RS: data preprocessing

20.07.30

컴퓨터과학전공 1715237 이혜승

Brunch data

2018.10.1 ~ 2019.3.14까지의 브런치 서비스에서 수집된 정보의 일부 데이터.

- 목표: 브런치 사용자를 위한 글 추천

- 파일 구성:

		#사용자id @읽은글들 @순서대로
read.tar	본(read) 글의 정보	- 읽은 (클릭한) 글이 순서대로 나열(사용자의 클릭 행동 순서O) - 머문 시간의 정보 or timestamp 존재하지 않기때문에 바로 이탈했는지 여부 알 수 없음.
metadata.json	글의 메타데이터	작가의 글들의 메타데이터 확인
/contents	글의 본문 정보	형태소 분석 결과 등의 본문 정보
users.json	사용자 정보	독자 및 작가의 사용자 정보
magazines.json	매거진 정보	

Brunch data

- metadata.json에 나온 '글'들과 read디렉토리 하위 파일들의 '글'과 비교하면,
  cold-start 문제 / long-tailed한 item(글)을 찾을 수 있지 않을까?

→ BUT, metadata 자체가 이미 2018.10.01 ~ 2019.03.14에 **독자들이 본 글**의 정보.

(대안)

1. Cold start item:
   - 최신 글
2. Long-tailed item:
   - Heavy user가 읽지 않은 글
   - 조회수가 낮은 글

Brunch data

- metadata.json에 나온 '글'들과 read디렉토리 하위 파일들의 '글'과 비교하면,
  cold-start 문제 / long-tailed한 item(글)을 찾을 수 있지 않을까?

→ BUT, metadata 자체가 이미 2018.10.01 ~ 2019.03.14에 **독자들이 본 글**의 정보.

(대안)

1. Cold start item:
   - 최신 글
2. Long-tailed item:
   - Heavy user가 읽지 않은 글
   - 조회수가 낮은 글

(방법)

1. 데이터 수집 당시 날짜 기준의 **최신 글**을 cold start item으로 간주하되,
   최신 글에 대한 정보는 활용X(조회수 등)
2. heavy user를 추려낸 후, 그들이 읽지 않은 글을 long-tailed item으로 간주
   어떤 user를 heavy user로 볼 것인지? 구독작가수 ↑ /조회한 글 ↑
3. 기간 내 조회수가 낮은 글을 long-tailed item으로 간주
   글의 조회수 column 생성해서, 조회수가 현저히 적어서 추천을 위한 충분한 평가가 이루어지지 못하는 item들을 추려내고자 함
   # 독자가 선호하는 글이 long tail에 존재할 수도 있기때문에 고려

Brunch data

2. heavy user를 추려낸 후, 그들이 읽지 않은 글을 long-tailed item으로 간주

   ① users: 구독하는 작가가 많은 user?

	keyword_list	following_list	id
0	[]	[@perytail, @brunch]	#901985d8bc4c481805c4a4f911814c4a
1	[]	[@holidaymemories, @wadiz, @sciforus, @dailydu...	#1fd89e9dcfa64b45020d9eaca54e0eed
2	[]	[@commerceguy, @sunsutu, @kakao-it, @joohoonja...	#1d94baaea71a831e1f33e1c6bd126ed5
3	[]	[@amberjeon48, @forsy20, @nemotokki, @hawann, ...	#04641c01892b12dc018b1410e4928c0d
4	[]	[@dwcha7342, @iammento, @kakao-it, @dkam, @ant...	#65bcaff862aadff877e461f54187ab62

Brunch data

2. heavy user를 추려낸 후, 그들이 읽지 않은 글을 long-tailed item으로 간주

   ① users: 구독하는 작가가 많은 user?

users
id
keyword_list
following_list

사용자 아이디

검색 키워드

구독 작가 목록

users	
user_id	
keyword_list	
following_list	
following_cnt	구독하는 작가 수
following_cnt_rank	구독하는 작가 수 랭킹 (얼마나 많이 구독하는 지)

Brunch data

2. heavy user를 추려낸 후, 그들이 읽지 않은 글을 long-tailed item으로 간주

② read: 조회한 글이 많은 user?

read
date
hour
user_id
article_id

접속 일자

접속 시간 구간
(1시간 동안)
사용자 아이디

해당 사용자가 읽은
글의 아이디들[@~, @~,]

read
date
hour
user_id
article_id
article_cnt

해당 사용자가 조회한 글의 수
(얼마나 많이 읽었는 지)

Brunch data

2. heavy user를 추려낸 후, 그들이 읽지 않은 글을 long-tailed item으로 간주

   ① users: 구독하는 작가가 많은 user?

   ② read: 조회한 글이 많은 user?

read
date
hour
user_id
article_id
article_cnt

users
user_id
keyword_list
following_list
following_cnt
following_cnt_rank

heavyuser
user_id
article_cnt
article_cnt_rank
following_cnt
following_cnt_rank

Read테이블은 user_id의 중복 존재(날짜, 시간이 다를 때 글을 읽은 경우)

→ 사용자 groupby를 하여 사용자 id의 누적 article_cnt를 구한 후 랭킹 column(article_cnt_rank)을 생성

Brunch data

2. heavy user를 추려낸 후, 그들이 읽지 않은 글을 long-tailed item으로 간주

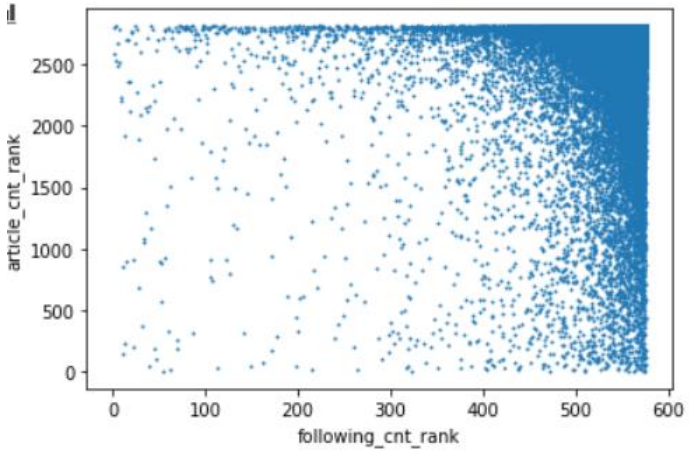
   ① users: 구독하는 작가가 많은 user?

   ② read: 조회한 글이 많은 user?

heavyuser
user_id
article_cnt
article_cnt_rank
following_cnt
following_cnt_rank

조회한 글의 수가 많은 사람을 heavy user로 볼 것인지, 혹은 구독한 작가의 수가 많은 사람을 heavy user로 볼 것인지를 정해야 함.

혹은 적절히 두 column을 잘 활용해보고자, 상관 관계를 시각화해봄.
(article_cnt_rank와 following_cnt_rank가 함께 증가하기를 바랐으나…)



Read테이블은 user_id의 중복 존재(날짜, 시간이 다를 때 글을 읽은 경우)

→ 사용자 groupby를 하여 사용자 id의 누적 article_cnt를 구한 후 랭킹 column(article_cnt_rank)을 생성

Brunch data

(방법)

1. 데이터 수집 당시 날짜 기준의 최신 글을 cold start item으로 간주하되, **최신 글**에 대한 정보는 활용X(조회수 등)

: 글이 작성된 시간의 정보(reg_ts)활용

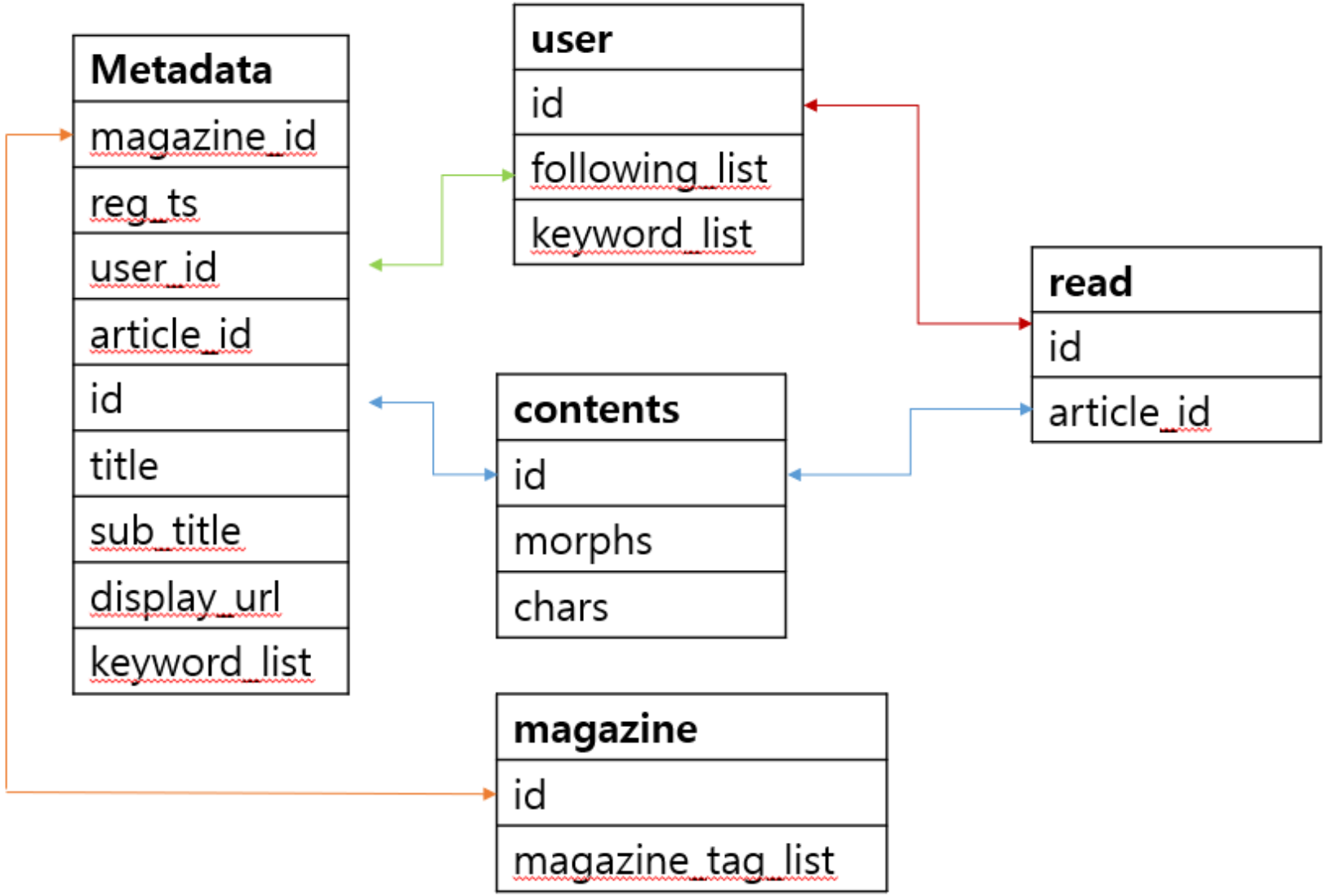
: 최신 글이란 얼마나 최신 글을 할 것인지?, reg_ts = 0으로 입력된 데이터 오류의 경우 어떻게 처리할 것인지?

3. 기간 내 **조회수가 낮은 글**을 long-tailed item으로 간주

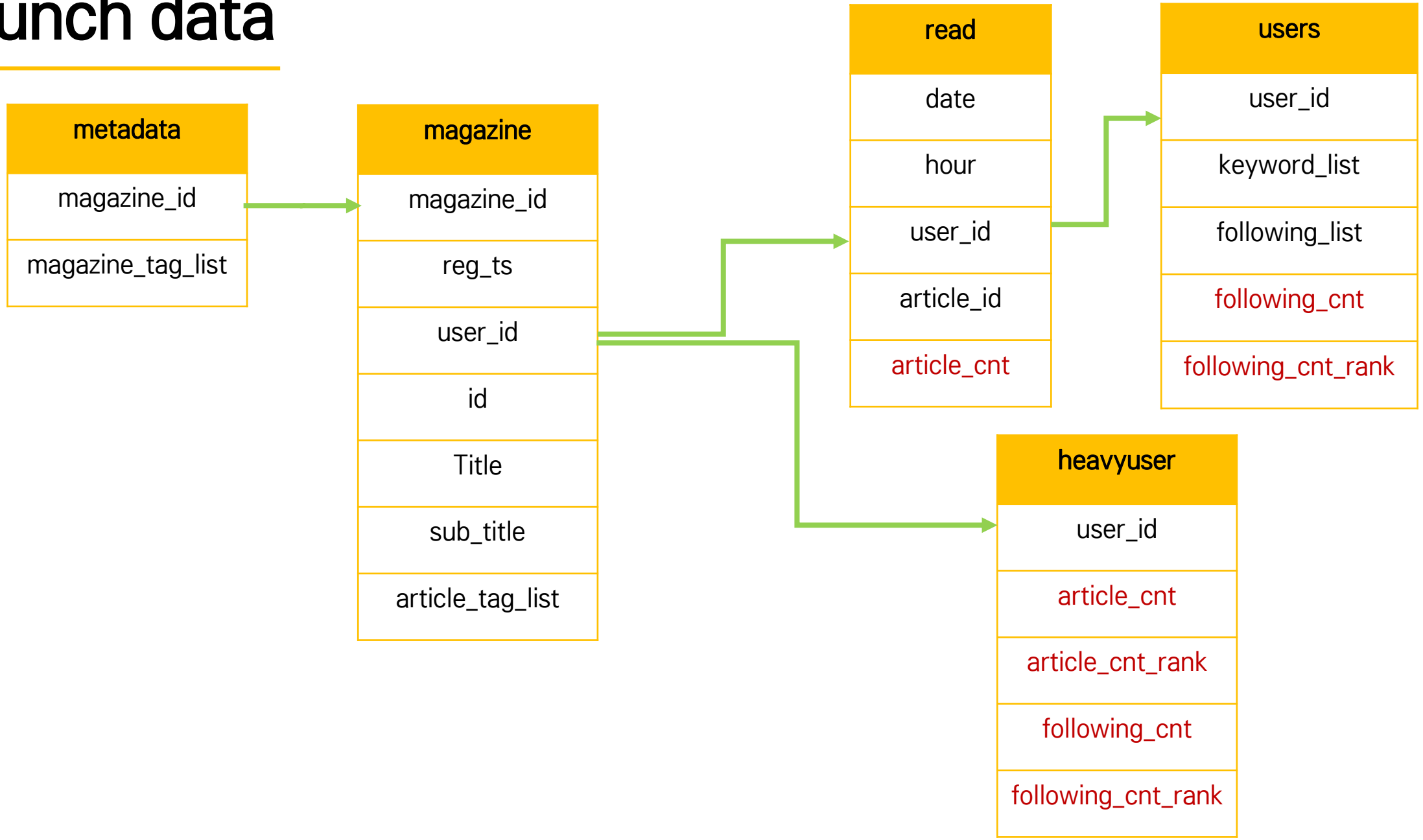
: (아직 안 만듦) read.article_id의 글 목록을 하나씩 뽑아내서, read.article_id에 isin이면 조회수+1

Brunch data

Original data:



Brunch data


metadata
magazine_id
magazine_tag_list
magazine
magazine_id
reg_ts
user_id
id
Title
sub_title
article_tag_list
read
date
hour
user_id
article_id
article_cnt
users
user_id
keyword_list
following_list
following_cnt
following_cnt_rank
heavyuser
user_id
article_cnt
article_cnt_rank
following_cnt
following_cnt_rank

Brunch data

Data preprocessing 남은 일:

- 글의 조회수 추출
    - 다른 사람 코드: 조회수를 전체 조회수와, 최근 조회수로 나누어서 반영

- Reg_ts(글 작성시간)을 date형식으로 변환하여, 최신 글 알아내기 # metadata['reg_ts'].apply(lambda x: int(datetime.datetime.fromtimestamp(x/1000).strftime('%Y%m%d')))

    → 값이 0이면, 1970.01.01이라는 의미. 데이터 수정 필요

- Read에는 있지만 metadata에 없는 글 정보 존재 (비공개 전환, 삭제 등의 이유)

    → 분석 불가하므로 해당 레코드는 삭제

Brunch data

분석 방법:

1. contents/ 데이터: 참여한 사람들이 활용하지 않음.
2. 알리바바: Item CF recommendations are very novel and very good at recommending items in the long tail.

   아이템 기반 CF:
3. ㅇㅇ

reference

- Kakao arena 참여 팀 깃헙1 [github: hyeonho1028]: 데이터 loading에 참고
- Kakao Arena 2nd Competition[Arena: Brunch Article Recommedations]
- Kakao arena 깃헙[github:kakao arena]: 베이스라인 제공
- 참여팀 깃헙2[github: jihoo-kim]
- 참여팀 깃헙3[github: yeonmin]