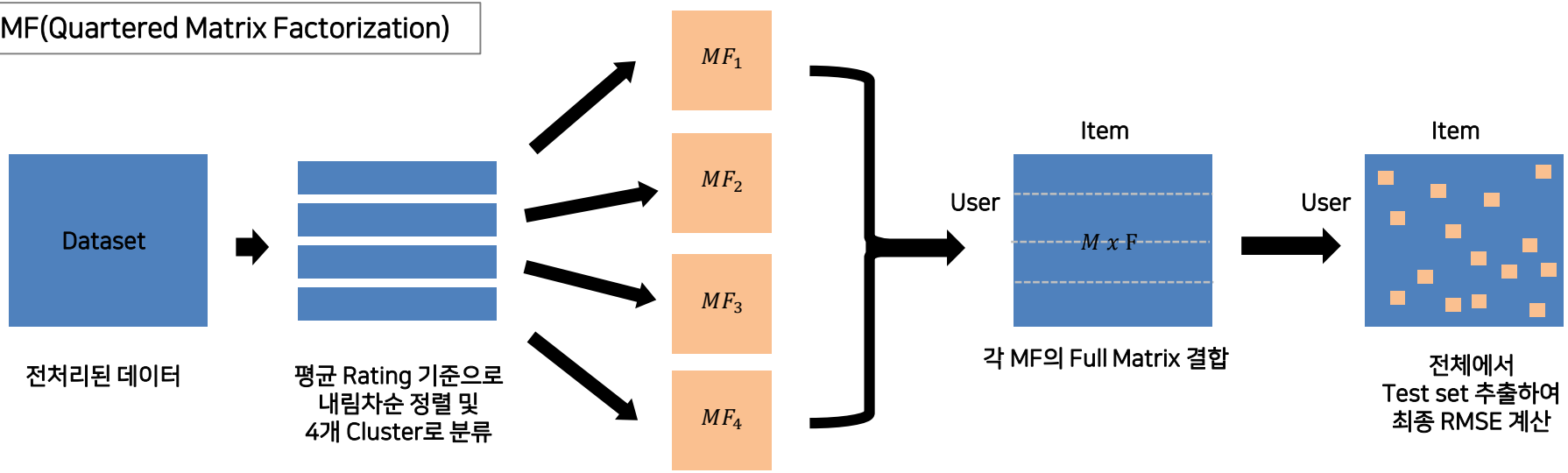
추천시스템 기말 프로젝트

강민수 권홍욱 김미소 박나무

CONTENTS

01. 핵심 아이디어

QMF(Quartered Matrix Factorization)



- 평가 경향이 서로 다른 모든 유저들을 포함하여 단일 모델로 학습을 진행 시, 노이즈가 커지는 단점 발생
- 따라서, 평가 경향이 비슷한 유저들끼리 군집화하여 군집 별 학습의 필요성 확인
- 평균 Rating이 근사하면 비슷한 평가 경향을 보인다고 가정한 뒤, 평균 Rating 기준으로 내림차순 정렬하여 25%씩 4개의 Cluster로 나눔
- 각 Cluster 별로 MF(Matrix Factorization) 모델 학습 진행
- 각 MF의 Full Matrix값들을 단일 User x Item Matrix형태로 결합하여, Test set을 추출하여 최종 RMSE 계산

02. 핵심 코드

QMF(Quartered Matrix Factorization)

```
class QMF():
    # Initializing the object
    def __init__(self, rating_matrix, K, alpha, beta, iterations, verbose=True):

        self.R_ = rating_matrix
        self.K = K
        self.alpha = alpha
        self.beta = beta
        self.iterations = iterations
        self.verbose = verbose

        # Calculate and sorting by user mean
        self.user_mean = self.R_.mean(axis=1).sort_values(ascending=False)
        # Set divide size(# of user in each Matrix)
        self.div_size = int(np.ceil(len(self.user_mean)/4))

        # Get indexes of each Matrix
        self.idx1 = self.user_mean.index[:self.div_size]
        self.idx2 = self.user_mean.index[self.div_size:self.div_size*2]
        self.idx3 = self.user_mean.index[self.div_size*2:self.div_size*3]
        self.idx4 = self.user_mean.index[self.div_size*3:]

        # Make quartered Matrixes
        self.R1 = np.array(self.R_.loc[self.idx1].fillna(0))
        self.R2 = np.array(self.R_.loc[self.idx2].fillna(0))
        self.R3 = np.array(self.R_.loc[self.idx3].fillna(0))
        self.R4 = np.array(self.R_.loc[self.idx4].fillna(0))

        # Make full Matrix for validation
        self.R0 = np.concatenate([self.R1, self.R2, self.R3, self.R4])
```

유저 별 평균 rating 계산 후 내림차순 정렬 그룹화할 임계치 계산

임계치를 기준으로 유저를 4개의 군집으로 군집화

각 군집별 Quartered matrix 생성

검증을 위한 Full matrix 생성

```
def train(self):
    ...

    # Train each matrixes
    for self.R in tqdm([self.R1, self.R2, self.R3, self.R4]):
        self.num_users, self.num_items = np.shape(self.R)

        make sample...

        for i in tqdm(range(self.iterations)):

            training...

        self.total_matrix = np.concatenate((self.total_matrix, self.full_matrix), axis=0)
```

각 군집별로 개별적인 학습 진행 이후 Full matrix로 concatenation

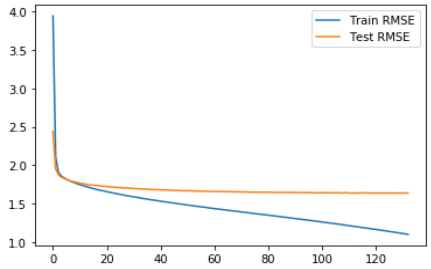
03. 알고리즘 개선 과정

STEP 1

전체 유저를 대상으로 다양한 알고리즘 구현 시도하여 각 알고리즘 별 성능 비교
→기본 MF를 최종 알고리즘으로 선정

알고리즘	RMSE
SVD++	1.6292
기본 MF	1.6133
NCF	1.7078
Keras-MF	1.6639
deep learning	1.63
NCFC	1.6553

[기본 MF 구현 결과]

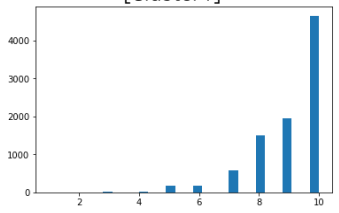


STEP 2

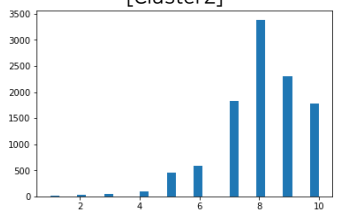
- 유저 별 평균 rating 점수를 구한 뒤 유저 별 평가 경향에 차이가 있다는 점을 발견

→ 전체 유저의 평균 rating 점수를 sorting하여 전체 유저를 25%씩 4개의 Cluster로 나누어 각 Cluster 별 기본 MF 알고리즘 적용

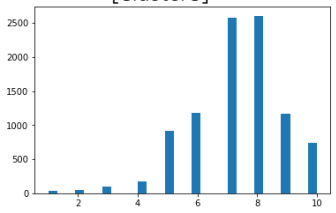
[Cluster1]



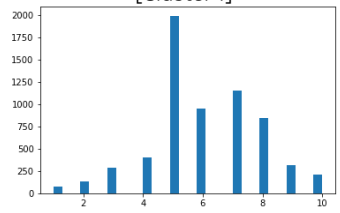
[Cluster2]



[Cluster3]



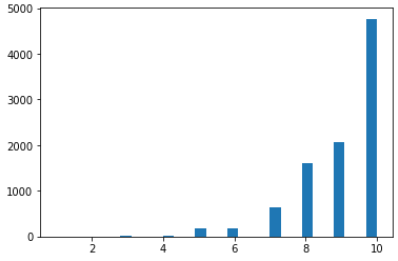
[Cluster4]



- Book의 파생 변수인 major_author를 Book의 bias로 반영함

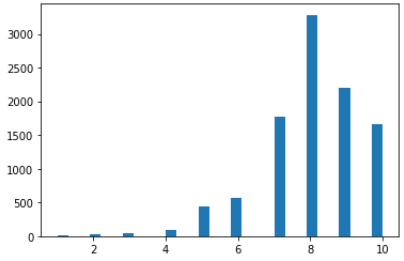
04. 최종 결과

1st Quarter



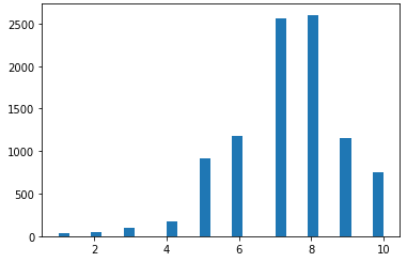
Train RMSE	1.0887
Test RMSE	1.272814

2nd Quarter



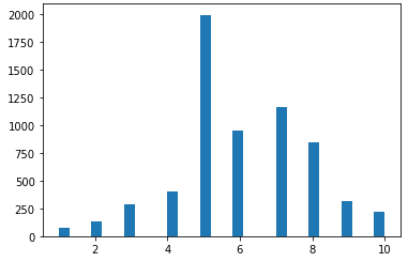
Train RMSE	1.2693
Test RMSE	1.482958

3rd Quarter



Train RMSE	1.3723
Test RMSE	1.638933

4th Quarter



Train RMSE	1.5534
Test RMSE	1.855046

개선 사항 반영하여 최종 모델 학습 시 **RMSE 1.528677**

05. Lessons learned

Lesson 1

평가 경향 비슷한 유저끼리 군집화하여 모델 학습 시, 모델의 예측 정확성 향상

Lesson 2

NCF, NCFC 등등 추천시스템에 다양한 알고리즘 존재

Lesson 3

종속변수에 영향을 끼치는 유의한 파생변수 생성 및 모델 반영의 어려움

Discussion

- 본 프로젝트에서는 단순하게 데이터를 4가지로 등분하여 각각 모델을 학습하는 QMF(Quartered Matrix Factorization) 방식으로 진행. 향후, 최적의 k값을 찾아 알고리즘을 개선시킬 것으로 기대됨
- Book의 파생 변수인 major_author를 Book의 Bias로 반영하여 모델링 설계. 향후, 다양한 방식으로 Rating을 제외한 변수 반영하여 알고리즘 개선시킬 것으로 기대됨

Thank you:)