

AI LOVE YOU^o

Brunch 데이터를 활용해 사용자의 취향에 맞는 작가 및 글 예측하기

안녕하세요.
인공지능은 너를 사랑해 AI LOVE YOU 입니다.

지수영



재료공학, 생명과학,
경영정보학, 컴퓨터공학이
어우러진

AI LOVE YOU

안태용



유진아



이찬영



CONTENT^o

AI LOVE YOU

플젝 소개

01

분석 배경

02

분석 과정

03

분석 결과

04

활용 방안

05

01.

프로젝트 소개

프로젝트 주제, 목적, 방향

Mid Project

글이 작품이 되는 공간, 브런치



Magazine

주제

Subject

Brunch 데이터를 활용해
사용자의 취향에 맞는 작가 및
글 예측하여 추천해주기

목적

Subject

관심 글과 관심 작가를 바탕으
로 취향에 맞는 글과 작가 추천
을 통해 검색 소요시간 줄이기

방향

Subject

작가들이 쓴 글 +독자가 읽은 글
+작가의 키워드
=>유사도 분석
=>관심 작가 및 글 추천

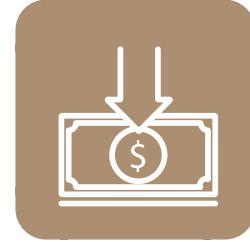
02.

분석 배경

주제 선정 과정 및 데이터 소개

주제 선정 과정

Notion을 이용한 순위 선정



해외 시장 분석하여 마케팅 포지셔닝

- 장점: 흥미있는 주제, 사회적 필요성 부각
- 단점: 경영학적 분석 결과 도출, 외국 사이트 크롤링 필요, 비지도학습에 대한 평가 어려움



기사 악플 분류하기

- 장점: 라벨링이 되어있음, 예측 모델 제공
- 단점: 욕설 필터링 분류 어려움, 딥러닝 필요



브런치 데이터를 이용한 취향에 맞는 글 예측

- 장점: 카카오 아레나 주제, 국내 회사 실무 체험
- 단점: 한글 자연어 처리의 어려움, 데이터의 크기

Brunch 데이터를 활용해 사용자의 취향에 맞는 작가 및 글 예측하기

사용 데이터

read.tar

“ 2018.10.01~2019.03.01

일부 브런치 독자들이 본 글의 정보

파일 이름: '시작일_종료일'

총 3,625개 파일 ”

user.json

“ ['keyword_list', 'following_list', 'id']

-가입한 사용자 정보

-keyword_list: 최근 며칠간 작가 글로

유입된 검색 키워드

-following_list: 구독 작가 리스트

-id: 사용자 식별자

총 310,758명의 정보 ”

megazine.json

“ ['megazine_tag_list', 'id']

-id: 메거진 식별자

-magazine_tag_list: 작가가 부여한 메거진의 태그 정보

총 27,967개의 브런치 메거진 정보 ”

metadata.json

2019.10.01~2019.03.14

megazine_id

이 글의 브런치 메거진 아이디
(없을 시 0)

user_id

작가 아이디

id

글 식별자

sub_title

부제목

reg_ts

이 글이 등록된 시간
([유닉스 시간], 밀리초)

article_id

글 번호

title

제목

display_url

웹주소

keyword_list

작가가 부여한 글의 태그 정보

03.

분석 과정

데이터 전처리, TFIDF, 코사인 유사도를
활용한 데이터 분석 과정

추천에 사용될 요약된 데이터

● [작가 4000명의 글의 키워드에 관한 요약]

```
1 writer_4000 = pd.read_csv(directory+'writer_4000most3_keyword_keywords.csv', index_col = 'Unnamed: 0')
2 writer_4000
```

	writer_id	page_num	keyword_list	keywords
0	@bookfit	4106	['성공', '행복', '인생']	['창업', '비즈니스', '문제', '경제', '살림살이', '경제학', '미세먼...']
1	@wikitree	3192	['영화', '뉴스', '연합뉴스']	['지휘관', '장교', '청계천', '서울', 'IT', '번호', '중소기업', ...]
2	@jordan777	2797	['중국', '미국', '트럼프']	['석유에너지', '베네수엘라', '경제', '일자리', '청년실업', '저출산', ...]
3	@hitchwill	2115	['국내여행', '영화', '여행']	['국내여행', '한국', '청양', '여행', '가을', '코스모스', '감귤농장...']
4	@tenbody	1759	['다이어트', '운동', '건강']	['운동', '근육', '다이어트', '운동', '뱃살', '다이어트', '운동', ...]
...
3995	@starshines	39	['영화', '사랑', '전쟁']	['영화', '저니스엔드', '전쟁', '영화', '하정우', '민주주의', '히트...']
3996	@jeolmiing	39	['독서', '독서모임', '인터뷰']	['독서', '일상', '인터뷰', '후기', '독서모임', '가이드', '인터뷰', ...]
3997	@prague	39	['프라하', '체코', '유럽여행']	['프라하', '유럽여행', '체코', '유럽여행', '프라하', '체코', '프라...']
3998	@tilltue	39	['IT', 'iOS', '개발']	['테스트코드', '테스트', 'IT', 'IT', 'iOS', 'IT', '테스트...', ...]
3999	@moeanee	39	['에세이', '사랑', '감성']	['마음', '독서', '에세이', '브런치X어라운드', '생각', '에세이', '...']

‘‘[유저 100명의 팔로우 리스트에 있던 작가들의 키워드]’

1 user

	user_id	keyword_list
0	#00001ba6ca8d87d2fc34d626ba9cfe6f	[사랑, 성공, 인생, 브런치, 작가, 글쓰기, 해외취업, 취직, 싱가포르]
1	#0000e87158c1426d6ffb72cebac6cb64	[브런치, 가해자, 여성혐오, 페미니스트]
2	#0000eea6d339abfd02ed590bc451fc63	['인간관계', '해외생활', '회사']
3	#0000fdb8f35c76eacab74c5c6bc7f1a	['도쿄', '여행', '그림에세이', '도쿄', '여행', '그림에세이', '운동...']
4	#000127ad0f1981cae1292efdb228f0e9	['사랑', '결혼', '결혼생활', '사랑', '결혼', '결혼생활', '연애',...]
...
95	#00160ec29677bde7de10cff7c24ee52	['절약', '가성비', '중고', '절약', '가성비', '중고', '달러', '...']
96	#0016322831caa01cc9338a1f789080e1	['대장암', '대장암초기증상', '혈변', '여행경비', '세계여행', '토사',...]
97	#001634052c50355eb7acc2873c2ebe13	['비전공자', '디자인', '커리큘럼', '육아에세이', '행복', '공감에세이',...]
98	#00169d555ccb64b6fd923c4966cbfb98a	['광고', '네이버', '마케팅', '카카오뱅크', 'UX', '인터넷은행', '...']
99	#00169d555ccb64b6fd923c4966cbfb98a	['네이버', '네이버', '마케팅', '카카오뱅크', 'UX', '인터넷은행', '...']

100 rows x 3 columns

Word Cloud

● ● [작가들이 많이 사용한 키워드]



[유저 100명의 팔로우 리스트에 있던 작가들의 키워드]



코사인 유사도 분석 결과 - 함수화

TF-IDF

독자가 읽은 글 기반

1 reader_read(1)

#0000e87158c1426d6fffb72cebac6cb64님께 추천드리는 작가는 @goodwriting, @metamon, @brunch 입니다.

```
14 def reader_read(i):
15     # 독자데이터 vectorization
16     read_key = user['keyword_list']
17     read_key[i] = eval(read_key[i])
18     read1 = ' '.join(read_key[i])
19     reader = vectorizer.transform([read1]).todense()
20
21     # Cosine similarity 계산
22     cosine_similarity(reader, X)
23     result = cosine_similarity(reader, X)
24
25     # 결과 값 도출
26     global writer_4000
27     writer_4000['read_sim'] = result[0]
28     writer_4000.sort_values(by='read_sim', ascending=False)
29     writer_4000 = writer_4000.sort_values(by='read_sim', ascending=False)
30     writer_top3 = writer_4000.head(3)
31     writer_results = writer_top3['writer_id'].tolist()
32     print("%s님께 추천드리는 작가는 %s, %s, %s 입니다."%(user['user_id'][i], writer_results[0], writer_results[1], writer_results[2]))
```

코사인 유사도 분석 결과 - 함수화

TF-IDF

독자가 구독한 작가 기반

```

1 writers_results = reader_follow(1)
2 writers_results
3 print("%s님께서 팔로우하시는 작가 기반으로 추천드리는 작가는 %s, %s입니다."%(following['id'][1], writers_results[0], writers_results[1])
#1fd89e9dcfa64b45020d9eaca54e0eed님께서 팔로우하시는 작가 기반으로 추천드리는 작가는 @glgom, @hapi19입니다.

```

	writer_id	page_num	keyword_list	keyword_list_20	keywords	read_sim
2978	@goodwriting	52	['페미니즘', '여성혐오', '페미니스트']	['페미니즘', '여성혐오', '페미니스트', '기자', '여성', '감정', '노...']	['차별', '페미니즘', '여성혐오', '교회', '페미니즘', '페미니스트', ...]	0.293767
1772	@metamon	83	['페미니즘', '교육', '페미니스트']	['페미니즘', '교육', '페미니스트', '생각', '사회', '에세이', '철학...']	['도전', '성장', '생각', '생각', '선택', '강연', '진리', '철학...']	0.260030
824	@brunch	142	['브런치', '작가', '인터뷰']	['브런치', '작가', '인터뷰', '브런치북', '토크콘서트', '출간', '출...']	['인터뷰', '작가', '아나운서', '작가', '인터뷰', '브런치', '브런치...']	0.199536
3010	@youngmusic	52	['페미니즘', '여성혐오', '가부장제']	['페미니즘', '여성혐오', '가부장제', '페미니스트', '성평등', '성차별...']	['나문희', '가부장제', '페미니즘', '다이어트', '페미니즘', '헝거', ...]	0.183588
78	@yangpayangpa	550	['페미니즘', '여혐', '육아']	['페미니즘', '여혐', '육아', '연애', '결혼', '여성혐오', '여성', ...]	['번역', '노동자', '일자리', '올림픽', '러시아', '독재자', '취업', ...']	0.165463
...
1563	@yettie	92	['행복', '성공', '생각']	['행복', '성공', '생각', '인간관계', '인생', '실천', '목표', '...']	['행복', '인생', '습관', '생각', '마음', '행복', '인간관계', '...']	0.000000

코사인 유사도 분석 결과 - 함수화

TF-IDF

추천 받은 작가 기반

```
1 article = recoomd_article(writers_results, 1)
2 print("%s님께서 추천 받은 작가 중에서 취향에 맞는 글은 '%s' 입니다."%(following['id'][1], article))
```

#1fd89e9dcfa64b45020d9eaca54e0eed님께서 추천 받은 작가 중에서 취향에 맞는 글은 '이 관계를 망치는 것 같지' 입니다.

magazine_id	user_id	title	keyword_list	display_url	sub_title	reg_ts	article_id	id	upload_time
14690	25818	@hapi19	널 지 워 버 렸 기 때 문 에	[관계]	https://brunch.co.kr/@hapi19/288		1549907802000	288	@hapi19_288 2019-02-11 17:56:42
91163	25818	@hapi19	여 러 번 사 랑 을 하	[감성에세이, 공감에세이]	https://brunch.co.kr/@hapi19/287	나도	1549207836000	287	@hapi19_287 2019-02-03 15:30:36

04.

분석 결과

TFIDF 및 코사인 유사도 분석 결과

워드 클라우드 시각화

코사인 유사도 분석 결과 - 독자가 읽을 글 기반 키워드 분석

```
1 writer_4000.sort_values(by='read_sim', ascending=False)
```

독자 1의 관심 키워드

▶ ['사랑 성공 인생 브런치 작가 글쓰기 해외취업 취직 싱가포르']

	writer_id	page_num	keyword_list	keywords	read_sim
3487	@singayoung	44	['해외취업', '싱가포르', '취업']	['면접', '싱가포르', '구직', '해외', '백수', '자기소개', '인맥', ...]	0.583730
726	@swimmingstar	156	['유럽여행', '해외취업', '싱가포르']	['싱가포르', '해외취업', '회사', '도서관', '코펜하겐', '음악', '프...']	0.449799
824	@brunch	142	['브런치', '작가', '인터뷰']	['인터뷰', '작가', '아나운서', '작가', '인터뷰', '브런치', '브런치...']	0.322218
3721	@isntitallright	42	['싱가포르', '식단', '운동']	['컬러런', '싱가포르', '축제', '싱가포르', '해외취업', '일기', 'f...']	0.301686
3303	@yeonsilyoo	47	['해외취업', '취업', '해외']	['마케팅', '페이스북', '스타트업', '사업', '창업', '스타트업', '런...']	0.300144
...
2564	@embrassemoi93	59	['생각', '고양이', '달래']	['고양이', '달래', '동행', '여행', '센과치히로의행방불명', '눈물', ...]	0.000000
3248	@bestwriter1lf1	48	['맛집', '남포동', '영화리뷰']	['영도', '삼계탕', '맛집', '코코', '영화리뷰', '애니메이션', '다큐...']	0.000000
640	@unitimes	171	['문장', '결핍', '타투스토리후']	['문장', '결핍', '타투스토리후', '타투잘하는곳', '음악', '감각', '...']	0.000000
3251	@oliverpark	48	['서울', '골목길', '사진']	['은퇴', '글씨체', '손글씨', '산책', '교차로', '일요일', '주말', ...]	0.000000
3081	@readme999	50	['증여세', '은행', '바보아저씨']	['담보대출', '총부채상환비율', '주택담보대출', '증여세', '증여세계산방법'...]	0.000000

4000 rows × 5 columns

코사인 유사도 분석 결과 - 독자가 팔로우한 작가 기반 키워드 분석

```
1 writer_4000.sort_values(by='follow_sim', ascending=False)
```

독자 1의 관심 키워드

▶ ['사랑 성공 인생 브런치 작가 글쓰기 해외취업 취직 싱가포르']

writer_id	page_num	keyword_list	keywords	read_sim	follow_sim	
824	@brunch	142	['브런치', '작가', '인터뷰']	['인터뷰', '작가', '아나운서', '작가', '인터뷰', '브런치', '브런치...']	0.322218	1.000000
2325	@siso-writers	65	['작가', '글쓰기', '출판']	['편집자', '출판', '영상', '편집자', '출판', '에디터', '편집자', ...]	0.228795	0.354143
1905	@thecommaa	78	['인터뷰', '디지털노마드', '디자인']	['스타벅스', '디지털노마드', '수필', '인터뷰', '수영복', '패션', '...']	0.014125	0.329961
3071	@theeee	50	['인터뷰', '생각', '독서모임']	['미래', '기회', '선택', '독후감', '힘', '관계', '인터뷰', '질...']	0.009296	0.317211
217	@ihearyou	333	['음악', '인터뷰', '리뷰']	['음악', '인터뷰', '사람', '음악', '음원차트', '롱런', 'OST', ...]	0.086460	0.246591
...	
2788	@wineflora	55	['와인', '보르도', '프랑스']	['산책', '메이플', '샴페인', '와인여행', '와인', '미국와인', '와인...']	0.000000	0.000000
2442	@jw2022	62	[]	[]	0.000000	0.000000
2794	@mindgolf	55	['마인드골프', '골프', '골프컬럼']	['골프', '마인드골프', '에티켓', '골프컬럼', '골프', '마인드골프', ...]	0.000000	0.000000
498	@imymemine89	200	['필사', '시']	['필사', '시']	0.000000	0.000000
1695	@gjgksk	86	[]	[]	0.000000	0.000000

4000 rows × 6 columns

함수화 - 독자가 읽은 글 기반 작가 추천

```

1 # 작가 데이터 vectorization
2 writer_key = []
3
4 for i in range(4000):
5     writer_key.append(' '.join(eval(writer_4000['keywords'][i])))
6
7 corpus = writer_key
8 vectorizer = TfidfVectorizer()
9 X = vectorizer.fit_transform(corpus).todense()
10 pd.DataFrame(X)
11
12
13 def reader_read(i):
14     # 독자 데이터 vectorization
15     read_key = user['keyword_list']
16     read_key[i] = eval(read_key[i])
17     read1 = ' '.join(read_key[i])
18     reader = vectorizer.transform([read1]).todense()
19
20     # Cosine similarity 계산
21     cosine_similarity(reader, X)
22     result = cosine_similarity(reader, X)
23
24     # 결과 값 도출
25     global writer_4000
26     writer_4000['read_sim'] = result[0]
27     writer_4000.sort_values(by='read_sim', ascending=False)
28     writer_4000 = writer_4000.sort_values(by='read_sim', ascending=False)
29     writer_top3 = writer_4000.head(3)
30     writer_results = writer_top3['writer_id'].tolist()
31     print("%s님께 추천드리는 작가는 %s, %s, %s 입니다."%(user['user_id'][i], writer_results[0], writer_results[1], writer_results[2]))
32

```

```

1 reader_read(1)

```

#0000e87158c1426d6ffb72cebac6cb64님께 추천드리는 작가는 @goodwriting, @metamon, @brunch 입니다.

함수화 - 독자가 팔로우한 작가 기반 작가 추천

```
1 user['user_id'][1]
```

```
'#0000e87158c1426d6ffb72cebac6cb64'
```

```
1 def reader_follow(i):
2     global writer_4000
3
4     # 독자 데이타 vectorization
5     list_t = get_user_following_author_keywords(following['id'][i])
6     read_to_list = [l + " " for l in list_t.split(" ")][:-1]
7     read_to_list = [eval(l) for l in read_to_list]
8     read2 = ' '.join(read_to_list[0])
9     reader2 = vectorizer.transform([read2]).todense()
10
11     # Cosine similarity 계산
12     result_follow = cosine_similarity(reader2, X)
13
14     # 결과 값 도출
15     writer_4000['follow_sim'] = result_follow[0]
16     writer_4000 = writer_4000.sort_values(by='follow_sim', ascending=False)
17     writers_top3 = writer_4000.head(3)
18     writers_results = writers_top3['writer_id'].tolist()
19     print("%s님께서 팔로우하시는 작가 기반으로 추천드리는 작가는 %s, %s, %s 입니다."%(user['user_id'][i], writers_results[0], writers_r
```

```
1 reader_follow(1)
```

#0000e87158c1426d6ffb72cebac6cb64님께서 팔로우하시는 작가 기반으로 추천드리는 작가는 @g1gom, @hapi19, @giewookkoo 입니다.

분석 결과 - 함수화

```

1 # 작가 데이터 vectorization
2 writer_key = []
3
4 for i in range(4000):
5     writer_key.append(' '.join(eval(writer_4000['keywords'][i])))
6
7 corpus = writer_key
8 vectorizer = TfidfVectorizer()
9 X = vectorizer.fit_transform(corpus).todense()
10 pd.DataFrame(X)
11
12
13 def reader_read(i):
14     # 독자 데이터 vectorization
15     read_key = user['keyword_list']
16     read_key[i] = eval(read_key[i])
17     read1 = ' '.join(read_key[i])
18     reader = vectorizer.transform([read1]).todense()
19
20     # Cosine similarity 계산
21     cosine_similarity(reader, X)
22     result = cosine_similarity(reader, X)
23
24     # 결과 값 도출
25     global writer_4000
26     writer_4000['read_sim'] = result[0]
27     writer_4000.sort_values(by='read_sim', ascending=False)
28     writer_4000 = writer_4000.sort_values(by='read_sim', ascending=False)
29     writer_top3 = writer_4000.head(3)
30     writer_results = writer_top3['writer_id'].tolist()
31     print("%s님께 추천드리는 작가는 %s, %s, %s 입니다."%(user['user_id'][i], writer_results[0], writer_results[1], writer_results[2]))
32
33
34 reader_read(1)

```

#0000e87158c1426d6ffb72cebac6cb64님께 추천드리는 작가는 @goodwriting, @metamon, @brunch 입니다.

분석 결과 - 함수화

독자가 구독한 작가 기반 계산

```
1 user['user_id'][1]
```

```
'#0000e87158c1426d6ffb72cebac6cb64'
```

```

1 def reader_follow(i):
2     global writer_4000
3
4     # 독자 대 OIEI vectorization
5     list_t = get_user_following_author_keywords(following['id'][i])
6     read_to_list = [l + "]" for l in list_t.split(",")][:-1]
7     read_to_list = [eval(l) for l in read_to_list]
8     read2 = ' '.join(read_to_list[0])
9     reader2 = vectorizer.transform([read2]).todense()
10
11    # Cosine similarity 계산
12    result_follow = cosine_similarity(reader2, X)
13
14    # 결과 값 도출
15    writer_4000['follow_sim'] = result_follow[0]
16    writer_4000 = writer_4000.sort_values(by='follow_sim', ascending=False)
17    writers_top3 = writer_4000.head(3)
18    writers_results = writers_top3['writer_id'].tolist()
19    print("%s님께서 팔로우하시는 작가 기반으로 추천드리는 작가는 %s, %s, %s 입니다."%(user['user_id'][i], writers_results[0], writers_r

```

```
1 reader_follow(1)
```

```
#0000e87158c1426d6ffb72cebac6cb64님께서 팔로우하시는 작가 기반으로 추천드리는 작가는 @glgom, @hapi19, @giwookkoo 입니다.
```

05.

활용 방안

분석 결과를 바탕으로 한
추후 활용 방안 및 계획

소비자 선호를 파악하여 브런치 내 양질의 콘텐츠 생성에 기여

활용도

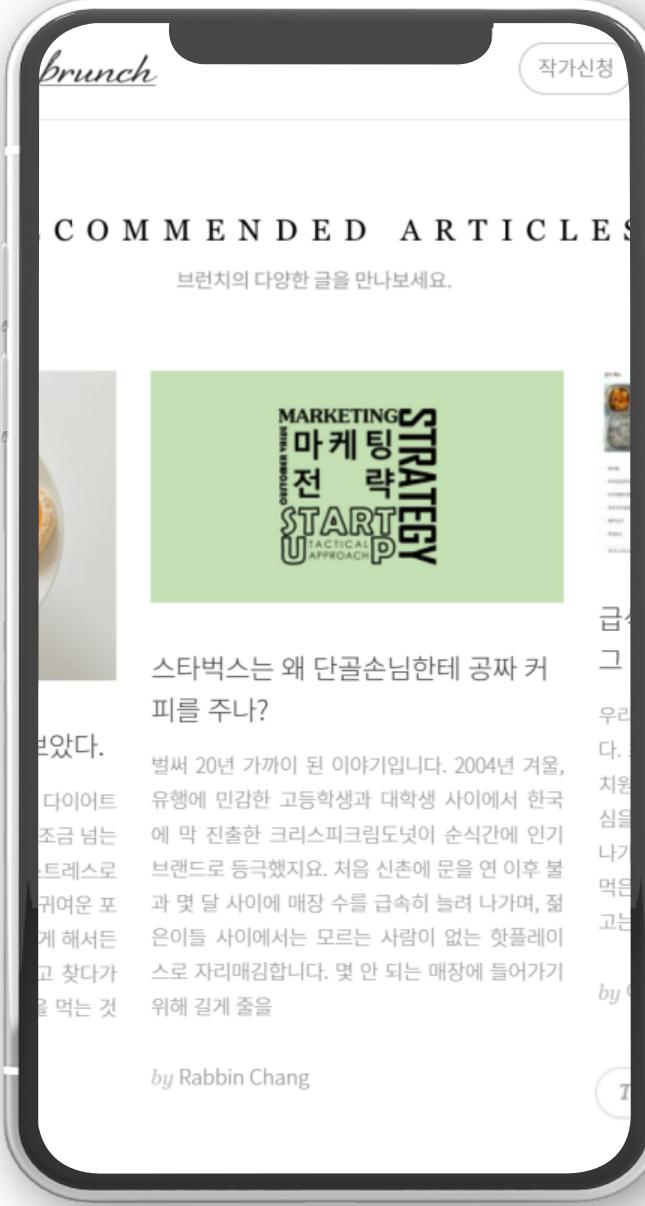
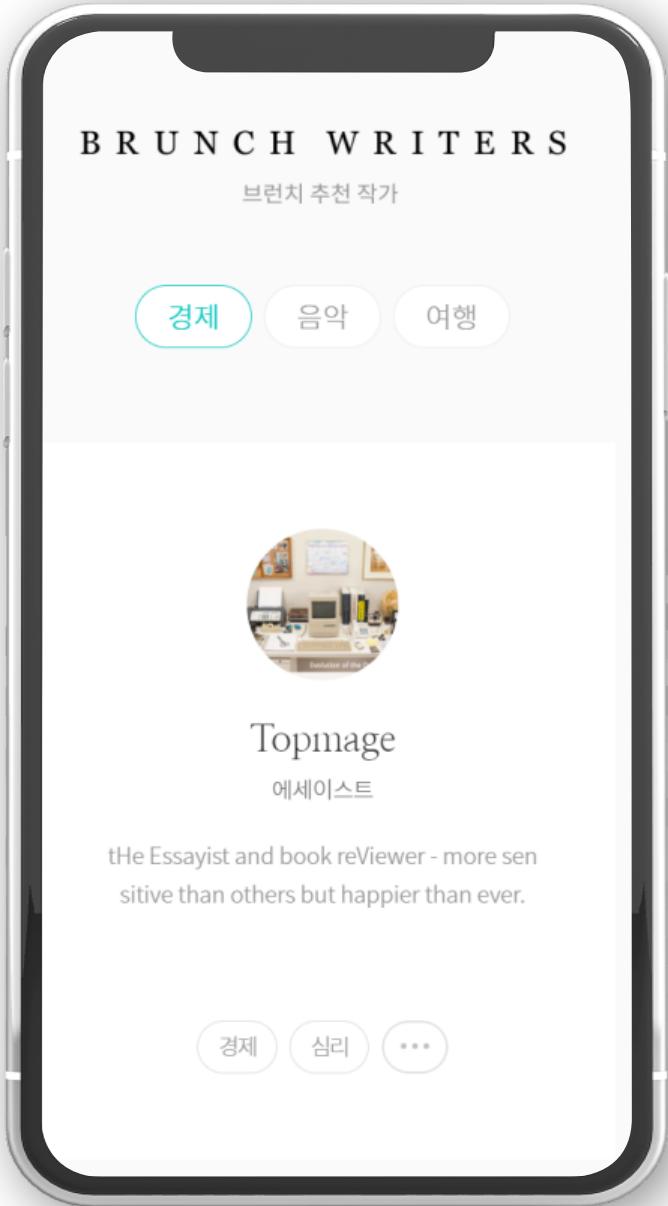
향상



추천



유입



<작가 추천>

<글 추천>

• 발전 방향

협업필터링 등의 추천 알고리즘을 사용하여 사용자의 소비맥락에 맞는 글 추천 시스템 구현

• 작가 기대효과

소비자 선호를 파악하여 글 작성 가능
소비자 선호에 맞춘 보다 풍부한 콘텐츠 제공

• 유저 기대효과

맞춤형 글 추천을 통해 브런치 내 순환율 향상
브런치 유입 증가 기대

Q&A



AI LOVE YOU

THANK YOU°

📞 AI LOVE YOU

지수영, 안태용

유진아, 이찬영