Book Rating Prediction

RecSys_06 추천해조

노관옥 박경원 이석규 이진원 장성준

목차

- EDA & Preprocessing
- Modeling
- Leaderboard Score

최종 스코어

Public / Private

2.1181 / 2.1137

1st

EDA & Preprocessing (Text Data)

Text Data 전처리를 통해 범주 최소화

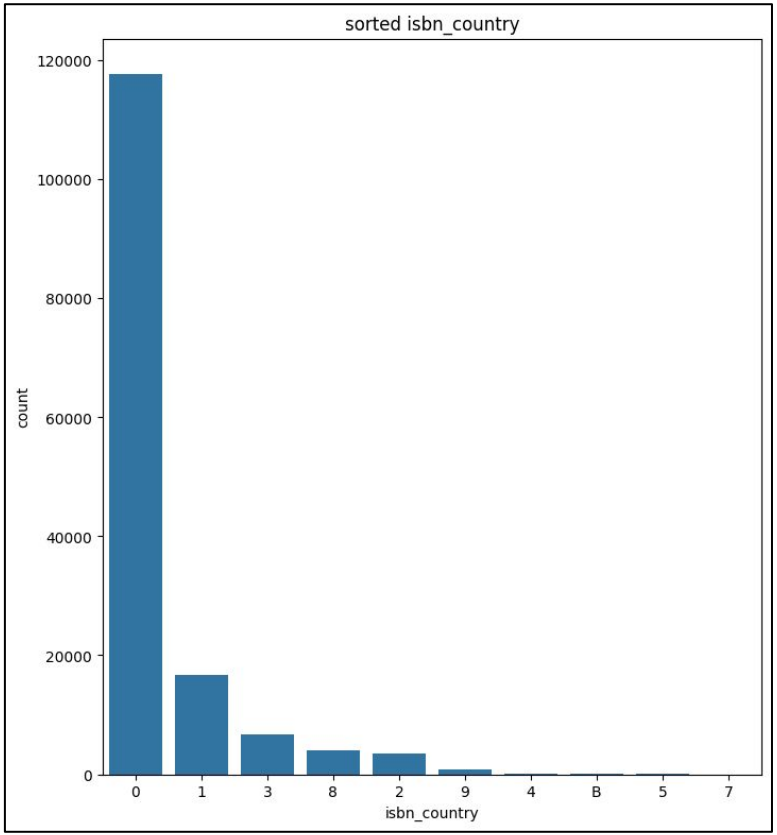
```
def text_preprocessing(summary):
    """
    Parameters
    ----------
    summary : pd.Series
        정규화와 같은 기본적인 전처리를 하기위한 텍스트 데이터를 입력합니다.
    ----------
    """
    summary = re.sub("[.,\'\"''""!?]", "", summary)
    summary = re.sub("[^0-9a-zA-Z\\s]", " ", summary)
    summary = re.sub("\s+", " ", summary)
    summary = summary.lower()
    return summary
  return go(f, seed, [])
}
```

EDA & Preprocessing (isbn)

ISBN-10

Each ISBN-10 has four sections: group identifier, publisher identifier, title identifier, and check digit. A typical 10-digit example is: ISBN 0-545-01022-5. The group identifier is used to identify the country or region. This section may have one to five digits. The example has a global identifier of 0.

The publisher identifier represents the publisher of the book. This section may have up to seven digits. In the example, the publisher identifier is 545.



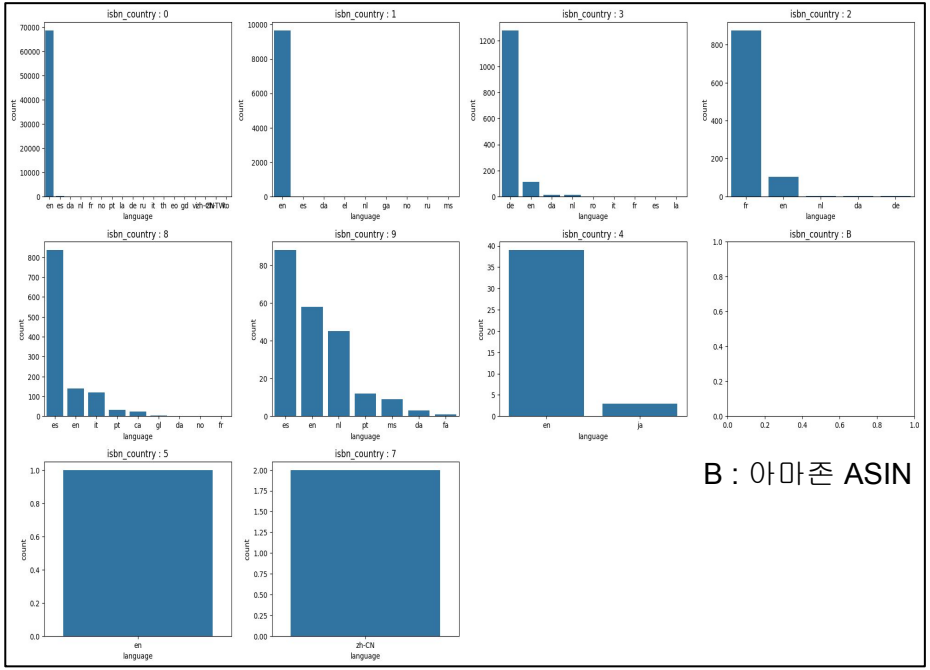
ISBN-10 Format : X-XXX-XXXXX-X

첫 번째 숫자를 국가 코드로 활용

국가 코드 0이 약 78.7%를 차지

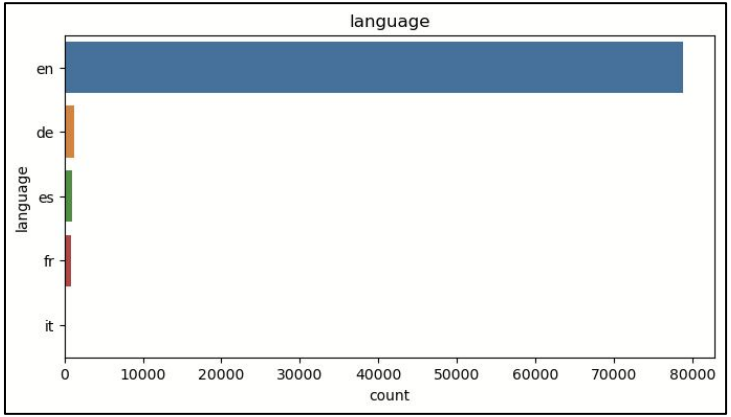
EDA & Preprocessing (language)

국가코드 별 언어 분포



B : 아마존 ASIN

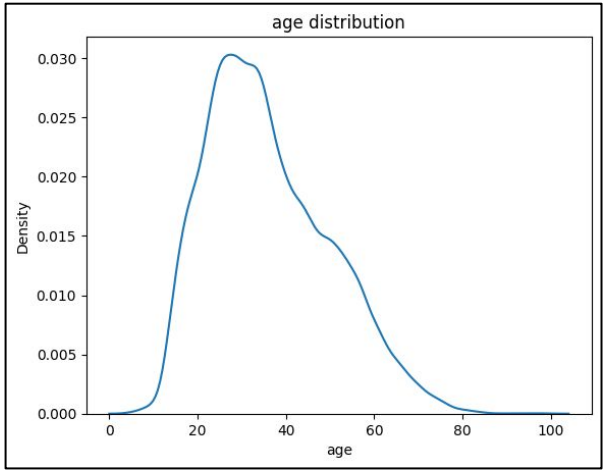
책 언어 **Top 5**



27개의 언어 중 가장 많은 언어는 **영어**

→ isbn의 첫 자릿값(국가 코드)을 사용해서 language 결측치 대체

EDA & Preprocessing (age)




```python
labels = ['3-6','6-8','8-12','12-15','15-18','18-25','25-34','35-44','45-54','55-64','65-74','75+']
bins = [3, 6, 8, 12, 15, 18, 25, 34, 44, 54, 64, 74, 100]

users['age'] = users['age'].apply(lambda x: 100 if x>100 else x)
users.loc[(users['age'] > 90), 'age'] = np.nan

users['age'] = users['age'].fillna(users['age'].mean())
users['age'] = users['age'].astype(np.int64)

users['new_age'] = pd.cut(users.age, bins, labels = labels,include_lowest = True)
```

- 성인 이전은 3년 단위 / 성인 이후는 10년 단위로 범주화
- 결측치는 평균으로 대체

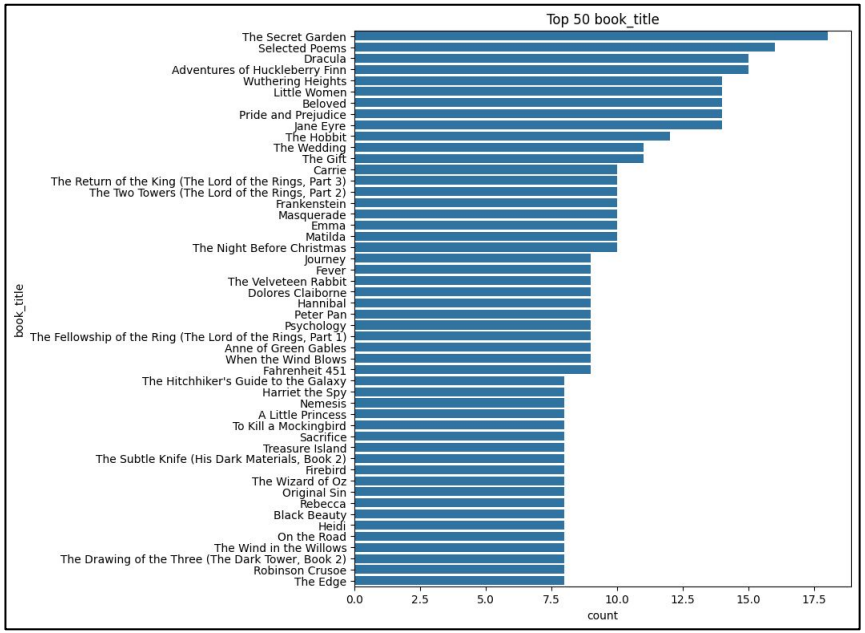
```
----------------------- ANOVA TEST -----------------------
H0 : 연령대 간 Rating의 평균이 동일하다.
H1 : 연령대 간 Rating의 평균이 적어도 하나는 동일하지 않다.

F-statistic : 58.92791
p-value : 2.493623833684452e-108

p-value < alpha이므로 유의수준 5% 하에서 H0를 기각한다.
즉, 연령대 간 Rating의 평균이 적어도 하나는 동일하지 않다.
```

연령대 그룹 간의 Rating의 평균 차이를 확인해보기 위해 **ANOVA 검정** 수행

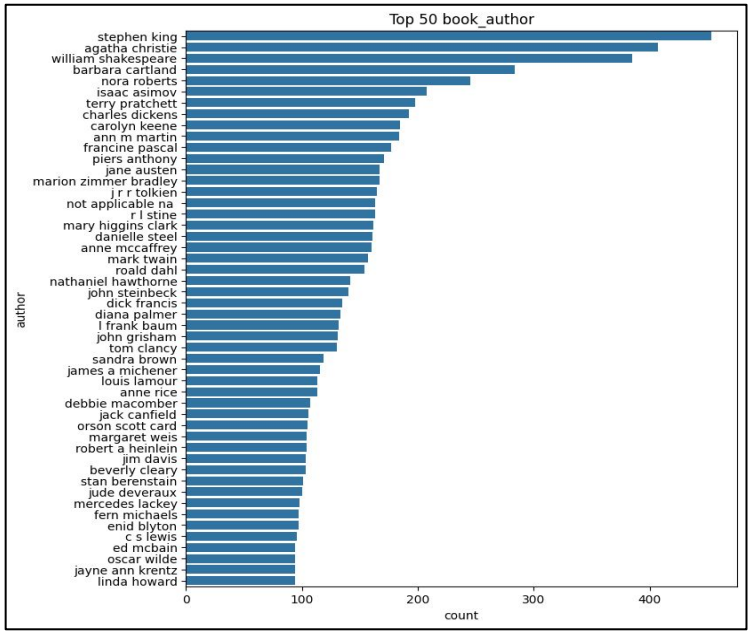
→ 검정 결과, 연령대 그룹 간 Rating의 평균에 차이가 존재

EDA & Preprocessing (book_title, book_author)

책 제목 **Top 50**



책 작가 **Top 50**



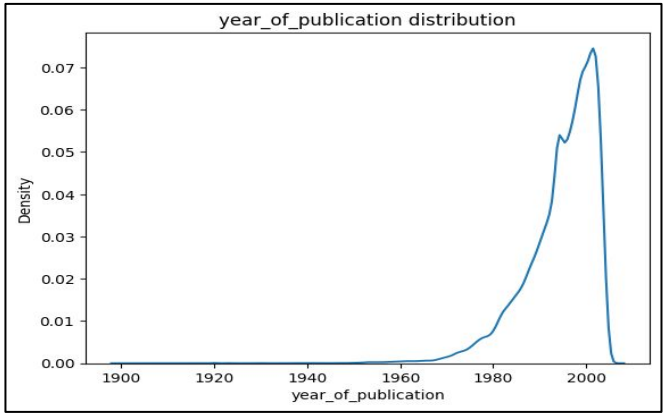
EDA & Preprocessing (book_author)

	isbn	book_title	book_author	year_of_publication	publisher
73737	0751352497	A+ Quiz Masters:01 Earth	NaN	1999.0	Dorling Kindersley

- book_author가 결측치인 책은 실제 저자가 없는 책
- 동일한 book_title & isbn 데이터도 없음

→ 동일한 publisher의 최빈값으로 대체

EDA & Preprocessing (year_of_publication)




```python
def preprocess_year(x) :
    if x <= 1970 :
        return 1970
    elif (x > 1970) and (x <= 1980) :
        return 1980
    elif (x > 1980) and (x <= 1985) :
        return 1985
    elif (x > 1985) and (x <= 1990) :
        return 1990
    elif (x > 1990) and (x <= 1995) :
        return 1995
    elif (x > 1995) and (x <= 2000) :
        return 2000
    else:
        return 2006
```

- 출판 년도가 대부분 1980년 이후
- 데이터가 적은 1970년 이전, 2000년 이후를 제외한 나머지 년도는 **5년 단위로 범주화**

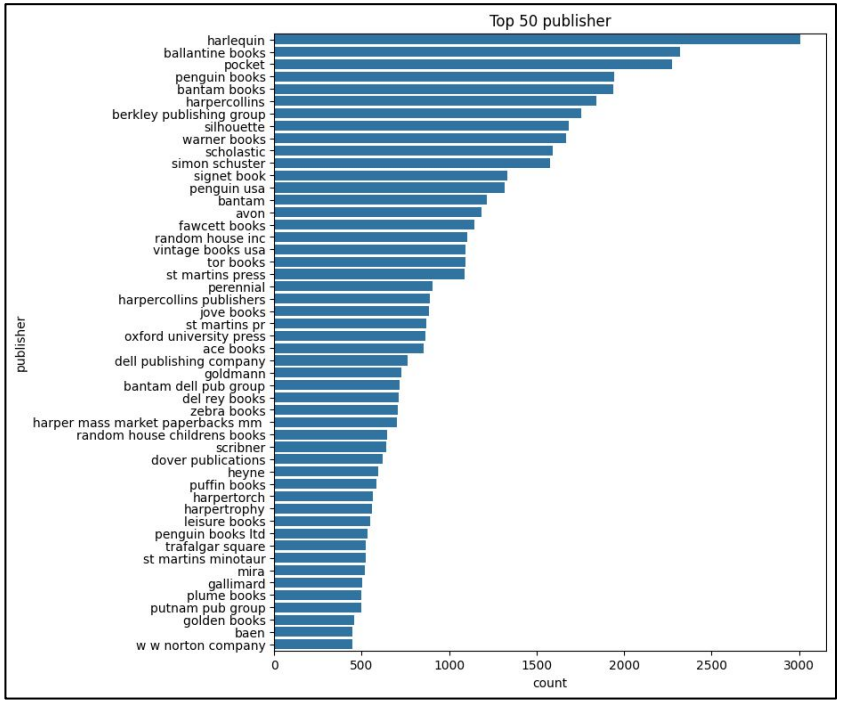
```
------------------------ ANOVA TEST ------------------------
H0 : 출판년도 그룹 간 Rating의 평균이 동일하다.
H1 : 출판년도 그룹 간 Rating의 평균이 적어도 하나는 동일하지 않다.

F-statistic : 66.49107
p-value : 5.347374704534725e-83

p-value < alpha이므로 유의수준 5% 하에서 H0를 기각한다.
즉, 출판년도 그룹 간 Rating의 평균이 적어도 하나는 동일하지 않다.
```

출판년도 그룹 간의 Rating의 평균 차이를 확인해보기 위해 **ANOVA 검정** 수행

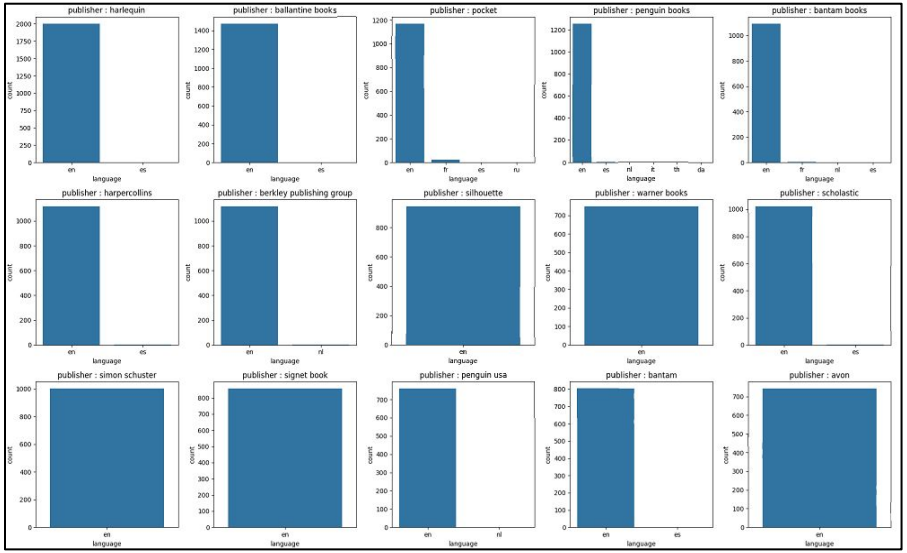
→ 검정 결과, 출판년도 그룹 간 Rating의 평균에 차이가 존재

EDA & Preprocessing (publisher)

출판사 **Top 50**



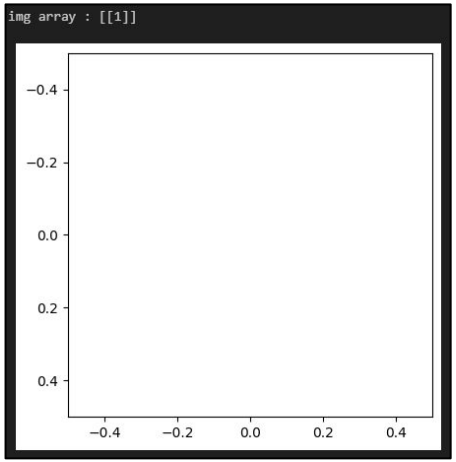
출판사 별 **language** 시각화



- 대부분의 출판사에서 **영어로 출간**
- publisher로 language의 결측치를 채우려고 했으나 isbn을 활용하는 것보다 성능이 안좋았음

EDA & Preprocessing (img_url)

```python
# 이미지는 있지만 비어있음
img = Image.open("/home/code/data/images/0425139441.01.THUMBZZZ.jpg")
img_array = np.array(img)
print("img array :",img_array)
img_array = img_array.astype(float)
plt.imshow(img)
plt.show()
```



img_url 확인 결과 **책 표지가 없는 데이터가 존재**

Binarization 되어있는 Image의 array 길이가 1인 경우를 **결측치로 판단**

→ 전체 데이터의 28%인 41,802개가 결측치인 것을 확인

EDA & Preprocessing (category)

```
from sklearn.neural_network import MLPClassifier
NN_clf = MLPClassifier(activation='logistic', alpha=0.00003, batch_size='auto',
                       beta_1=0.9, beta_2=0.999, early_stopping=True,
                       epsilon=1e-08, hidden_layer_sizes=(20,), learning_rate='constant',
                       learning_rate_init=0.003, max_iter=200, momentum=0.9,
                       nesterovs_momentum=True, power_t=0.5, random_state=1, shuffle=True,
                       solver='adam', tol=0.0001, validation_fraction=0.1, verbose=False,
                       warm_start=False)
NN_clf.fit(X_train, y_train)
pred = NN_clf.predict(X_test)
print (metrics.f1_score(y_test, pred, average='macro'))
print (metrics.accuracy_score(y_test, pred))
```

book_image를 활용하여 category 결측치를 대체 → book_image의 **결측치가 많아서 포기**

book_title로 category 결측치를 MLP를 통하여 예측 → Accuracy가 46%로 **성능이 너무 안좋아서 사용 X**

EDA & Preprocessing (category)

	category	count
1	fiction	33016
2	juvenile fiction	5835
3	biography autobiography	3326
4	history	1927
5	religion	1818
6	juvenile nonfiction	1418

	category	count
1	fiction	39678
2	biography autobiography	3326
3	history	1927
4	religion	1824
5	nonfiction	1427
6	humor	1291

- category의 **상위 카테고리 category_high** 생성
- category가 5개 이하면 others

EDA & Preprocessing (summary)

```
summary 결측치 개수 : 67227
summary 결측치 비율 : 44.95%
```

책 요약 정보 결측치가 **44.95%**로 다수 존재
데이터를 채워넣을 Solution이 없어서 제거

최종 Variable	
isbn	new_age
book_title	years
book_author	city
language	state
publisher	country
category_high	isbn_country
user_id	review_counts

대부분 범주형 변수!

Modeling

범주형 변수에 대해 효과적인 Gradient Boosting 라이브러리

publication_of_year와 age를 범주화하여 **모든 변수를 범주화**

Optuna를 활용하여 HPO(Hyper Parameter Optimization) 수행

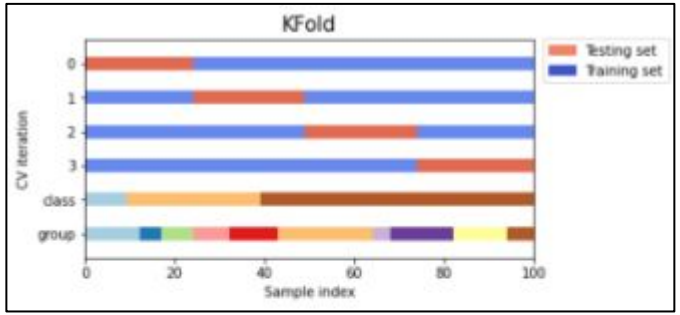
Modeling (Catboost)

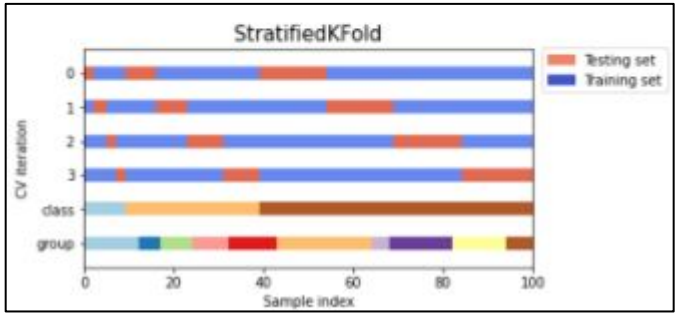
```python
def objectiveCAT(trial : Trial, X_train, y_train, X_valid, y_valid) :
    param = {
        'depth' : trial.suggest_int('depth', 4, 15),
        'learning_rate' : trial.suggest_categorical('learning_rate', [1e-3, 0.01, 0.1, 0.5]),
        'colsample_bylevel' : trial.suggest_categorical('colsample_bylevel', [1e-3, 0.01, 0.1, 0.5, 1.0]),
        'boosting_type' : trial.suggest_categorical('boosting_type', ['Ordered', 'Plain']),
        'bootstrap_type' : trial.suggest_categorical('bootstrap_type', ['Bayesian', 'Bernoulli', 'MVS'])
    }

    if param['bootstrap_type'] == 'Bayesian':
        param['bagging_temperature'] = trial.suggest_float('bagging_temperature', 0, 10)
    elif param['bootstrap_type'] == 'Bernoulli':
        param['subsample'] = trial.suggest_float('subsample', 0.1, 1)

    train_data = Pool(data = X_train, label = y_train, cat_features = cat_col)
    valid_data = Pool(data = X_valid, label = y_valid, cat_features = cat_col)

    model = CatBoostRegressor(**param, iterations = 5000, loss_function = 'RMSE', eval_metric = 'RMSE',
                              use_best_model = True, random_state = SEED, # task_type = 'GPU', devices = '0'
                              cat_features = [i for i in range(0, 12)])
    pruning_callback = CatBoostPruningCallback(trial, 'RMSE', eval_set_index = 1)
    cat_model = model.fit(train_data, eval_set = [train_data, valid_data], verbose = 500, use_best_model = True,
                          early_stopping_rounds = 100, callbacks = [pruning_callback])
    pruning_callback.check_pruned()

    score = mean_squared_error(y_valid, cat_model.predict(X_valid), squared = False)
    return score
```

CatBoostPruningCallback 사용

- HPO 중 가지치기(Pruning)을 수행하여 불필요한 실험을 중단하는 역할
- GPU 지원 X

Modeling (Catboost)

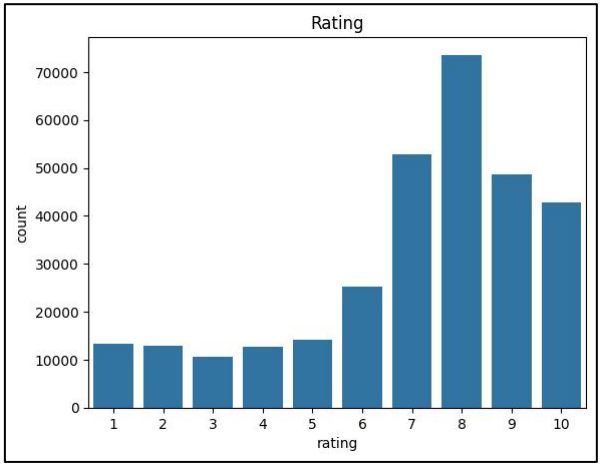




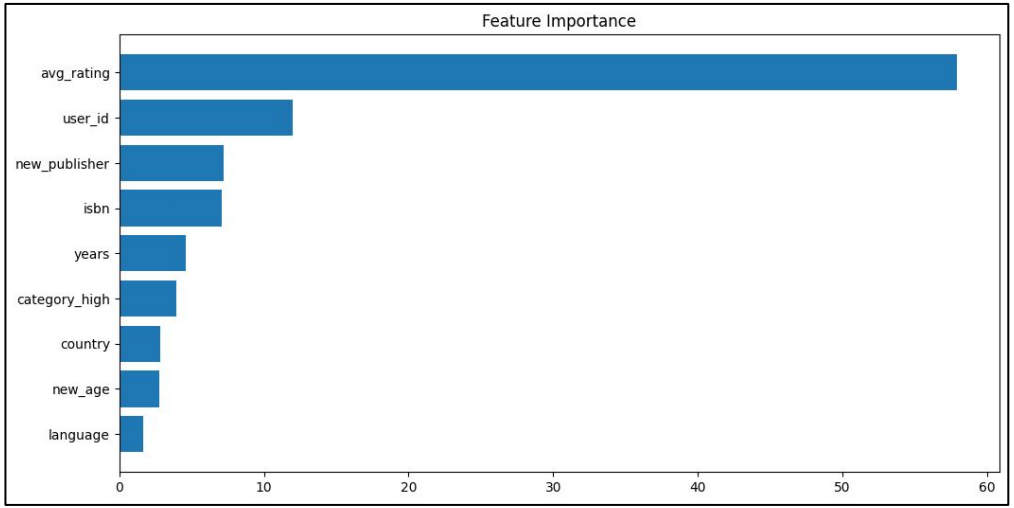
일반적으로 Regression 문제에서 Label의 값이 연속형이므로
Stratified K-Fold 지원 X

→ But, Rating이 이산형(정수)로 되어있기 때문에,
Stratified K-Fold 사용 가능

Rating 값의 분포의 차이가 크므로, Stratified K-Fold 수행



Modeling (Catboost)



Best Parameter	
learning_rate	0.1
depth	11
colsample_bylevel	0.5
boosting_type	Plain
bootstrap_type	MVS

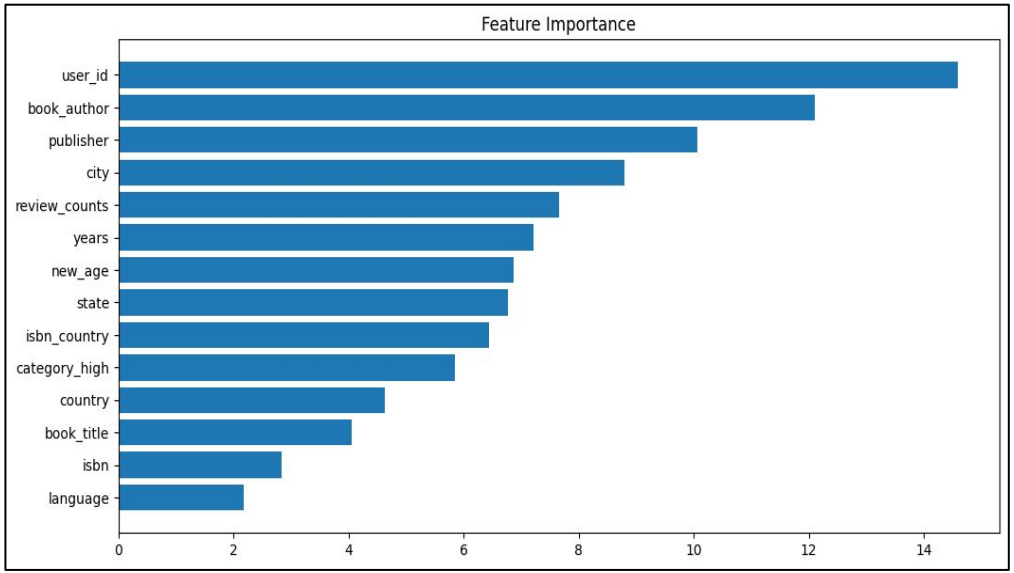
avg_rating 변수의 Feature Importance 값이 **다른 변수에 비해 매우 높음**

- LB : 2.1850 / CV : 1.7657
- **LB - CV의 Gap이 매우 큼**

모델이 avg_rating 변수에 **지나치게 의존**하여

모델의 학습이 더 안된다고 판단 !

Modeling (Catboost)



추가한 변수	제거한 변수
isbn_country review_counts	avg_rating

LB : 2.1226 / CV : 2.1254

→ LB - CV Gap도 작으며,

LB 성능도 오히려 상승

성능을 더 끌어올릴 수 있는 방법은 없을까 ?

→ **이미지 & 텍스트 데이터를 활용**

Modeling (CNN_FM & DeepCoNN)

```python
def objective_CNN_FM(trial : Trial, dataset) :

    setting = Setting()
    setting.seed_everything(SEED)

    log_path = setting.get_log_path(args)
    setting.make_dir(log_path)

    logger = Logger(args, log_path)
    logger.save_args()

    # Common Parameter
    args.batch_size = trial.suggest_categorical('batch_size', [256, 512, 1024])
    args.lr = trial.suggest_categorical('lr', [1e-3, 0.01, 0.1, 0.5])
    args.optimizer = trial.suggest_categorical('optimizer', ['SGD', 'Adam', 'AdamW', 'NAdam'])
    args.weight_decay = trial.suggest_categorical('weight_decay', [1e-7, 5e-7, 1e-6, 5e-6, 1e-5])

    # CNN_FM Parameter
    args.cnn_embed_dim = trial.suggest_int('cnn_embed_dim', 16, 64)
    args.cnn_latent_dim = trial.suggest_int('cnn_latent_dim', 8, 16)

    args.model = 'CNN_FM'
    args.loss_fn = 'RMSE'
    args.epochs = 30

    data = image_data_split(args, dataset)
    data = image_data_loader(args, dataset)

    model = CNN_FM(args, data).to(args.device)
    model, minimum_loss = train(args, model, data, logger, setting)
    return minimum_loss
```

LB Score	2.1739
CV Score	2.1674

CNN_FM

```python
def objective_DeepCoNN(trial : Trial, dataset) :

    setting = Setting()
    setting.seed_everything(SEED)

    log_path = setting.get_log_path(args)
    setting.make_dir(log_path)

    logger = Logger(args, log_path)
    logger.save_args()

    # Common Parameter
    args.batch_size = trial.suggest_categorical('batch_size', [256, 512, 1024])
    args.lr = trial.suggest_categorical('lr', [1e-3, 0.01, 0.1, 0.5])
    args.optimizer = trial.suggest_categorical('optimizer', ['SGD', 'Adam', 'AdamW', 'NAdam'])
    args.weight_decay = trial.suggest_categorical('weight_decay', [1e-7, 5e-7, 1e-6, 5e-6, 1e-5])

    # DeepCoNN Parameter
    args.deepconn_embed_dim = trial.suggest_int('deepconn_embed_dim', 16, 64)
    args.deepconn_latent_dim = trial.suggest_int('deepconn_latent_dim', 4, 16)
    args.conv_1d_out_dim = trial.suggest_int('conv_1d_out_dim', 32, 128)
    args.kernel_size = 3
    args.word_dim = 768
    args.out_dim = trial.suggest_int('out_dim', 16, 64)

    args.model = 'DeepCoNN'
    args.loss_fn = 'RMSE'
    args.epochs = 30

    data = text_data_split(args, dataset)
    data = text_data_loader(args, dataset)

    model = DeepCoNN(args, data).to(args.device)
    model, minimum_loss = train(args, model, data, logger, setting)
    return minimum_loss
```

LB Score	2.2211
CV Score	2.2161

DeepCoNN

Optuna를 활용하여 HPO를 수행

Modeling (Ensemble)

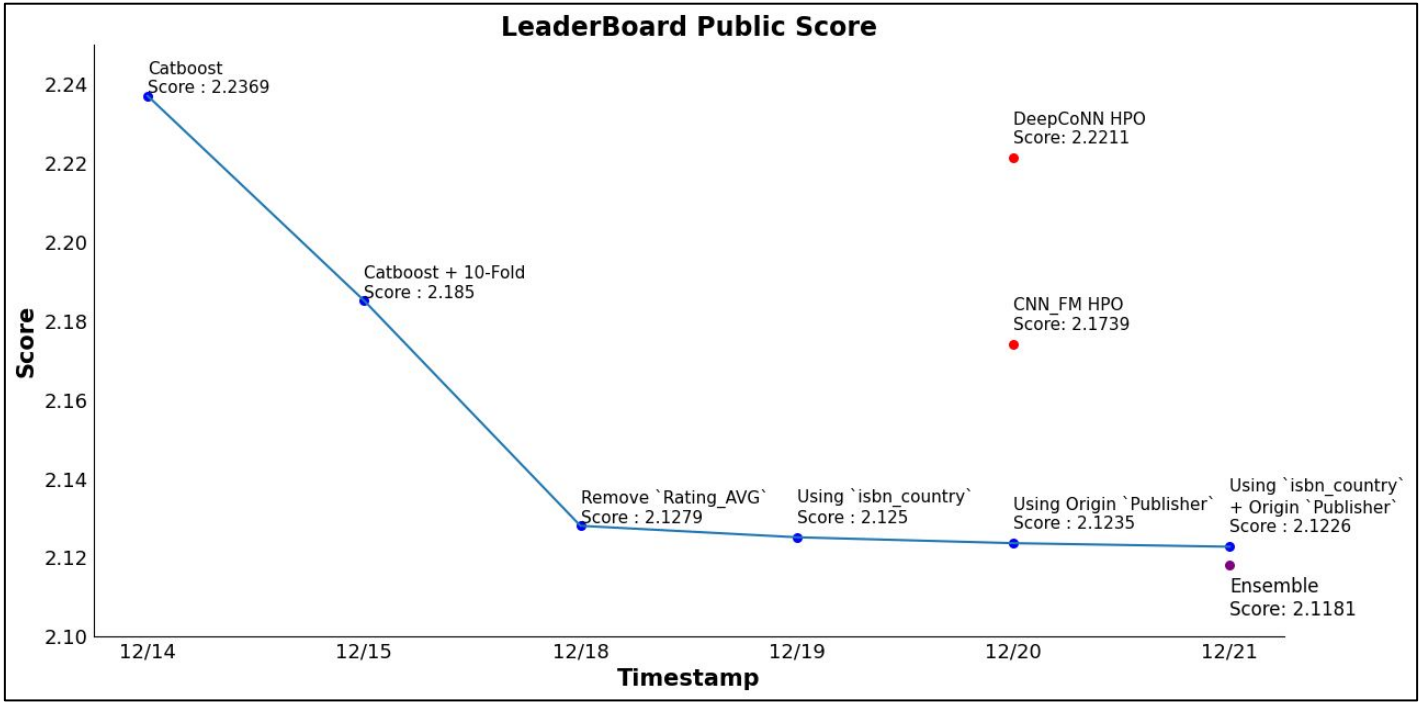
모델	Catboost	CNN_FM	DeepCoNN
LB Score	2.1226	2.1739	2.2211

Ensemble	
LB Score	**2.1187**

모델의 성능에 따라 **Catboost : CNN_FM : DeepCoNN = 0.8 : 0.15 : 0.05**으로 앙상블 진행

Catboost 단일 모델의 성능보다 이미지 & 텍스트 데이터를 활용하는 CNN_FM, DeepCoNN과 앙상블을 한 모델의 성능이 Best !

Leaderboard Score



1	RecSys_06조		2.1137

감사합니다!

RecSys_06 추천해조

노관옥 박경원 이석규 이진원 장성준