추천 시스템의 목적: 사용자도 좋아하는지 몰랐던 취향을 시스템이 발견. → 그에 맞는 콘텐츠를 제공

추천 시스템의 유형 ① 콘텐츠 기반 필터링
② 협업 필터링 ⟨ ②-① 최근접 이웃
②-② 잠재 요인

콘텐츠 기반 필터링 추천 시스템

: 특정한 아이템을 선호할 경우, 그 아이템과 비슷한 콘텐츠를 가진...

ex) 영화 추천.

최근접 이웃 협업 필터링

: 사용자 행동 양식만을 기반으로 추천을 수행하는 것. (협업 필터링 방식)

→ 사용자-아이템 평점 매트릭스 기반. 측적된 사용자 행동 데이터 기반으로 아직 평가하지 않은 아이템을 예측 평가.

	ITEM1	ITEM2	ITEM3
USER1			
USER2			
USER3			

희소행렬↑

최근접 이웃 협업 필터링 ⟨ 사용자 기반
아이템 기반 ← 많이 이용., 코사인 유사도
(메모리 협업 필터링)

잠재요인 협업 필터링

: 사용자-아이템 평점 행렬 데이터만을 이용해 '잠재 요인'을 끄집어 낸다.

평점: 사용자의 장르 선호도 벡터와 영화의 장르별 특징 벡터의 곱.

행렬분해

평점행렬 R (M x N) → 사용자 K차원 잠재 요인 + 아이템 행렬 Q.T.
사용자 아이템 (M x k) (K x N)

$$R = P * Q.T$$

→ R을 P와 Q로 어떻게 분해?

행렬분해는 주로 SVD 방식을 이용 (but SVD는 null 값이 없는 행렬에만 가능.

R행렬은 희소행렬 ⟹ SGD(경사하강법) / ALS 이용.

<실습>

① 콘텐츠 기반 필터링 실습 (TMDB 5000 영화 데이터)

-장르 속성 기반 (영화/상품/서비스) 간의 유사성 파악.

1) 장르속성 기반

· 장르 칼럼 값의 유사도를 비교 → 높은 평점을 가지는 영화를 추천

1. 필터링에 사용할 칼럼만 추출

→ genre, keywords 칼럼에는 리스트 안에 여러 개의 딕셔너리로 구성. (가공필요)

· ast 모듈의 literal_eval() 함수를 이용. → list [dict 1, dict 2] 로 생성

· apply lambda 식을 이용하여 필요한 값만 추출.
'name'

2. 장르 콘텐츠 유사도 측정

가장 간단한 방식) genres → 문자열로 변경 → Count Vectorizer로 피처 벡터화 → 코사인 유사도

3. 장르 유사도에 따라 영화 추천 함수 생성

→ 추천 기준이 되는 영화 제목 (ex) Godfather), 추천할 건수를 입력

→ 입력된 값의 index 객체를 ndarray 로 반환

→ genre_sorted_ind 유사도 순으로 가까운 추출.

⇒ 출력 결과, 제한이 필요해 보임.

4. 더 많은 후보군 + 영화의 평점으로 필터링

* but. 영화 평점은 왜곡된 데이터일 확률↑

⇒ 평가 횟수에 대한 가중치가 부여된 평점 방식을 이용.

가중평점 = (V/(V+m)) * R + (m/(V+m)) * C

→ 이를 weighted_vote_average 로 새로 생성.

→ 장르 유사성이 높은 영화를 2배수 추출 → weighted_vote 높은 순으로 1배수 최종 추출.

② 아이템 기반 최근접 이웃 협업 필터링 실습 (아이템 기반)

1. ratings 데이터를 로우에는 USER, 나머지를 column으로 놓다. (pivot table 이용)

2. 가독성을 위해 movieID 가 아닌 영화명 (movies 데이터) title로 변경.

→ ratings, movies를 merge → movieID 대신 title → pivot → NaN은 0으로 변경

3. 영화 간 유사도 산출. (코사인 유사도)

ratings matrix가 USER_ID 기준 행 레벨 데이터 → transpose 를 해야 '영화' 기준 유사도.

③ 아이템 기반 최근접 이웃 협업 필터링 (개인화)

개인화된 영화 추천: 어떤 개인이 관심있지 않은 영화를 추천.

1. 사용자별 예측 평점(R_{ui}) = {(사용자 u의 모든 영화에 대한 실제 평점) • (영화 i와 다른 모든 영화와의 코사인 유사도)} / $\sum_{i,N} S_{i,N}$
   (정규화)
2. 예측 결과가 원래의 실제 평점과 얼마나 차이가 있는지를 확인. (MSE)
   → 사용자가 평점을 부여한 영화에 대해서만...
   → MSE를 감소시키는 방향...
3. 특정 영화와 가장 비슷한 유사도를 가지는 영화에 대해서만 유사도 벡터를 적용.
   → 수행시간↑, MSE 향상됨.

④ 행렬 분해를 이용한 잠재 요인 협업 필터링

null 값이 많이 존재 → SGD 기반...

수업의 get_rmse()를 그대로 이용.

1. 영화 평점 정보를 USER-ITEM 행렬로 생성.
2. matrix_factorization() 을 이용해 행렬분해
3. get_unseen_movies() 다시 이용하여 추천.

⑤ 파이썬 추천 시스템 패키지 - Surprise

Surprise는 기본적으로 로우 레벨의 데이터를 칼럼 레벨의 데이터로 변경. (판다스 이용)

- 사용자 아이디 / 아이템 아이디 / 평점 순으로 된 데이터만 가능. (이 3개의 column 만을 이용)

- OS 파일을 로드: 헤더를 삭제해야한다.
- 무비렌즈 데이터 형식이 아닌 OS의 경우, Reader 클래스를 따로 설정.
- 판다스 DataFrame에서 로드 : 칼럼 순서 명시

⑥ 베이스라인 평점 (전체 평균 평점 + 사용자 편향 점수 + 아이템 편향 점수)

: 개인의 성향을 반영해 아이템 평가에 편향성 (bias) 요소를 반영

후한 사람, 짠 사람.

⑦ 교차검증과 하이퍼 파라미터 튜닝

: cross_validate() 이용.