9. 추천 시스템

작성일시	@2025년 1월 2일 오후 7:15
강의 번호	Euron
유형	스터디 그룹
복습	☐

1. 추천 시스템의 개요와 배경

추천 시스템 개요

- 콘텐츠 포털(ex. 유튜브, 애플 뮤직), 전자 상거래 업체 (ex. 아마존, 이베이) ...
- 사용자가 선택한 콘텐츠와 연관된 콘텐츠 추천
    - 추천한 콘텐츠가 사용자의 취향일 때 → 추천 콘텐츠 선택 → 데이터 축적 ⇒ 더 정확한 추천 결과

온라인 스토어 필수 요소, 추천 시스템

- 방대한 선택으로 인한 압박감 줄여줌

〈 데이터 기반의 추천 시스템 〉

추천 시스템 유형

1. 콘텐츠 기반 필터링
2. 협업 필터링
    a. 최근접 이웃 협업 필터링

b. 잠재 요인 협업 필터링
    - 행렬 분해 기법
    - 대부분의 온라인 스토어 적용 시스템
- 최근 경향: 하이브리드 (콘텐츠 기반 + 협업 기반 필터링)

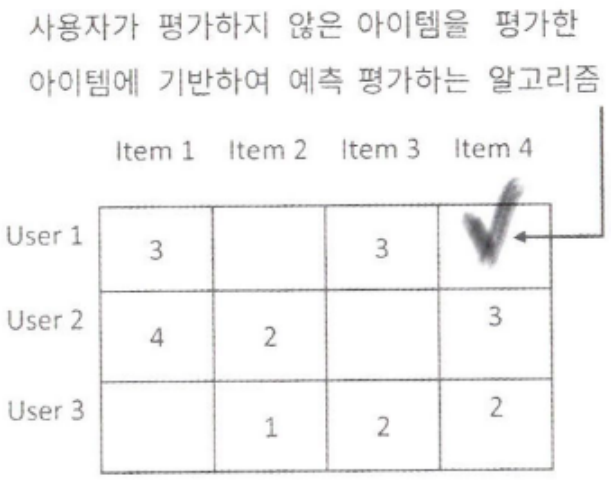
2. 콘텐츠 기반 필터링 추천 시스템

- 사용자 선호 아이템과 비슷한 콘텐츠의 아이템 추천
- ex. 영화 리뷰 기반 추천 시스템

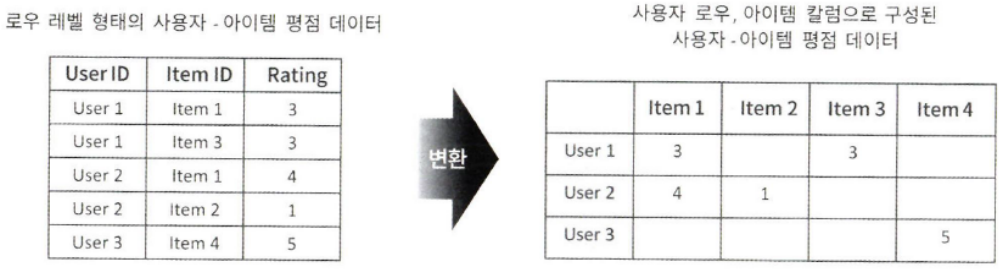
3. 최근접 이웃 협업 필터링

협업 필터링

- 사용자 행동 양식을 기반으로 추천하는 방식
    - 사용자 행동 양식: 아이템에 매긴 평점 정보, 상품 구매 이력 ...
- 목표
    - 사용자 행동 데이터 → 평가하지 않은 아이템을 예측 평가하는 것



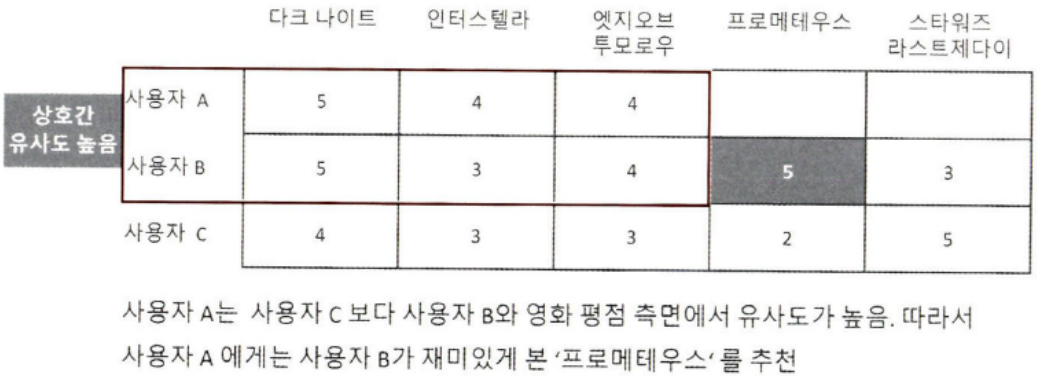
- 사용자-아이템 평점 행렬 데이터만을 기반으로 추천함
  - 사용자-아이템 평점 행렬
    - 행: 개별 사용자, 열: 개별 아이템, 값: 평점
    - 레코드 레벨 형태 데이터일 때 → pivot_table( )로 형태 변환



    - 희소 행렬

최근접 이웃 협업 필터링 (메모리 협업 필터링)

1. 사용자 기반 (User-User)
   - 사용자와 비슷한 고객들이 구매한 상품 추천
     1. 사용자와 타 사용자 간 유사도 측정
        - 유사도 측정: 주로 코사인 유사도 이용
     2. 유사도 가장 높은 TOP-N 사용자 추출
     3. 해당 사용자가 선호하는 아이템 추천



사용자 A는 사용자 C 보다 사용자 B와 영화 평점 측면에서 유사도가 높음. 따라서 사용자 A 에게는 사용자 B가 재미있게 본 '프로메테우스'를 추천

2. 아이템 기반 (Item-Item)
    - 해당 상품을 선택한 고객이 구매한 다른 상품 추천
        - 아이템 자체의 속성과 상관 없음
        - 사용자가 아이템을 선호하는 평가 척도가 기준
            - 평점 분포가 비슷할 때 → 아이템 간 유사도 높음

여러 사용자들의 평점을 기준으로 볼 때 '다크 나이트'와 가장 유사한 영화는 '프로메테우스'

- 행: 개별 아이템, 열: 개별 사용자
    - 사용자 기반 최근접 이웃 방식(User-User)과 반대
- 사용자 기반 보다 정확도 높은 편
    - 비슷한 상품를 구매 → 비슷한 취향을 갖고 있다고 보장할 수 없음
        - 유명한 영화는 취향과 관계없이 대부분의 사람이 관람하는 편
    - 사용자가 평점을 매긴 상품(영화)이 많지 않음

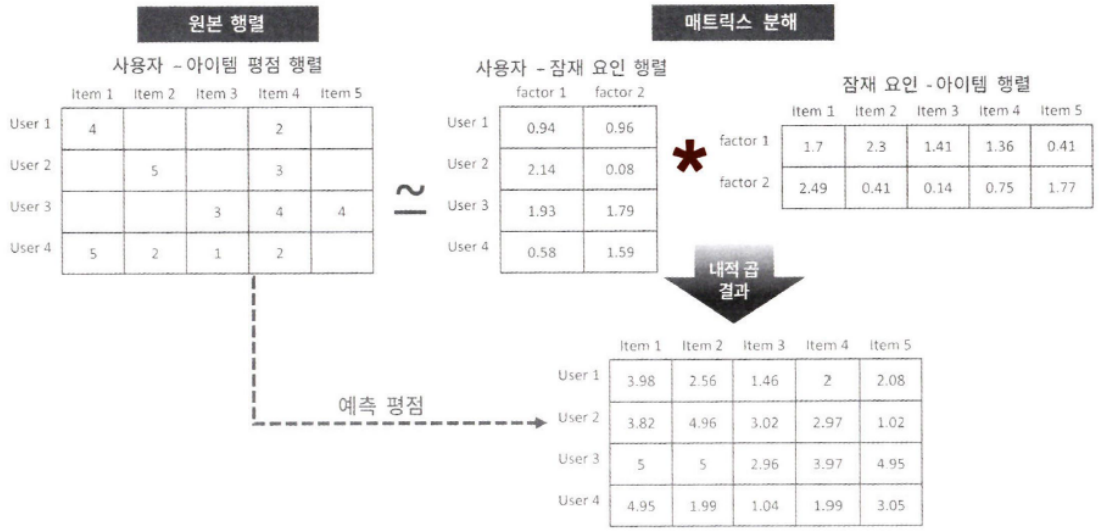
4. 잠재 요인 협업 필터링

잠재 요인 협업 필터링의 이해

- 사용자-아이템 평점 매트릭스 속 잠재 요인 추출 → 추천 예측
    - 사용자-아이템 행렬 데이터 (다차원 희소 행렬) → 사용자-**잠재 요인** 행렬, **잠재 요인**-아이템 행렬 (저차원 밀집 행렬)로 분해

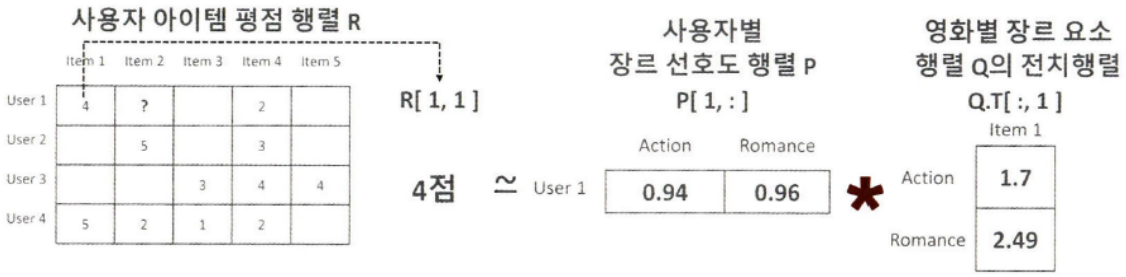
- 분해된 두 행렬 내적

⇒ 새로운 예측 사용자-아이템 평점 행렬 데이터 생성

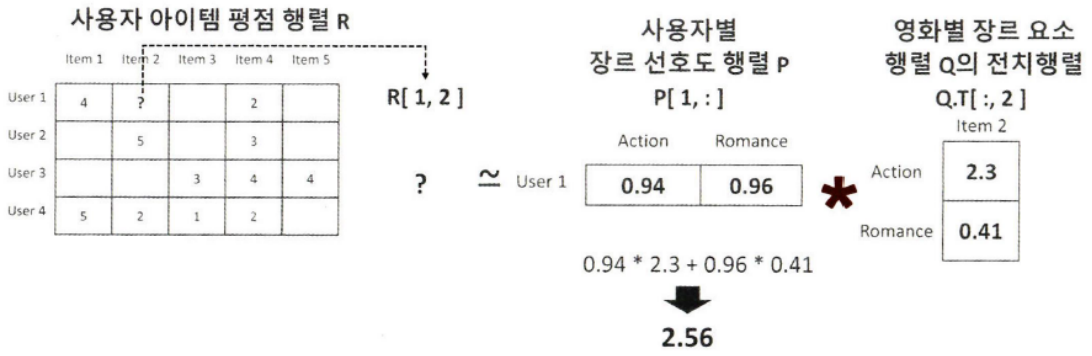


〈 행렬 분해를 통한 잠재 요인 협업 필터링 〉

- ex. 영화 평점 데이터
  - R: 사용자-아이템 평점 행렬, P: 사용자-잠재 요인(장르 선호도) 행렬, Q: 아이템- 잠재 요인 행렬(영화별 장르 특성값)

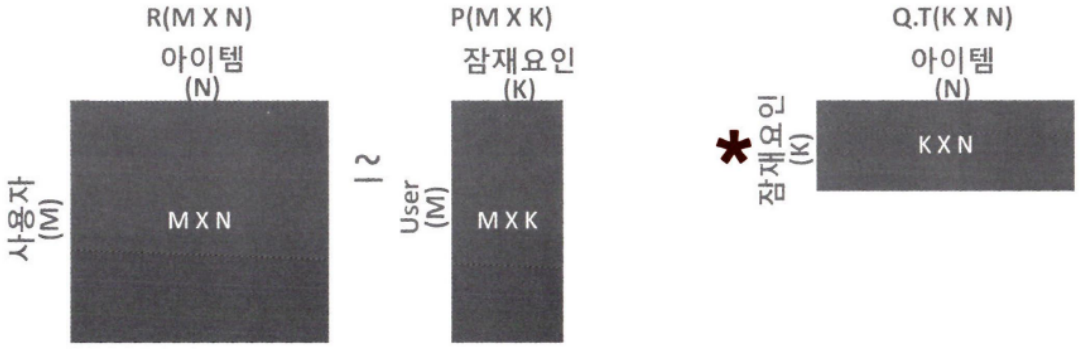


  - 행렬의 내적 결괏값 ⇒ 평점 예측



행렬 분해의 이해

- 다차원 행렬을 차원 감소 기법으로 분해
  - SVD (Singular Vector Decomposition), NMF (Non-Negative Matrix Factorization) ...
    - SVD
      - 주로 사용됨
      - NaN 값 없는 행렬에만 적용 가능함



  - R= P * Q.T
    - M: 총 사용자 수
    - N: 총 아이템 수
    - K: 잠재 요인 차원 수
    - R: 사용자-아이템 평점 행렬 (M*N)
    - P: 사용자-잠재 요인 행렬 (M*K)
    - Q: 아이템-잠재 요인 행렬 (N*K)
  - 내적 ⇒ 사용자-아이템 평점 행렬 (미정 값 포함) 모든 평점 값을 예측 평점으로 다시 계산

$$R \backeq \hat{R} = P * Q.T$$

P

	factor 1	factor 2
User 1	0.94	0.96
User 2	2.14	0.08
User 3	1.93	1.79
User 4	0.58	1.59

*

Q.T

	Item 1	Item 2	Item 3	Item 4	Item 5
factor 1	1.7	2.3	1.41	1.36	0.41
factor 2	2.49	0.41	0.14	0.75	1.77

예측 평점

예측 유저-아이템 Rating 매트릭스 $\hat{R}$

	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	3.98	2.56	1.46	2	2.08
User 2	3.82	4.96	3.02	2.97	1.02
User 3	5	5	2.96	3.97	4.95
User 4	4.95	1.99	1.04	1.99	3.05

확률적 경사 하강법을 이용한 행렬 분해

- 예측 R 행렬 값이 실제 값과 최소의 오류를 갖도록 반복적인 비용 함수 최적화 ⇒ P, Q 유추

1. 임의의 값을 갖는 P, Q 설정
2. 예측 R 행렬 계산 (P * Q.T) → 실제 R 행렬과의 오류 값 계산
3. 오류 값 최소화하는 P, Q 행렬로 업데이트
    - 비용 함수

$$min \sum (r_{(u,i)} - p_u q_i^t)^2 + \lambda (\| q_i \|^2 + \| p_u \|^2)$$

    - 업데이트

$$\acute{p_u} = p_u + \eta (e_{(u.i)} * q_i - \lambda * p_u)$$
$$\acute{q_i} = q_i + \eta (e_{(u.i)} * p_u - \lambda * q_i)$$

        - p_u: P 행렬 u 행
        - q_i: Q 행렬 i 행
        - r(u,i): R 행렬 (u,i) 값
        - e(u,i): r(u,i) - r^(u,i), 오류값
        - lambda: L2 규제 계수
4. 만족할 만한 오류 값 가질 때까지 업데이트

8. 파이썬 추천 시스템 패키지 - Surprise

Surprise 패키지 소개

- 패키지 설치

```
$ pip install scikit-surprise
```

- Surprise 주요 장점
    1. 다양한 추천 알고리즘 적용 가능함
    2. 사이킷런 API와 유사한 API명으로 작성됨
        - fit( ), predict( )

- train_test_split( )
- cross_validate( )

Surprise 주요 모듈 소개

- 데이터셋
  - 사용자 아이디 (user_id), 아이템 아이디 (item_id), rating (평점) 가 로우 레벨로 된 데이터 세트만 적용 가능
    - 로우 레벨: 기계 언어에 가까운 언어
  - 첫번째 칼럼: user_id, 두번째 칼럼: item_id, 세번째 칼럼: rating 가정하고 데이터 로딩
    - 칼럼 순서 지켜져야 함
    - 네 번째 칼럼부터는 로딩하지 않음

Dataset.load_builtin (name='ml-100k')	무비렌즈 데이터 내려받음 (입력 파라미터: name = ml-100k)
Dataset.load_from_file (file_path, reader)	OS 파일에서 데이터 로딩할 때 사용 - 콤마, 탭 등으로 칼럼 분리된 포맷의 파일에서 로딩
Dataset.load_from_df (df, reader)	pandas DataFrame에서 데이터 로딩 - dataframe은 반드시 3개의 칼럼이 uid, iid, r_ui 순서로 정해져 있어야 함

- OS 파일 데이터를 Surprise 데이터 세트로 로딩
  - Reader 클래스 주요 생성 파라미터

line_format (string)	칼럼 순서대로 나열 입력된 문자열은 공백으로 분리 (각 칼럼명)
sep (char)	칼럼 분리자 - 디폴트: '\t' - pandas DataFrame에서 입력 받을 때 기재할 필요 없음
rating_scale (tuple, optional)	평점 최소~최대값 설정

Surprise 추천 알고리즘 클래스

클래스명	설명
SVD	행렬 분해를 통한 잠재 요인 협업 필터링을 위한 SVD 알고리즘
KNNBasic	최근접 이웃 협업 필터링을 위한 KNN 알고리즘
BaselineOnly	SGD 베이스라인 알고리즘 - 사용자 bias, 아이템 bias 감안

- Surprise SVD 비용 함수

- 사용자 베이스라인 편향성 감안 + 규제 적용
  - 사용자 예측 Rating: $r\hat{}ui=\mu+bu+bi+qTipu$
  - Regularization을 적용한 비용 함수: $\Sigma(rui-r\hat{}ui)2+\lambda(b2i+b2u+||qi||2+||pu||2)$
- SVD 클래스 입력 파라미터

파라미터명	내용
n_factors	잠재 요인 K 개수 - 커질 수록 정확도 높아질 수 있음 - but 과적합 주의
n_epochs	SGD 수행 반복 횟수
biased (bool)	베이스 라인 사용자 편향 적용 여부 - True로 유지하는 것이 좋음

- 알고리즘 유형별 성능 평가 비교

알고리즘 유형	RMSE	MAE	Time
SVD	0.934	0.737	0:00:11
SVD++	0.92	0.722	0:09:03
NMF	0.963	0.758	0:00:15
Slope One	0.946	0.743	0:00:08
k-NN	0.98	0.774	0:00:10
Centered k-NN	0.951	0.749	0:00:10
k-NN Baseline	0.931	0.733	0:00:12
Co-Clustering	0.963	0.753	0:00:03
Baseline	0.944	0.748	0:00:01

베이스라인 평점

- 개인의 성향을 반영 → 아이템 평가 시 평향성 요소를 반영하여 평점 부과
- 평점 = 전체 평균 평점 + 사용자 편향 점수 + 아이템 편향 점수
  - 전체 평균 평점: 모든 사용자의 아이템에 대한 평점 평균값
  - 사용자 편향 점수: 사용자별 아이템 평점 평균 값 - 전체 평균 평점
  - 아이템 편향 점수: 아이템별 평점 평균 값 - 전체 평균 평점

모든 사용자의 평균 영화 평점 : 3.5

난 진정한 영화 매니아.
영화 평가는 언제나 깐깐하게

사용자 A 평균 평점
3.0

사용자 A의 어벤저스 3편 베이스 라인 평점 = 3.5 – 0.5 + 0.7 = 3.7

모든 사용자의 평균 영화 평점		사용자 편향 점수		아이템 편향 점수
3.5	+	3.0 – 3.5 = -0.5	+	4.2 – 3.5 = 0.7

어벤저스 3편 평균 평점
4.2