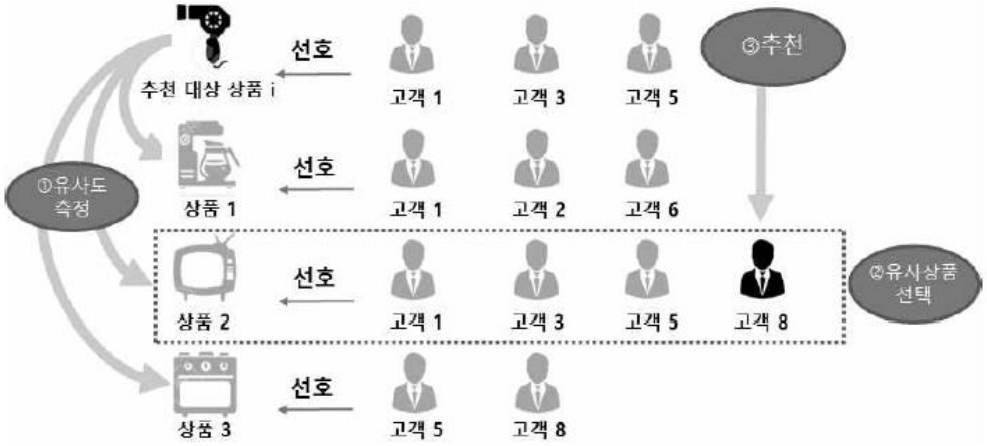
상품 기반 협업 필터링
-Surprise library

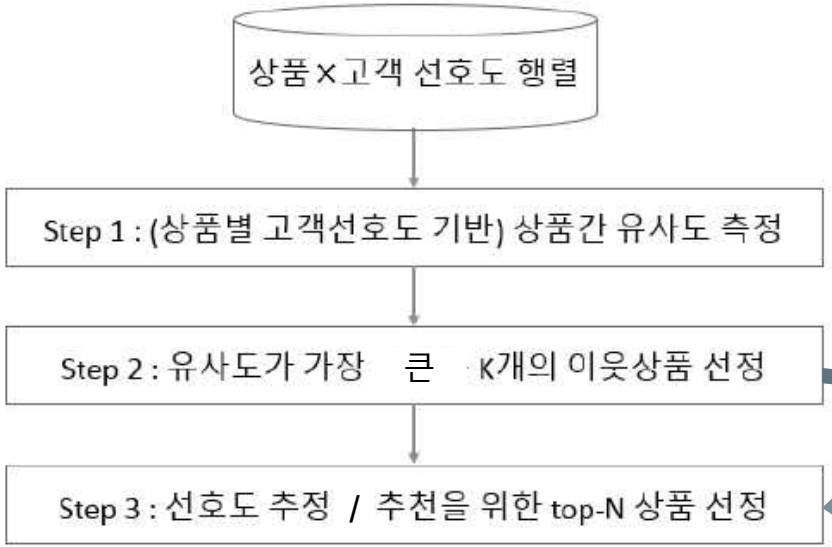
12기 정윤지

Contents

상품 기반 협업 필터링 알고리즘?



추천대상 상품 i를 구매하였거나 선호하는 고객목록을 살펴봄
→ 이 고객목록을 바탕으로 상품 i와 다른 상품 간의 유사도를 측정해 유사도가 가장 높은 유사상품을 선택
→ 상품 2를 구매한 고객 중에 아직 상품 i를 구매하지 않은 고객 8에게 상품 i를 추천



유사도 이용
유사한 상품들끼리 선호도가 비슷할 것이라는 가정

상품 간 유사도 측정 방법

Ex) 코사인 유사도

$$cosine\_sim(i,j) = \frac{\sum_{u \in U_{ij}} r_{ui} \times r_{uj}}{\sqrt{\sum_{u \in U_{ij}} r_{ui}^2} \times \sqrt{\sum_{u \in U_{ij}} r_{uj}^2}}$$

U_{ij} : 상품 i와 상품 j를 모두 평가한 고객 집합
r_{ui} : 고객 u의 상품 i에 대한 선호도
r_{uj} : 고객 u의 상품 j에 대한 선호도

고객 i 입장에서 두 상품이 비슷하게 느껴진다면 두 상품의 선호도가 비슷할 것이라는 가정
MSD, Pearson 다 이 기호들 이용

사용자의 아이템 선호도 추정 방법 : 앞에서 구한 유사도 이용

유사한 상품들끼리 선호도가 비슷할 것이라는 가정
결측치, 채워져 있는 값들 다 추정 가능

Ex) KNNBasic

k개의 이웃 상품 j들의 선호도 점수에 상품 i와의 유사도를 반영한 단순평균을 추정값으로 사용

$$\hat{r}_{ui} = \frac{\sum_{j \in N_i^k(i)} sim(i,j) \times r_{uj}}{\sum_{j \in N_i^k(j)} sim(i,j)}$$

Sim(I,j) : 앞의 방법들을 통해 측정한 유사도

$N_i^k(i)$: 상품 i와 가까운 상품들 중 rating 되어있는 최대 k개 상품들

KNNwithMeans, KNNwithZscore, KNNwithBaseline 다 이 기호들 사용

여기서 알 수 있는 것! 앞의 식을 잘 살펴보면…

Cold Start 문제
상품 i에 대한 rating(r_{ui})이 없으면 유사도 측정도, 선호도 추정도 할 수 없기 때문에 추천 불가

$$cosine\_sim(i,j) = \frac{\sum_{u \in U_{ij}} r_{ui} \times r_{uj}}{\sqrt{\sum_{u \in U_{ij}} r_{ui}^2} \times \sqrt{\sum_{u \in U_{ij}} r_{uj}^2}}$$

$$\hat{r}_{ui} = \frac{\sum_{j \in N_i^k(i)} sim(i,j) \times r_{uj}}{\sum_{j \in N_i^k(j)} sim(i,j)}$$

→ rating이 되어있는 상품들로만 구성된 데이터 이용!

모델 평가

1) 사용자가 실제 평가한 아이템에 대한 예상 평점을 예측하고 실제 평점과 차이 계산

⇒ 평가 지표가 작을수록 모형이 정확

a. $MAD = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m}(|r_{i,j}-p_{i,j}|\ if\ r_{i,j} \neq null, 0\ otherwise)}{k}$

b. $MSE = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m}((r_{i,j}-p_{i,j})^2 if\ r_{i,j} \neq null, 0\ otherwise)}{k}$

c. $RMSE = \sqrt{MSE} = \sqrt{\frac{\sum_{i=1}^{n}\sum_{j=1}^{m}((r_{i,j}-p_{i,j})^2 if\ r_{i,j} \neq null, 0\ otherwise)}{k}}$

k : 정확도 계산에 포함된 총 아이템 수
n : 검증용 데이터에 있는 사용자 수
m : 검증용 데이터에 있는 아이템 수
$r_{i,j}$: 사용자 i의 아이템 j에 대한 실제 평점
$p_{i,j}$: 사용자 i의 아이템 j에 대한 예상 평점

모델 평가

2) 추천한 아이템과 실제 선택된 아이템을 비교

a. **정확도(Precision, Accuracy)** = $\frac{\text{사용자가 실제 선택한 아이템 수}}{\text{전체 추천된 아이템 수}}$

b. **재현율(Recall)** = $\frac{\text{맞는 추천 아이템 수}}{\text{사용자가 선택한 전체 아이템 수}}$

c. **F 측정값(F measure)** = $\frac{2 \times precision \times recall}{precision + recall}$

Surprise library

Surprise library는 추천 알고리즘을 위한 파이썬 library (내장 함수들 보면 아마도 협업 필터링에 초점..?)

Prediction algorithm

- random_pred.NormalPredictor
- baseline_only.BaselineOnly
- knns.KNNBasic
- knns.KNNWithMeans
- knns.KNNWithZScore
- knns.KNNBaseline
- matrix_factorization.SVD
- matrix_factorization.SVDpp
- matrix_factorization.NMF
- slope_one.SlopeOne
- co_clustering.CoClustering

Model selection

- train_test_split
- KFold
- RepeatedKFold
- ShuffleSplit
- LeaveOneOut
- PredefinedKFold

Similarities module

- cosine
- msd
- pearson
- pearson_baseline

Accuracy module

- rmse
- mse
- mae
- fcp

Library 안에 저장되어 있는 Movie Lens dataset!
다양한 크기의 subset들 많음

협업 필터링 구현 시 kaggle에서 직접 다운받지 말고
이 data 이용하면 될 듯

https://grouplens.org/datasets/movielens/

MovieLens 100K Dataset

MovieLens 100K movie ratings. Stable benchmark dataset. 100,000 ratings from 1000 users on 1700 movies. Released 4/1998.

- README.txt
- ml-100k.zip (size: 5 MB, checksum)
- Index of unzipped files

Permalink: https://grouplens.org/datasets/movielens/100k/

MovieLens 1M Dataset

MovieLens 1M movie ratings. Stable benchmark dataset. 1 million ratings from 6000 users on 4000 movies. Released 2/2003.

- README.txt
- ml-1m.zip (size: 6 MB, checksum)

Permalink: https://grouplens.org/datasets/movielens/1m/

MovieLens 10M Dataset

MovieLens 10M movie ratings. Stable benchmark dataset. 10 million ratings and 100,000 tag applications applied to 10,000 movies by 72,000 users. Released 1/2009.

- README.txt
- ml-10m.zip (size: 63 MB, checksum)

코드 해석

더 공부해야 할 점

- 구현해보니 surprise library 내 함수들에 대한 이해 더 필요

- 각 사용자 별로 10개 영화를 추천해봤는데 영화 제목이 아닌 id로 출력해 영화 제목을 모름
  이를 좀 더 명확하게 정리한 결과를 얻고 싶음
  Ex)

```
특정 9번 유저가 본 영화 수: 46
추천한 영화 개수: 9696
전체 영화수: 9742

######## Top-10 추천영화 리스트 ########
* 추천 영화 이름:  Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb (1964)
* 해당 영화의 예측평점:  4.322402915230924

* 추천 영화 이름:  Philadelphia Story, The (1940)
* 해당 영화의 예측평점:  4.172423325516417
```

-Cross validation 등 이론들에 대한 이해 더 필요

-피어슨 유사도와 KNNwithMeans를 이용해봤음
하지만 실전에선 다양한 유사도 측정법, 선호도 추정법 별로 가장 정확한 추천을 해주는 기법 선정하고 싶음

model_type	similarity_option	k	train_rmse	test_rmse
KNNWithMeans	pearson	10	0.802911	1.268071
KNNWithZScore	pearson	10	0.803766	1.268229
KNNWithZScore	pearson	20	0.876658	1.270155
KNNWithMeans	pearson	20	0.875939	1.270204
KNNWithZScore	pearson	40	0.903048	1.274972
KNNWithMeans	pearson	40	0.902398	1.275318
KNNWithMeans	MSD	20	0.875226	1.278689
KNNWithMeans	MSD	10	0.789032	1.278780
KNNWithZScore	MSD	20	0.877514	1.281942

출처

https://towardsdatascience.com/how-to-build-a-memory-based-recommendation-system-using-python-surprise-55f3257b2cf4
https://surprise.readthedocs.io/en/stable/FAQ.html
https://surprise.readthedocs.io/en/stable/index.html
https://github.com/MatePocs/boardgame_recommendation/blob/master/02_modelling_neighbours.ipynb
https://notebook.community/zzsza/Datascience_School/
https://techblog-history-younghunjo1.tistory.com/117

Thank You!