

Chapter 10

추천 시스템

추천시스템의 중요성

아마존 등과 같은 전자상거래 업체부터 넷플릭스, 유튜브, 애플 뮤직 등 콘텐츠 포털까지 추천 시스템을 통해 사용자의 취향을 이해하고 맞춤 상품과 콘텐츠를 제공해 조금이라도 오래동안 자기 사이트에 고객을 머무르게 하기 위해 전력을 기울이고 있습니다



아마존 : 리뷰/평점 기반의 추천시스템

파레토와 롱테일의 법칙



파레토의 법칙

롱테일의 법칙

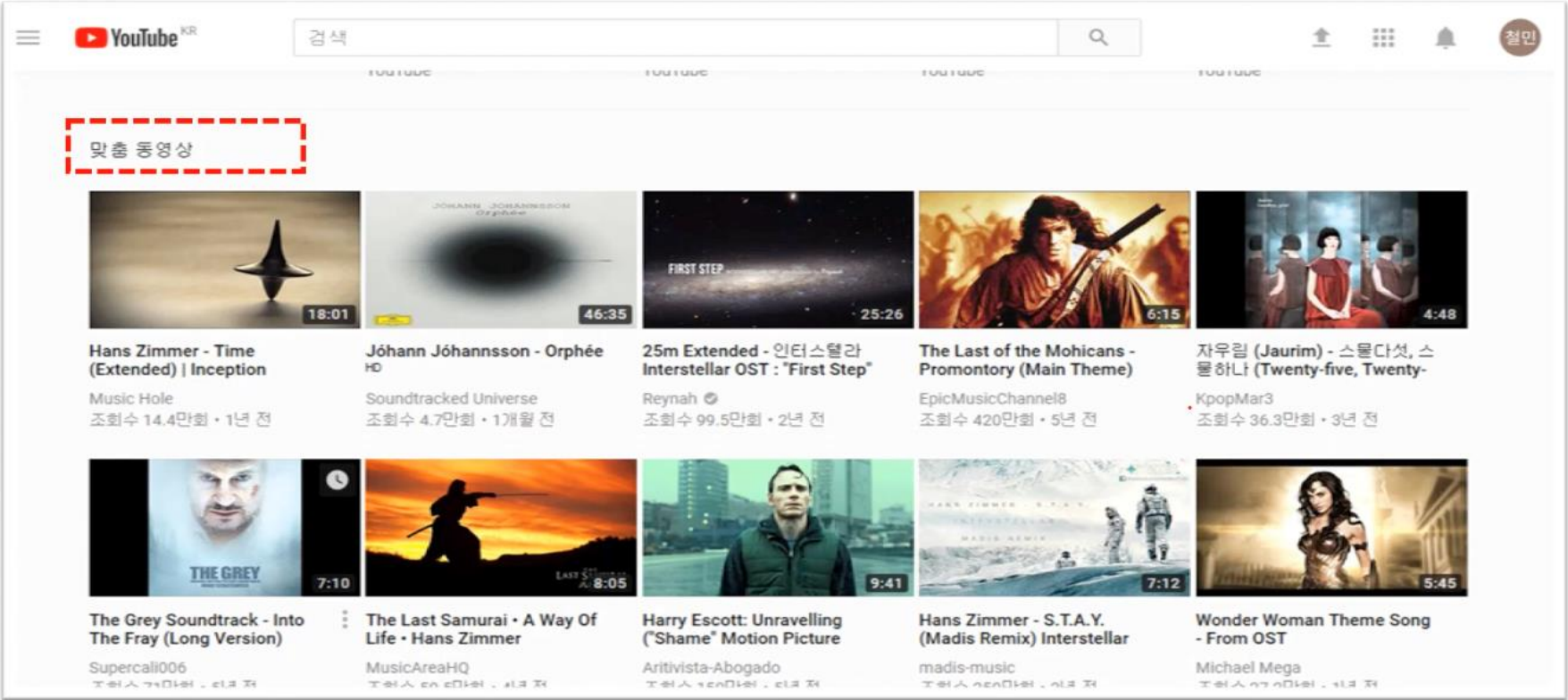
상위 20%가 80%의 가치를 창출한다 하위 80%가 상위 20%의 가치보다 크다.



인터넷의 발전 (아마존, 넷플릭스 등)

유튜브 : 사용자에게 맞춤 콘텐츠를 제공

인터스텔라 감상 후
한스짐머 감성 노래 추천



유튜브 : 사용자에게 맞춤 콘텐츠를 제공

당근마켓

당근마켓에서의 추천 사례


- 다른 사람들이 같이 본 상품 추천



이 상품과 함께 봤어요




아마존 : 리뷰/평점 기반의 추천시스템

amazon try Prime All  The Halloween Shop


Departments ▼ Browsing History ▼ chulmin's Amazon.com Today's Deals Gift Cards & Registry Sell Help EN Hello, chulmin Account & Lists Orders Try Prime 0 Cart

Your recently viewed items and featured recommendations


Best Sellers Page 1 of 7



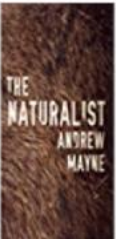
Wheat Belly: Lose the Wheat, Lose the...
William Davis MD
★★★★☆ 6,000
Kindle Edition
\$8.42




From Sand and Ash
Amy Harmon
★★★★☆ 2,948
Kindle Edition
\$4.99




A Dark Lure
Loreth Anne White
★★★★☆ 4,817
Kindle Edition
\$4.99



The Naturalist (The Naturalist Series Book...
Andrew Mayne
★★★★☆ 1,691
Kindle Edition
\$4.99



I Am Watching You
Teresa Driscoll
★★★★☆ 1,392
Kindle Edition
\$4.99



Gulp: Adventures on the Alimentary Canal
Mary Roach
★★★★☆ 1,072
Kindle Edition
\$9.99

추천엔진의 필요성

너무 많은 상품으로 가득찬 온라인 스토어



VS

추천엔진은 사용자가 무엇을 원하는지 빠르게 찾아내어 사용자의 온라인 쇼핑 이용 즐거움을 배가 시킨다.

한정된 시간, 어떤 상품을 골라야 할지 선택의 압박



추천시스템을 위한 다양한 데이터

- 사용자가 어떤 상품을 구매했는가?
- 사용자가 어떤 제품을 Browse 했는가?
- 사용자가 무엇을 클릭했는가?
- 사용자가 평가한 영화 평점은?



추천 엔진

“ 당신만을 위한 최신 상품 “

“ 이 상품을 선택한 다른
사람들이 좋아하는 상품들 “

“ 이 상품을 좋아하시나요? 아래
있는 다른 상품은 어떠신가요? “

추천시스템의 묘미 : 아하 모멘트



추천 시스템의 묘미는 사용자 자신도 좋아하는지 모르고 있었던 취향을 발견하는 것이다. 추천 시스템에 신뢰가 높아지면서 사용자는 추천 아이템을 더 많이 선택하게 되고, 이로 인해 더 많은 데이터가 추천 시스템에 축적되면서 추천이 정확해지고 다양해진다.

넷플릭스 : 자체 생산 콘텐츠 추천하는 경향

넷플릭스의 경우 자사가 생성한 콘텐츠
위주로 추천 영화가 치우치는 경향이 시작됨.



추천시스템 방식



콘텐츠 기반 필터링
Content Based Filtering

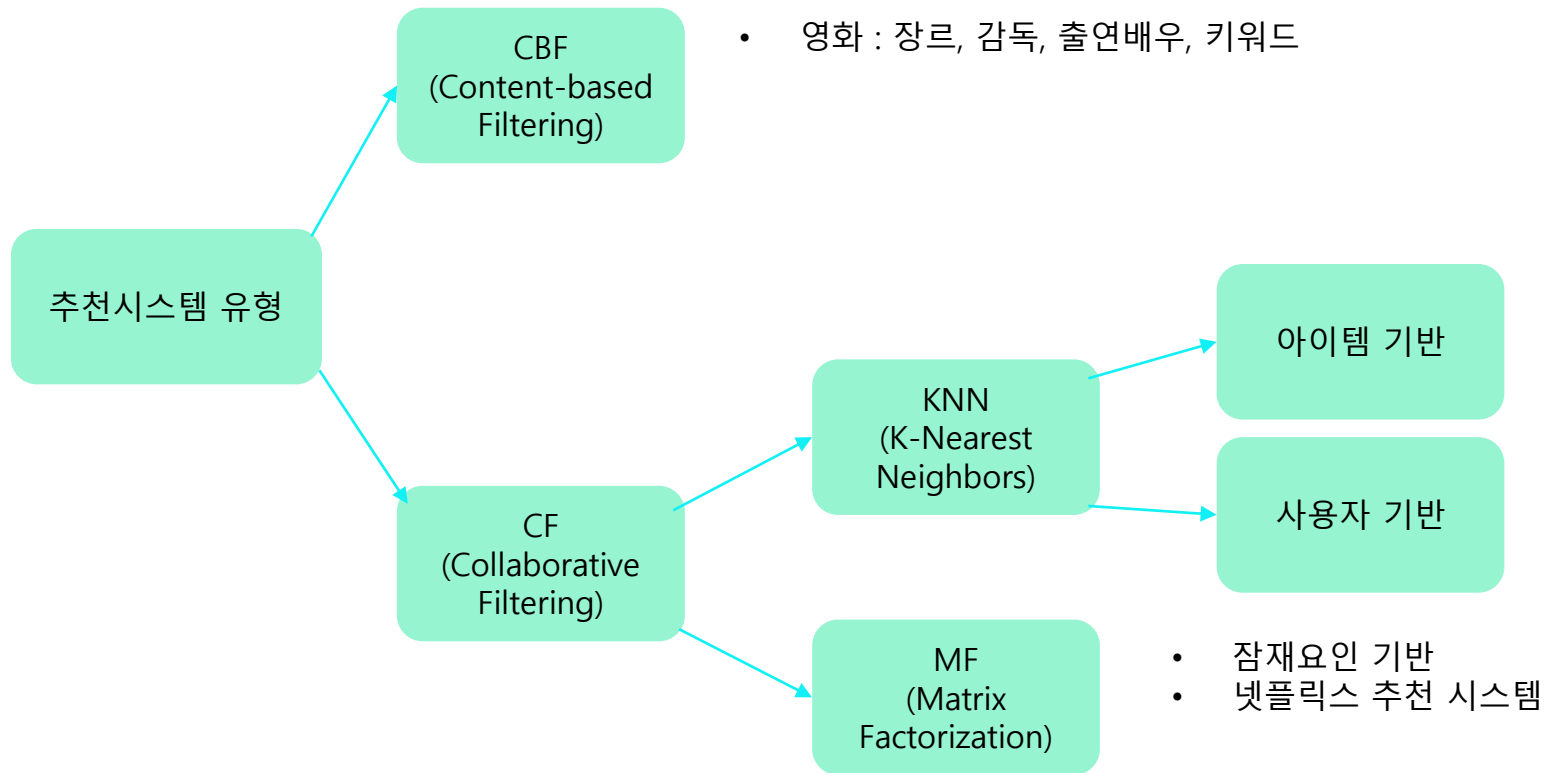
추천 시스템은 이들 방식중 1가지를
선택하거나 이들을 결합하여 hybrid
방식으로 사용

(예 : Content Based + Collaborative
Filtering)



협업 필터링
Collaborative Filtering

추천시스템 방식



콘텐츠 기반 필터링 추천시스템

장르 : SF, 드라마, 미스터리
감독 : 드니 빌뇌브
출연 : 에이미 아담스,
제레미 러너
키워드 : 외계인 침공,
예술성, 스릴러 요소



평점 : 8.0



장르 : SF, 액션, 스릴러
감독 : 리들리 스콧
출연 : 노미 라마스,
마이클 패스벤더
키워드 : 에일리언 프리퀼,
액션과 스릴러의 조화



평점 : 9.0

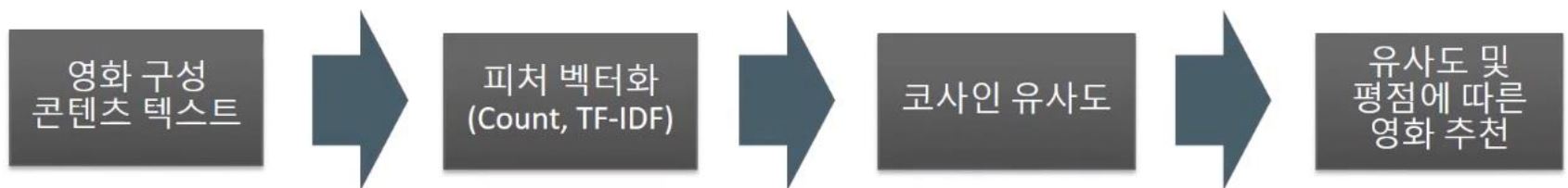
사용자 선호 프로필
선호 장르 : SF, 액션, 스릴러
선호 배우 : 에이미 아담스, 마이클
패스벤더 등
선호 감독 : 리들리 스콧, 드니 빌뇌브



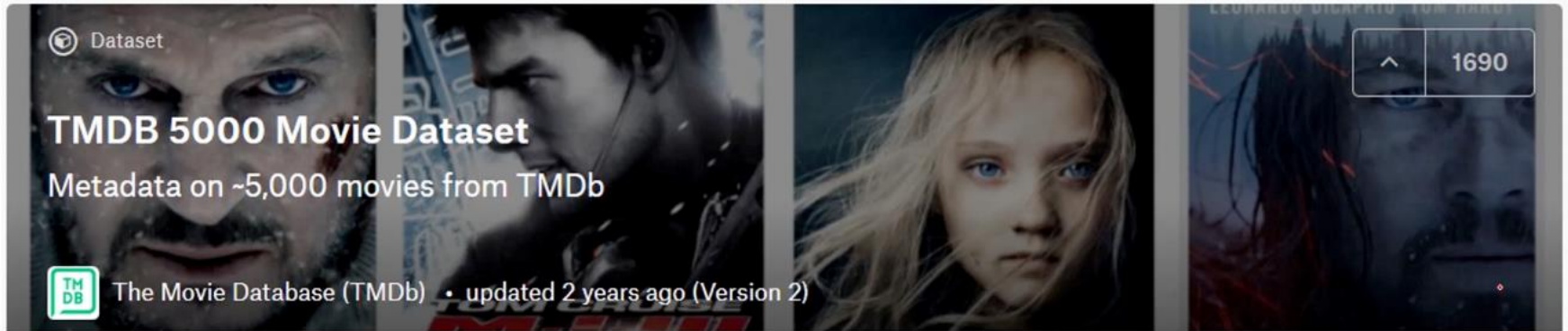
장르 : SF, 액션, 스릴러
감독 : 드니 빌뇌브
출연 : 라이언 고슬링,
해리슨 포드
키워드 : 리들리 스콧
감독의 전작을 리메이크

콘텐츠 기반 필터링 추천시스템

감독, 배우, 영화 설명, 장르등 영화를 구성하는
다양한 콘텐츠들을 텍스트 기반 문서 유사도로
비교하여 추천



CBF 실습 : 장르 유사도 기반 영화 추천시스템



kaggle tmdb dataset 5000



전체

이미지

뉴스

동영상

지도

더보기

설정

도구

검색결과 약 8,420개 (0.35초)

www.kaggle.com › tmdb › tmdb-movie-metadata ▼

TMDB 5000 Movie Dataset | Kaggle

2017. 9. 28. — **TMDB 5000 Movie Dataset**. Metadata on ~5,000 movies from TMDb. The Movie Database (TMDb). • updated 3 years ago (Version 2).

Metadata on

Metadata on ~5000 movies from

TMDB 5000 Movie Dataset

TMDB 5000 Movie Dataset.

CBF 실습 : TMDB 5000 데이터

콘텐츠 기반 필터링 구현 프로세스

1. 콘텐츠에 대한 여러 텍스트 정보들을 피처 벡터화
2. 코사인 유사도로 콘텐츠별 유사도 계산
3. 콘텐츠 별로 가중 평점을 계산
4. 유사도가 높은 콘텐츠 중에 평점이 좋은 콘텐츠 순으로 추천

실습 : 데이터 읽어오기

예제 소스 : 9.5 콘텐츠 기반 필터링 실습 _ TMDb 5000 Movie Dataset.ipynb

```
import pandas as pd
import numpy as np
import warnings; warnings.filterwarnings('ignore')

movies = pd.read_csv('tmdb_5000_movies.csv')

print(movies.shape)
movies.head(3)
```

(4803, 20)

장르 유사도 기반 영화 추천시스템

	budget	genres	homepage	id	keywords	original_language	original_title	overview	popularity	production_co
0	237000000	[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"}]	http://www.avatarmovie.com/	19995	[{"id": 1463, "name": "culture clash"}, {"id": 726, "name": "ocean"}]	en	Avatar	In the 22nd century, a paraplegic Marine is di...	150.437577	[{"name": "I", "id": 19995, "type": "Production Company"}]
1	300000000	[{"id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"}]	http://disney.go.com/disneypictures/pirates/	285	[{"id": 270, "name": "ocean"}, {"id": 726, "name": "ocean"}]	en	Pirates of the Caribbean: At World's End	Captain Barbosa, long believed to be dead, ha...	139.082615	[{"name": "W", "id": 285, "type": "Production Company"}]
2	245000000	[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"}]	http://www.sonypictures.com/movies/spectre/	206647	[{"id": 470, "name": "spy"}, {"id": 818, "name": "spy"}]	en	Spectre	A cryptic message from Bond's past sends him o...	107.376788	[{"name": "I", "id": 206647, "type": "Production Company"}]

budget	genres	homepage	id	keywords	original_lang	original_title	overview	popularity	production_co	production_co	release_date	revenue	runtime	spoken_langu	status	tagline	title	vote_average	vote_count
237000000	[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"}]	http://www.avatarmovie.com/	19995	[{"id": 1463, "name": "culture clash"}, {"id": 726, "name": "ocean"}]	en	Avatar	In the 22nd century, a paraplegic Marine is di...	150.437577	[{"name": "I", "id": 19995, "type": "Production Company"}]	[{"name": "I", "id": 19995, "type": "Production Company"}]	2009.12.10	2787965087	162	[{"iso_639_1": "en", "name": "English"}]	Released	Enter the World of Avatar	Avatar	7.2	11800
300000000	[{"id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"}]	http://disney.go.com/disneypictures/pirates/	285	[{"id": 270, "name": "ocean"}, {"id": 726, "name": "ocean"}]	en	Pirates of the Caribbean: At World's End	Captain Barbosa, long believed to be dead, ha...	139.082615	[{"name": "W", "id": 285, "type": "Production Company"}]	[{"name": "W", "id": 285, "type": "Production Company"}]	2007.5.19	961000000	169	[{"iso_639_1": "en", "name": "English"}]	Released	At the end of Pirates of the Caribbean	Pirates of the Caribbean: At World's End	6.9	4500
245000000	[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"}]	http://www.sonypictures.com/movies/spectre/	206647	[{"id": 470, "name": "spy"}, {"id": 818, "name": "spy"}]	en	Spectre	A cryptic message from Bond's past sends him o...	107.376788	[{"name": "C", "id": 206647, "type": "Production Company"}]	[{"name": "C", "id": 206647, "type": "Production Company"}]	2015.10.26	880674609	148	[{"iso_639_1": "en", "name": "English"}]	Released	A Plan No Or Spectre	Spectre	6.3	4466
250000000	[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"}]	http://www.sony.com/electronics/movies/the-dark-knight-rises	49026	[{"id": 849, "name": "action"}, {"id": 851, "name": "action"}]	en	The Dark Knight Rises	Following the events of The Dark Knight and The Dark Knight Returns, Batman must once again save Gotham City from a powerful enemy.	112.31295	[{"name": "L", "id": 49026, "type": "Production Company"}]	[{"name": "L", "id": 49026, "type": "Production Company"}]	2012.7.16	1084939099	165	[{"iso_639_1": "en", "name": "English"}]	Released	The Legend of The Dark Knight	The Dark Knight Rises	7.6	9106
260000000	[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"}]	http://www.sonypictures.com/movies/spider-man-3	49529	[{"id": 818, "name": "action"}, {"id": 851, "name": "action"}]	en	Spider-Man 3	When Peter Parker returns home, he finds out that his friend, Flash Thompson, is actually the villain, Venom.	43.926995	[{"name": "W", "id": 49529, "type": "Production Company"}]	[{"name": "W", "id": 49529, "type": "Production Company"}]	2012.3.7	284139100	132	[{"iso_639_1": "en", "name": "English"}]	Released	Lost in our world	Spider-Man 3	6.1	2124
258000000	[{"id": 14, "name": "Fantasy"}, {"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}]	http://www.sony.com/electronics/movies/spider-man-3	559	[{"id": 851, "name": "action"}, {"id": 851, "name": "action"}]	en	Spider-Man 3	When Peter Parker returns home, he finds out that his friend, Flash Thompson, is actually the villain, Venom.	115.699814	[{"name": "C", "id": 559, "type": "Production Company"}]	[{"name": "C", "id": 559, "type": "Production Company"}]	2007.5.1	890871626	139	[{"iso_639_1": "en", "name": "English"}]	Released	The battle with Spider-Man 3	Spider-Man 3	5.9	3576
260000000	[{"id": 16, "name": "Animation"}, {"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}]	http://disney.go.com/disneypictures/tangled	38757	[{"id": 1562, "name": "action"}, {"id": 1562, "name": "action"}]	en	Tangled	When the kingdom of Corona is threatened, a young woman with long, golden braided hair must embark on a journey of self-discovery.	48.681969	[{"name": "W", "id": 38757, "type": "Production Company"}]	[{"name": "W", "id": 38757, "type": "Production Company"}]	2010.11.24	591794936	100	[{"iso_639_1": "en", "name": "English"}]	Released	They're taking Tangled	Tangled	7.4	3330
280000000	[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"}]	http://marvel.com/movies/avengers-ag	99861	[{"id": 8828, "name": "action"}, {"id": 8828, "name": "action"}]	en	Avengers: Age of Ultron	When Tony Stark and his fellow Avengers discover that a powerful, ancient force is awakening, they must band together to stop it.	134.279229	[{"name": "M", "id": 99861, "type": "Production Company"}]	[{"name": "M", "id": 99861, "type": "Production Company"}]	2015.4.22	1405403694	141	[{"iso_639_1": "en", "name": "English"}]	Released	A New Age of Avengers	Avengers: Age of Ultron	7.3	6767
250000000	[{"id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"}]	http://harrypotter.com	767	[{"id": 616, "name": "action"}, {"id": 616, "name": "action"}]	en	Harry Potter and the Deathly Hallows - Part 2	As Harry Potter and his friends prepare to enter their final year at Hogwarts, they discover a dark secret about the wizard Voldemort.	98.885637	[{"name": "W", "id": 767, "type": "Production Company"}]	[{"name": "W", "id": 767, "type": "Production Company"}]	2009.7.7	933959197	153	[{"iso_639_1": "en", "name": "English"}]	Released	Dark Secrets	Harry Potter and the Deathly Hallows - Part 2	7.4	5293

장르 데이터 전처리

```
# 장르 데이터 전처리
```

```
from ast import literal_eval
```

```
movies_df['genres'] = movies_df['genres'].apply(literal_eval)
```

```
movies_df['keywords'] = movies_df['keywords'].apply(literal_eval)
```

```
movies_df['genres'] = movies_df['genres'].apply(lambda x : [ y['name'] for y in x])
```

```
movies_df['keywords'] = movies_df['keywords'].apply(lambda x : [ y['name'] for y in x])
```

```
movies_df[['genres', 'keywords']][:1]
```

	genres	keywords
0	[Action, Adventure, Fantasy, Science Fiction]	[culture clash, future, space war, space colony, society, space travel, futuristic, romance, spa...

우리에게 필요한 장르명만 뽑아온다

장르 피쳐 벡터화 – 영화간 유사도 계산

장르 문자열을 Count 기반 피쳐 벡터화 후에 코사인 유사도로 각 영화를 비교

```
from sklearn.feature_extraction.text import CountVectorizer

# CountVectorizer를 적용하기 위해 공백문자로 word 단위가 구분되는 문자열로 변환.
movies_df['genres_literal'] = movies_df['genres'].apply(lambda x : (' ').join(x))
count_vect = CountVectorizer(min_df=0, ngram_range=(1,2))
genre_mat = count_vect.fit_transform(movies_df['genres_literal'])
print(genre_mat.shape)
```

(4803, 276) bigram으로 피쳐 수 276개로 증가

```
from sklearn.metrics.pairwise import cosine_similarity

genre_sim = cosine_similarity(genre_mat, genre_mat)
print(genre_sim.shape)
print(genre_sim[:2])
```

(4803, 4803) 영화 간 장르 유사도를 코사인 유사도로 계산

```
[[1.          0.59628479 0.4472136  ... 0.          0.          0.          ]
 [0.59628479 1.          0.4       ... 0.          0.          0.          ]]
```

```
genre_sim_sorted_ind = genre_sim.argsort()[::-1]
print(genre_sim_sorted_ind[:1])
```

[[0 3494 813 ... 3038 3037 2401]]

첫번째 영화와 유사도가 높은 영화 순서

특정 영화와 장르 유사도가 높은 영화 추천하기

```
# 특정 영화와 장르 유사도가 높은 영화를 반환하는 함수
def find_sim_movie(df, sorted_ind, title_name, top_n=10):

    # 인자로 입력된 movies_df DataFrame에서 'title' 컬럼이 입력된 title_name 값인 DataFrame추출
    title_movie = df[df['title'] == title_name]

    # title_named을 가진 DataFrame의 index 객체를 ndarray로 반환하고
    # sorted_ind 인자로 입력된 genre_sim_sorted_ind 객체에서 유사도 순으로 top_n 개의 index 추출
    title_index = title_movie.index.values
    similar_indexes = sorted_ind[title_index, :(top_n)]

    # 추출된 top_n index들 출력. top_n index는 2차원 데이터 임.
    #dataframe에서 index로 사용하기 위해서 1차원 array로 변경
    print(similar_indexes)
    similar_indexes = similar_indexes.reshape(-1)

    return df.iloc[similar_indexes]

similar_movies = find_sim_movie(movies_df, genre_sim_sorted_ind, 'The Godfather',10)
similar_movies[['title', 'vote_average']]
```

`[[2731 1243 3636 1946 2640 4065 1847 4217 883 3866]]`

영화 갓파더와 장르 유사도가 높은 순서

	title	vote_average
2731	The Godfather: Part II	8.3
1243	Mean Streets	7.2
3636	Light Sleeper	5.7
1946	The Bad Lieutenant: Port of Call - New Orleans	6.0
2640	Things to Do in Denver When You're Dead	6.7
4065	Mi America	0.0
1847	GoodFellas	8.2
4217	Kids	6.8
883	Catch Me If You Can	7.7
3866	City of God	8.1

문제
평가횟수가 현저히 적은 영화들이 추천되는 것도 있음
low quality 추천 문제

우리가 전혀 모르는 영화를 추천받는 것은 엉뚱한 추천 결과를 낼 수 있음

-> 평가횟수를 반영한 추천 시스템이 필요

가중평점 반영한 영화 추천

가중평점(평점&평가횟수) 반영한 영화 추천

@ 가중평점(Weighted Rating):

$$(v/(v+m))*R + (m/(v+m))*C$$

- v : 영화별 평점을 투표한 횟수(vote_count) ★ 투표횟수가 많은 영화에 가중치 부여
- m : 평점을 부여하기 위한 최소 투표 횟수 -> 여기서는 투표수 상위 60%
- R : 개별 영화에 대한 평균 평점(vote_average)
- C : 전체 영화에 대한 평균 평점(movies_df['vote_average'].mean())

C, m은 고정값
v,R은 영화마다 변동값

투표 횟수가 많으면
가중치가 붙는다.

최종적으로

1. 장르가 유사한 영화 중
2. 가중평점이 높은 영화가 추천되게 된다.

```
In [86]: # 상위 60%에 해당하는 vote_count를 최소 투표 횟수인 m으로 지정
C = movies_df['vote_average'].mean()
m = movies_df['vote_count'].quantile(0.6)
```

```
In [84]: # C: 전체 영화에 대한 평균평점 = 약 6점
# m: 평점을 부여하기 위한 최소 투표 횟수 = 370회(상위 60% 수준)
print('C:',round(C,3), 'm:',round(m,3))
```

C: 6.092 m: 370.2

가중평균점 계산하는 함수

가중평균점을 계산하는 함수

```
In [85]: ▶ def weighted_vote_average(record):  
    v = record['vote_count']  
    R = record['vote_average']  
  
    return ( (v/(v+m)) * R ) + ( (m/(m+v)) * C )
```

가중평균점을 return값으로 돌려준다.

```
In [87]: ▶ # 기존 데이터에 가중평균점 칼럼 추가  
movies_df['weighted_vote'] = movies_df.apply(weighted_vote_average, axis=1)
```

장르 유사도 기반 영화 추천시스템

추천 ver2. 먼저 장르 유사성 높은 영화 20개 선정 후, 가중평점순 10개 선정

```
In [95]: ▶ def find_sim_movie_ver2(df, sorted_ind, title_name, top_n=10):  
    title_movie = df[df['title'] == title_name]  
    title_index = title_movie.index.values  
  
    # top_n의 2배에 해당하는 장르 유사성이 높은 index 추출  
    similar_indexes = sorted_ind[title_index, :(top_n*2)]  
    similar_indexes = similar_indexes.reshape(-1)  
  
    # 기준 영화 index는 제외  
    similar_indexes = similar_indexes[similar_indexes != title_index]  
  
    # top_n의 2배에 해당하는 후보군에서 weighted_vote 높은 순으로 top_n 만큼 추출  
    return df.iloc[similar_indexes].sort_values('weighted_vote', ascending=False)[:top_n]
```

먼저, 장르 유사성 높은 것 20개를 먼저 뽑은 뒤

최종적으로 상위 10개를
sort해서 보여준다

장르 유사도 기반 영화 추천시스템

영화 Godfather에 대해 장르 유사성, 가중평점 반영한 추천 영화 10개를 뽑아보자

```
In [96]: > similar_movies = find_sim_movie_ver2(movies_df, genre_sim_sorted_ind, 'The Godfather', 10)
similar_movies[['title', 'vote_average', 'weighted_vote', 'genres', 'vote_count']]
```

Out[96]:

	title	vote_average	weighted_vote	genres	vote_count
2731	The Godfather: Part II	8.3	8.079586	[Drama, Crime]	3338
1847	GoodFellas	8.2	7.976937	[Drama, Crime]	3128
3866	City of God	8.1	7.759693	[Drama, Crime]	1814
1663	Once Upon a Time in America	8.2	7.657811	[Drama, Crime]	1069
883	Catch Me If You Can	7.7	7.557097	[Drama, Crime]	3795
281	American Gangster	7.4	7.141396	[Drama, Crime]	1502
4041	This Is England	7.4	6.739664	[Drama, Crime]	363
1149	American Hustle	6.8	6.717525	[Drama, Crime]	2807
1243	Mean Streets	7.2	6.626569	[Drama, Crime]	345
2839	Rounders	6.9	6.530427	[Drama, Crime]	439

평가 횟수가 반영된
고품질의 추천이 적용된 모습

장르 유사도 기반 영화 추천시스템

요약 : Godfather를 좋아하는 사람에게 영화 추천해주기

Godfather장르가 Drama, Crime이다.

우선 Drama, Crime 장르 기준으로 상위 20개 영화를 뽑아보고,
그 중 평가횟수를 반영한 가중평점 기준 상위 10개 영화를 뽑아서 추천해준다.

장르 유사도 기반 영화 추천시스템

응용 : Spider-Man 3 좋아하는 사람 기준으로 장르가 유사한 영화를 추천해주자

```
In [30]: ► similar_movies = find_sim_movie_ver2(movies_df, genre_sim_sorted_ind, 'Spider-Man 3', 10)
similar_movies[['title', 'vote_average', 'weighted_vote', 'genres', 'vote_count']]
```

Out[30]:

	title	vote_average	weighted_vote	genres	vote_count
329	The Lord of the Rings: The Return of the King	8.1	8.011871	[Adventure, Fantasy, Action]	8064
262	The Lord of the Rings: The Fellowship of the Ring	8.0	7.922175	[Adventure, Fantasy, Action]	8705
330	The Lord of the Rings: The Two Towers	8.0	7.910111	[Adventure, Fantasy, Action]	7487
19	The Hobbit: The Battle of the Five Armies	7.1	7.027274	[Action, Adventure, Fantasy]	4760
98	The Hobbit: An Unexpected Journey	7.0	6.961224	[Adventure, Fantasy, Action]	8297
126	Thor: The Dark World	6.8	6.748873	[Action, Adventure, Fantasy]	4755
30	Spider-Man 2	6.7	6.652034	[Action, Adventure, Fantasy]	4321
129	Thor	6.6	6.572735	[Adventure, Fantasy, Action]	6525
20	The Amazing Spider-Man	6.5	6.478296	[Action, Adventure, Fantasy]	6586
38	The Amazing Spider-Man 2	6.5	6.466812	[Action, Adventure, Fantasy]	4179

Other Methods

- 최근접 이웃 협업 필터링
- 잠재 요인 협업 필터링



〈 백만 달러의 상금이 걸린 넷플릭스 추천 엔진 경연 대회 우승팀 사진 〉



Thank You