

bagging\_temperature: 0.049726643578359875  
border\_count: 254  
depth: 6  
devices: '0'  
iterations: 871  
l2\_leaf\_reg: 7.501546772536532  
learning\_rate: 0.14197321614133618  
loss\_function: RMSE  
model\_size\_reg: 0.5  
od\_type: Iter  
od\_wait: 20  
random\_seed: 0  
rsm: 1.0  
task\_type: GPU  
thread\_count: -1  
verbose: true

## DeepFM

embed\_dim: 16  
mlp\_dims: [32, 16]  
batchnorm: True  
dropout: 0.4  
optimizer.type: Adam  
optimizer.args.lr: 0.0005  
lr\_scheduler.use: True  
type: ReduceLROnPlateau  
train.epochs: 20

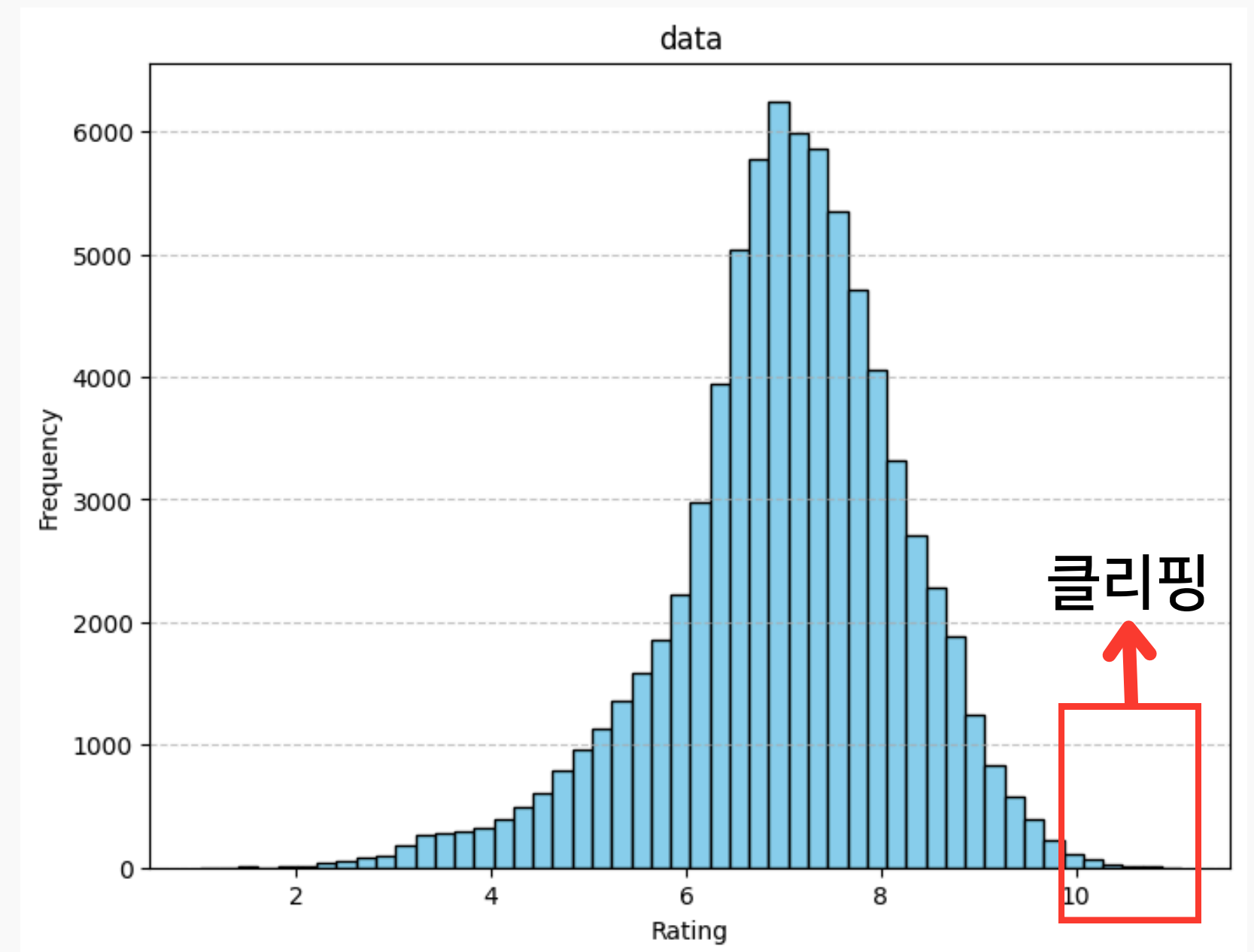
## Text\_DeepFM

embed\_dim: 16  
pretrained\_model:  
'sentence-transformers/all-MiniLM-L6-v2'  
mlp\_dims: [32, 16]  
batchnorm: True  
dropout: 0.1  
optimizer.type: Adam  
optimizer.args.lr: 0.005  
lr\_scheduler.use: False  
train.epochs: 10

# 앙상블

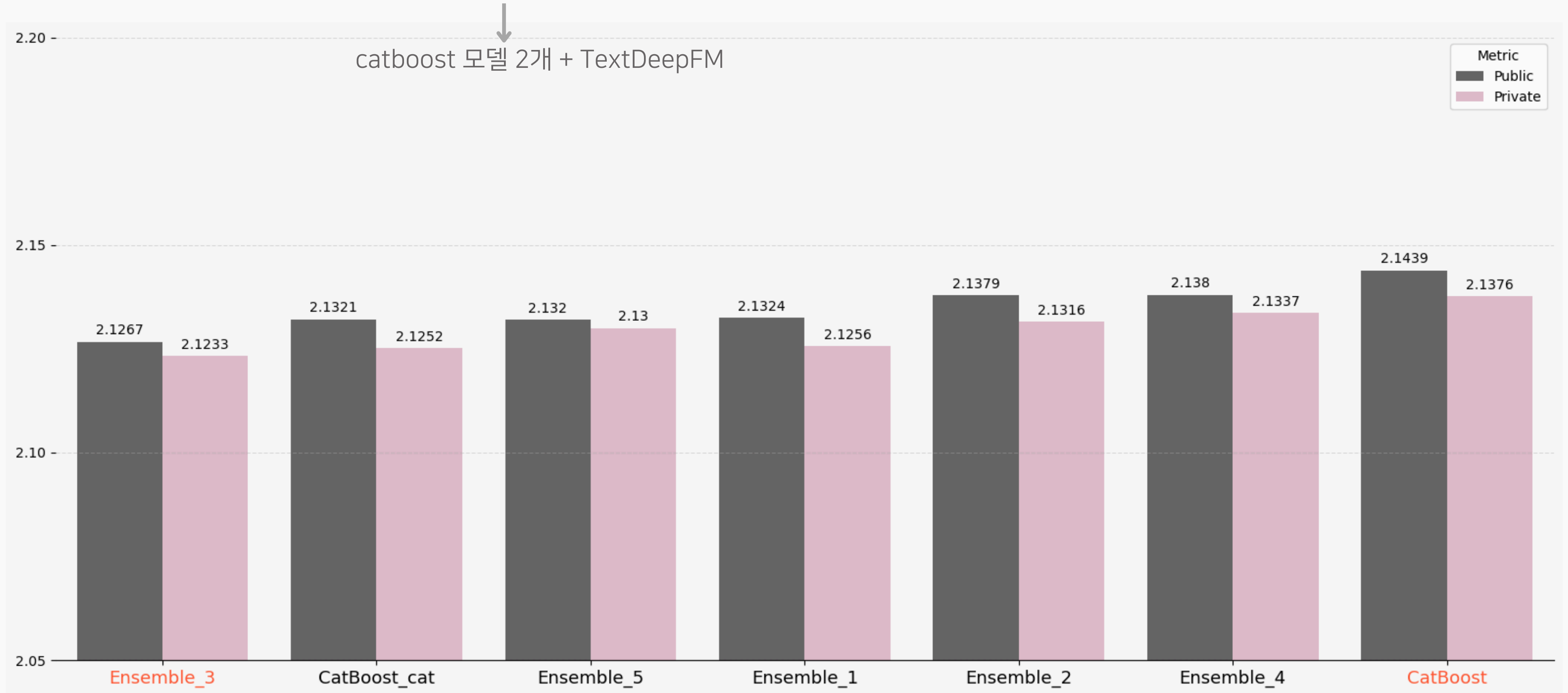
Average Ensemble -> (1, 10) clip

No.	Ensemble Configuration
1	Catboost 2개 e_Catboost + e_Catboost_c
2	Catboost 2개 + DeepFM e_Catboost + e_Catboost_c + e_DeepFM
3	Catboost 2개 + TextDeepFM e_Catboost + e_Catboost_c + e_TextDeepFM
4	Catboost 2개 + FM + TextDeepFM e_Catboost + e_Catboost_c + e_FM + e_TextDeepFM
5	Catboost_c + TextDeepFM e_Catboost_c + e_TextDeepFM



## 최종 결과

앙상블 중 가장 결과가 좋은 Ensemble\_3와 단일 모델 중 가장 결과가 좋은 Catboost 모델 제출



## 04. 회고

새롭게 배운 점  
아쉬웠던 점

# 회고

## 새롭게 배운 점

- 체계적으로 짜여있는 베이스라인 코드를 통해 코드 모듈화를 배울 수 있었다.
- shell script로 실행하는 코드의 편리함을 느꼈다.
- wandb를 활용하여 실험을 관리하고 협업하는 방법을 배웠다.

## 아쉬웠던 점

- 시간도 부족했고 summary에 null 값이 많아서 LDA나 Top2Vec와 같은 토픽 모델링을 적용시켜보지 못했다.
- 여러 베이스라인 모델이 학습되면서 Valid Loss가 수렴하지 않는 문제를 해결하지 못하였다.
- Text 모델의 vector 생성 프롬프트에 Category를 추가하는 방법을 시간이 부족해서 시도해보지 못했다.



# 감사합니다

들어주셔서 감사합니다

