**Recommendation System document**

**By. jhyungit**

1. 추천 시스템의 기술

* 협업 필터링 Collaborative Filtering
  + 구매 및 소비한 제품에 대한 소비자의 평가 패턴이 비슷한 집단 속에서 서로 접하지 않은 제품을 추천하는 기술
* 내용 기반 필터링 Content-Based Filtering
  + 제품의 내용을 분석해서 추천하는 기술
* 지식 기반 필터링 Knowledge-Based Filtering
  + 특정 분야 전문가의 도움을 받아서 그 분야에 대한 “전체적인 지식 구조”를 만들어서 활용하는 방법
* 딥러닝 Deep Lerning
  + AI 알고리즘 중에 현재 가장 많이 사용되는 딥러닝 방법
* 하이브리트 필터링 CF + CBF 등
  + 두 가지 이상의 알고리즘 혼합을 통한 하이브리드 형태

2. 협업 필터링 추천 시스템

-> 어떤 아이템에 대해 비슷한 취향을 가진 사람들은 다른 아이템 또한 비슷한 취향을 가질 것이다.

원리 : 취향이 비슷한 사람들의 집단 존재 가정

3. 유사도 지표

-> CF에서 사용자간 유사도를 구하는 것이 핵심

* 1. 상관계수 : 가장 이해하기 쉬운 유사도(-1 ~ 1의 값)
  2. 코사인 유사도

1. 협업 필터링에서 가장 널리 쓰임
2. 각 아이템 -> 하나의 차원, 사용자의 평가값 -> 좌표값
3. 두 사용자의 평가값 유사 -> theta는 작아지고, 코사인 값은 커짐
4. -1 ~ 1 사이의 값
5. 데이터 이진값 -> 타니모토 계수 사용 권장
   1. 자카드 계수
6. 타니모토 계수의 변형 -> 자카드 계수
7. 이진수 데이터 -> 좋은 결과

4. 기본 CF 알고리즘

모든 사용자 간 평가의 유사도 계산 -> 추천 대상과 다른 사용자간 유사도 추출 -> 추천 대상이 평가하지 않은 아이템에 대한 예상 평가값 계산(평가값 = 다른 사용자의 평가 X 다른 사용자의 유사도) -> 아이템 중에서 예상 평가값 가장 높은 N개 추천

5. 이웃을 고려한 CF

- KNN 방법

6. CF 정확도 개선 방법

>최적의 이웃 K를 찾기!

>사용자의 평가 경향을 고려한 CF(우리는 별점이 없기 때문에 아직은 고려 X)

>신뢰도 가중을 통해 일정 가중치 이상 값들만 고려

7. 사용자 기반 CF와 아이템 기반 CF

* 사용자 기반UBCF

: 데이터가 풍부한 경우 정확한 추천 가능, 결과에 대한 위험성 존재

* + 데이터 크기 적고, 사용자에 대한 정보가 있는 경우 적절
* 아이템 기반 IBCF

: 계산이 빠름, 업데이트에 대한 결과 영향이 적음

* + 데이터 크기 크고, 충분한 정보가 없는 경우 적절 (ex 아마존, 넷플릭스)

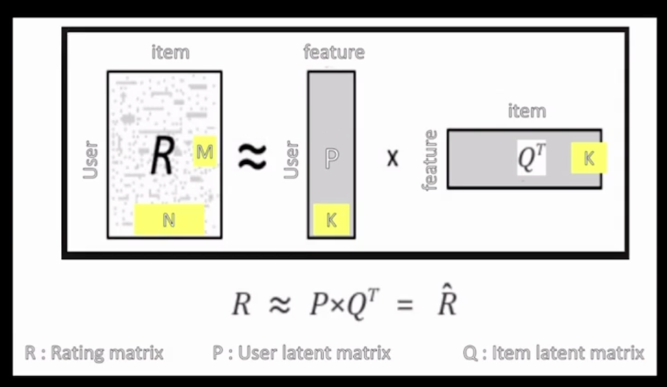
8. 추천 시스템의 성과측정지표

train/test set 분리 -> train set으로 학습 후, test set으로 평가 -> 각 아이템의 예상 평점과 실제 평점 차이를 계산 후 정확도 측정 or 추천 아이템과 사용자 실제 선택과 비교

* 메모리 기반 알고리즘(ex. CF)
  + 설명: 메모리에 있는 데이터를 계산해서 추천하는 방식
  + 특징: 개별 사용자 데이터 집중
  + 장점: 원래 데이터에 충실하게 사용
  + 단점: 대규모 데이터에 느리게 반응
* 모델 기반 알고리즘
  + 설명: 데이터로부터 미리 모델을 구성 후 필요 시 추천하는 방식
  + 특징: 전체 사용자 패턴 집중
  + 장점: 대규모 데이터에 빠르게 반응
  + 단점: 모델 생성 과정 오래 걸림

9. Matrix Factorization(MF) 기반 추천

MF(행렬분해) 방식의 원리



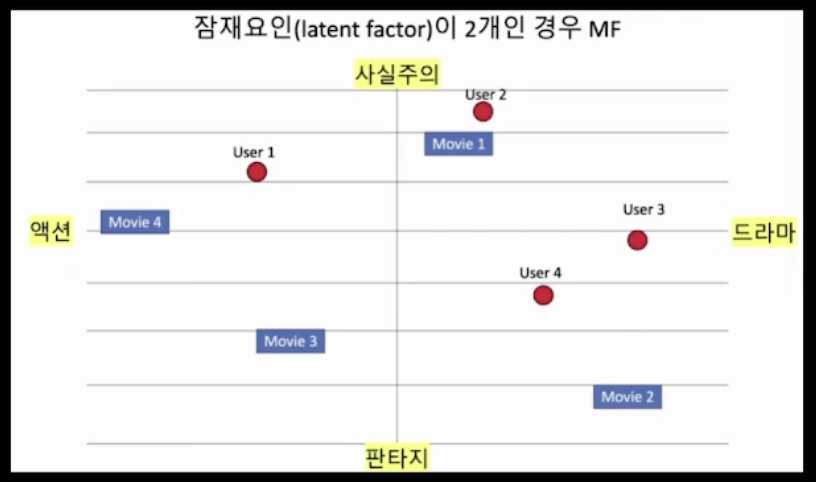
K : 잠재요인

ex) k = 2 일 떄,

테이블이(가) 표시된 사진

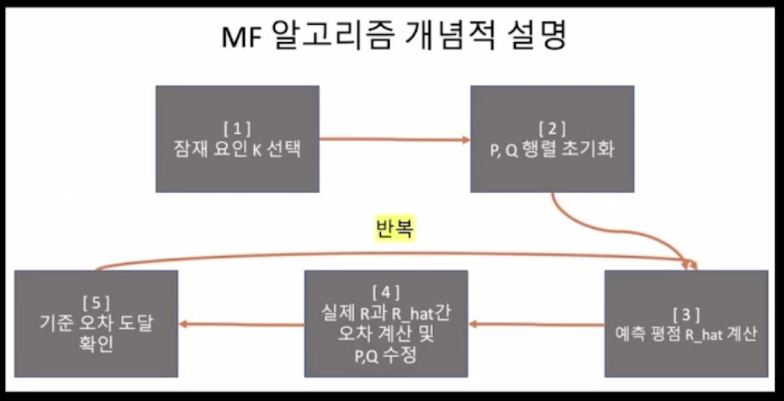
자동 생성된 설명 테이블이(가) 표시된 사진

자동 생성된 설명

 테이블이(가) 표시된 사진

자동 생성된 설명

10. SGD(Stochastic Gradient Decent): 확률적 경사 하강법

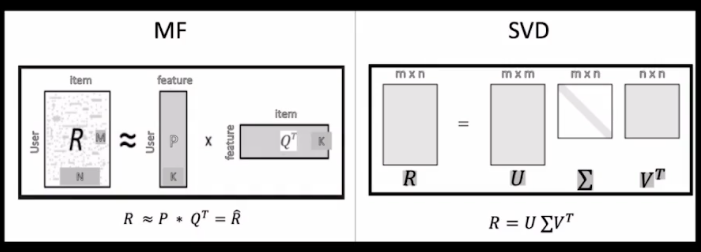


Overfitting 문제를 방지하기 위해서 regularization term을 추가해줌.

11. MF의 최적 파라미터 찾기

대략적인 최적의 K위치 찾기 -> 대략적 K 주변 탐색으로, 최적 K 찾기 -> 주어진 K를 통해 최적의 iterations 선택

12. MF와 SVD



* SVD(Singular Value Decomposition): 특이값 분해
  + 3개의 행렬로 분해한 후 학습 한 뒤, 원래 행렬로 만드는 방법
  + Null값 허용 X
  + 차원 축소에서 주로 사용, 추천 시스템에서는 잘 사용하지 않음

13. Surprise 패키지 사용

pip install scikit-surprise, conda install -c cond-forge scikit-surprise

surprise 패키지 내부 알고리즘

1. BaselineOnly: 사용자의 평점 평균과 아이템의 평점 평균 모델화 후 예측
2. KNNWithMeans: 사용자의 평가 경향까지 고려한 CF
3. SVD: MF기반 알고리즘
4. SVDpp: MF기반으로 사용자의 특정 아이템에 대한 평가여부를 이진값으로 추가한 SVD 모형

알고리즘 옵션 지정

* 유사도 지표 종류 : msd, cosine sim pearson sim, pearson baseline sim

다양한 조건의 비교도 가능함

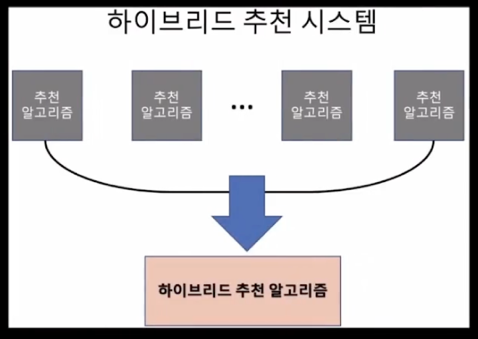
Ex) k 값, sim\_options의 name들(pearson\_baseline, cosine 등), user\_based(True,False) – False는 item\_based

외부 데이터 사용

Surprise 패키지의 Reader사용 : from surprise import Reader

14. 하이브리드 추천 시스템

다수의 추천 알고리즘 결합 -> 하이브리드 추천 알고리즘(하이브리드 추천 시스템)



장점 : 단순히 평균을 내는 것이 아니고, 알고리즘 간 상호 보완적이다. 즉, 한 알고리즘에서 잡아내지 못한 점을 다른 알고리즘으로 잡아낸다

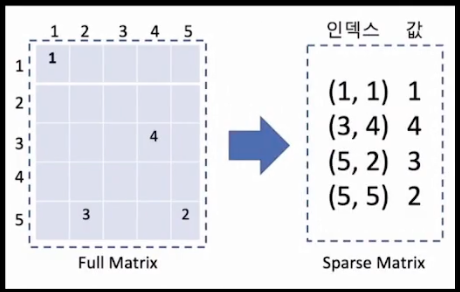
* 언제나 좋은 것만은 아님, 어떻게 결합하느냐에 따라 성능에 차이가 발생



15. Sparse Matrix(희소행렬)의 개념과 Python에서의 사용

추천 시스템의 Full Matrix는 대부분이 0값이므로 -> 메모리의 한계가 발생함

<실제 값을 가지고 있는 인덱스와 값을 저장하는 형식>



20개 원소 -> 4개 원소

데이터를 저장하거나 읽어올 때마다 값이 존재하는지 확인 후, 그에 대한 처리가 필요

= 데이터 처리에 대한 overhead cost(간접비)가 커짐

🡪 따라서 데이터가 너무 희박하지 않은 경우에는 굳이 사용하지 않는 것이 좋음.

변환방법

* COO(coordinate): 0이 아닌 데이터만 별도의 배열에 저장. 그 데이터가 가리키는 행렬의 위치를 별도의 배열에 저장하는 방식 (값 배열, 행 배열, 열 배열을 만들어 저장)
* CSR(compressed sparse row) : COO에서의 행 배열을 압축시켜서 더 효율적으로 저장하는 방식

Ex) coo의 행 배열이 [0,0,1,1,1,1,2,2] -> csr의 행 배열은 [0, 2, 6, 7]

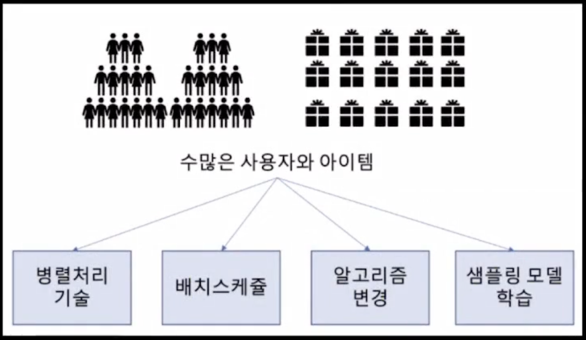
16. 실제 추천 시스템 구축 시 고려사항

* 신규 사용자와 아이템(Cold Start Problem): 신규 사용자와 아이템에 대한 추천은 어떻게 할 것인가

테이블이(가) 표시된 사진

자동 생성된 설명

* 확장성(Scalability): 대규모 사용자와 아이템을 대상으로 추천 알고리즘을 적용하기 위해서 확장성을 어떻게 확보할 것인가



병렬처리기술 : pyspark 사용,

배치 스케줄 : system trafic이 가장 적은 시간대(새벽 등) 돌려놓고 저장 후 추출

알고리즘 변경 : 성능을 낮추지만 계산량이 적은 알고리즘으로 변경, 계산량이 item\_base CF < user\_base CF이므로 이걸로 대체해서 사용

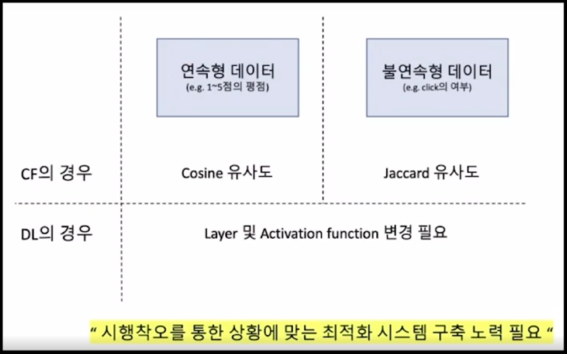
샘플링 모델 학습 : 대표적 사용자와 아이템을 선택해서 학습

* 추천의 활용(Presentation): 추천 알고리즘을 어떤 방식으로 사용할 것인가

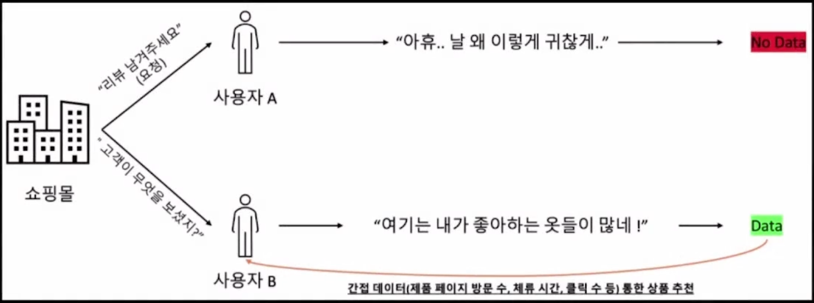
텍스트이(가) 표시된 사진

자동 생성된 설명

* 이진수 데이터의 사용(Binary Data): 데이터가 이진값인 경우는 어떻게 할 것인가



* 사용자의 간접 평가 데이터 확보(Indirect Evaluationn data): 사용자의 선호 데이터를 구하기 어려운 경우, 해당 데이터를 어떻게 구할 것인가



**Source.Inflearn**