영화 메타데이터의 증가에 따른 콘텐츠 기반 추천 시스템 성능 향상

서진경[O], 최다정[*], 백주련[*]
[O]평택대학교 데이터정보학과,
[*]평택대학교 데이터정보학과
e-mail: sjk2700@naver.com[O], {dajung2020, jrpaik}@ptu.ac.kr[*]

Performance Improvement of a Contents-based Recommendation System by Increasing Movie Metadata

Jin-kyeong Seo[O], Da-jeong Choi[*], Juryon Paik[*]
[O]Dept. of Digital Information & Statistics, Pyeongtaek University,
[*]Dept. of Digital Information & Statistics, Pyeongtaek University

● 요 약 ●

OTT 서비스의 이용자가 폭발적으로 증가하고 있는 지금, 사용자에게 맞춤형 상품을 추천하는 것은 해당 서비스에서 중요한 사안이다. 본 논문에서는 콘텐츠 기반 추천 시스템의 모델을 제안하고, 영화 데이터를 추가해가며 예측력을 높일 최종적인 모델을 채택하고자 한다. 이를 위해 GroupLens와 Kaggle에서 영화 데이터를 수집하고 총 1111개의 영화, 943명의 사용자에게서 나온 71026개의 영화 평가 데이터를 이용하였다. 모델 평가 결과, 장르와 키워드만을 이용한 추천 시스템 모델의 RMSE는 1.3076, 단계적으로 데이터를 추가해 최종적으로 장르, 키워드, 배우, 감독, 나라, 제작사를 이용한 추천 시스템 모델의 RMSE는 1.1870으로 모든 데이터를 추가한 모델의 예측력이 더 높았다. 이에 따라 장르, 키워드, 배우, 감독, 나라, 제작사를 이용해 구현한 모델을 최종적인 모델로 채택, 무작위로 추출한 한 명의 사용자에 대한 영화 추천 리스트를 뽑아낸다.

키워드: 콘텐츠 기반 추천 시스템(contents-based recommendation system), 메타데이터(metadata), 성능 향상(performance improvement)

I. Introduction

원하는 작품을 보기 위해 정해진 시간에 맞춰 시청해야 하는 TV보다, 개인이 원하는 시간에 원하는 작품을 마음대로 골라 볼 수 있는 OTT 서비스의 장점 덕분인지 최근 몇 년간 OTT 서비스의 이용자 수가 급격히 증가하고 있다. 이러한 OTT 서비스에서 빠질 수 없는 기술은 추천 시스템으로, 실제로 유튜브의 최고 상품 담당자인 닐 모한은 2019년 3월 뉴욕타임즈와의 인터뷰에서 "유튜브 이용자들의 시청 시간 70%가 추천 알고리즘에 의한 결과이며, 알고리즘의 도입으로 총 비디오 시청 시간이 20배 이상 증가했다"라고 밝혔으며 이러한 시청 시간의 증가는 매출로 이어진다. 따라서 이용자들의 취향을 고려해 더 정확한 추천을 해야한다는 것이 OTT 서비스를 제공하는 기업들의 중요한 사안으로 떠올랐다. 본 논문에서는 이러한 추천 시스템의 알고리즘 중 콘텐츠 기반 모델을 이용, 더 정확한 추천을 위한 모델 성능 향상의 방법으로 메타데이터의 다양성을 이용한다. 추천 모델에 메타데이터를 추가해가며 더 예측력이 높은 모델을 수립하고자 하며 이를 통해 메타데이터의 중요성을 알아보고자 한다. 최종적으로 가장 예측력이 좋은 모델을 수립되면 특정 사용자에 대한 추천 영화 리스트를 제공함으로 결과를 보여준다.

II. The Proposed Scheme

1. 데이터

1.1 데이터 수집 및 정제

본 논문에서는 Grouplens의 MovieLens 100K 데이터 중 u.item, u.data와 Kaggle의 The Movies Dataset 중 movie_matadata, credits, keywords 데이터를 이용한다. 이는 각 영화에 대한 전체 투표수와 평균 평점만 존재하는 The Movies Dataset 데이터의 문제점을 보완하기 위해 MovieLens의 사용자별 평점 데이터를 사용하기 위함이다. 두 데이터는 영화의 제목을 기준으로 병합되었으며, 시스템 구현에 필요한 변수만을 뽑아 새로운 하나의 데이터로 재구성하였다. 모든 데이터는 영어로 표현되어 있으며 아래의 Table 1은 최종 완성된 데이터에 대한 설명을 담고 있다.

Table 1. 변수 설명

변수명	설명
title	영화의 제목
user_id	사용자의 고유 ID
movie_id	영화의 고유 ID
rating	특정 사용자가 특정 영화에 준 평점
genres	영화의 장르
cast	영화에 출연한 배우 (역할, 이름, 성별, 우선순위 등)
crew	영화를 만든 사람들의 정보 (부서, 직업, 성별 등의 변수 포함)
keywords	영화의 핵심 키워드 정보

1.2 데이터 탐색

중복 값을 제거한 전체 영화의 수는 1111개이며, 총 943명의 사용자가 존재한다. 영화별 각 사용자 평점이 합해진 데이터는 총 71026개의 행이 존재하며, 이에 따른 평가 개수에 대한 정보는 다음과 같다.

Table 2. 평가 개수의 요약 통계량

	votes
min	1.000000
25%	8.000000
50%	32.000000
75%	86.000000
max	583.000000

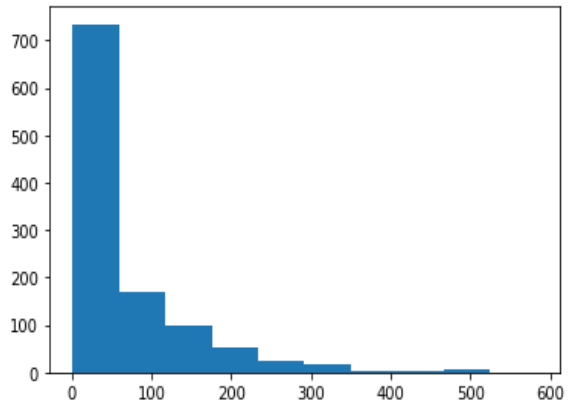

Fig. 1. 평가 개수 히스토그램

한 영화에 대해 최소 1개에서 최대 583개의 평가 개수가 존재하는 것을 볼 수 있으며, 대부분 영화에서는 0~100개의 평가 개수를 받은 것으로 나타난다.

이상값의 영향을 피해 중앙값을 봤을 때, 한 영화에 대해 평균 32개의 평가를 받은 것으로 볼 수 있다.

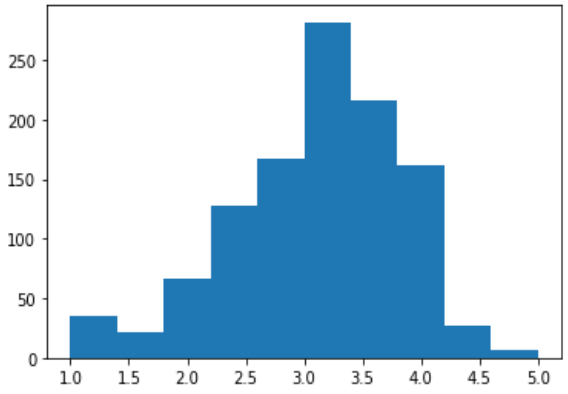

Fig. 2. 평점 히스토그램

각각에 대한 평가 점수를 담고 있는 rating 변수는 0점부터 5점까지의 범위를 가진다. 그 중 3.0에서 3.5점을 기준으로 데이터가 대칭적인 모습을 보인다.

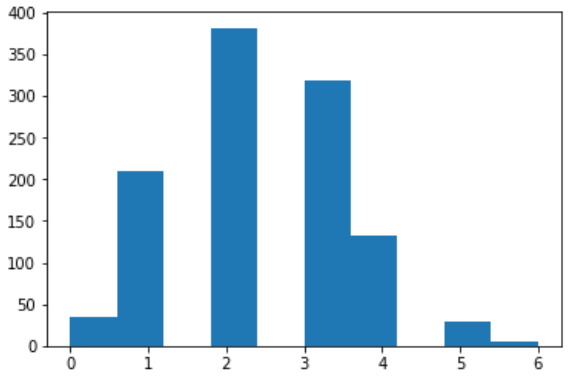

Fig. 3. 장르 수 히스토그램

영화의 장르는 총 20개로 다음과 같다.

[Action(액션), Adventure(모험), Animation(만화영화), Comedy(개그), Crime(범죄), Documentary(다큐멘터리), Drama (드라마), Family(가족), Fantasy(판타지), Foreign(외국), History (역사), Horror(공포), Music(음악), Mystery(미스터리), Romance (로맨스), Science Fiction(공상 과학), TV Movie(TV 영화), Thriller (스릴러), War(전쟁), Western(서부극)]

각 영화는 최소 0개부터 최대 6개까지의 장르를 가지며, 양극단을 제외한 중앙에 데이터가 몰려있는 모습을 나타낸다.

1.3 데이터 처리

영화 간 유사도 지표인 코사인 유사도를 구하기 위해선 카운트 벡터 행렬이 필요하다. 이 카운트 벡터 행렬 안에는 영화와 관련된 정보가 들어가며 이를 위한 데이터 정제가 필요하다. 키워드나 장르, 배우 및 제작사는 한 변수에 여러 개의 정보가 들어간 변수이다. 이러한 변수들의 개수에 대해선 최대 세 개를 제한으로 두어 데이터를 잘라주었다. 이때 배우 변수에 대해서는 캐스팅 우선순위로 내림차순 정렬하여 잘라주었다. 감독에 대한 정보는 crew 변수가 가지고 있으며

이를 추출하기 위해 job이 director인 crew의 이름을 받아 director라는 새로운 변수를 구성하였다. 이렇게 정리된 변수들은 후에 각각 모델에 맞게 합쳐져 soup 변수를 이룬다. 이 soup 변수가 바로 위에서 설명한 카운트 벡터 행렬과 코사인 유사도를 구할 때 사용할 변수이다. Table 3은 최종 정리된 변수에 대한 설명이다.

Table 3. 최종 데이터 변수 설명

변수명	설명
title	영화의 제목
user_id	사용자의 고유 ID
movie_id	영화의 고유 ID
rating	특정 사용자가 특정 영화에 준 평점
genres	영화의 장르 -최대 세 개까지 존재
cast	영화에 출연한 배우 -우선순위에 따라 세 명까지 존재
crew	영화를 만든 사람들의 정보 (부서, 직업, 성별 등의 변수 포함)
keywords	영화의 핵심 키워드 정보 -최대 세 개까지 존재
director	영화의 감독 이름
soup	카운트 벡터 행렬에 사용할 변수 (모델별 사용하는 변수가 다름)

최종적으로 모델을 학습하고 평가하기 위해 전체 데이터를 훈련 데이터(75%)와 시험 데이터(25%)로 분리하였으며, 그 결과 훈련 데이터 56820개와 시험 데이터 14206개로 나누어졌다.

2. 모델 수립, 예측 및 평가

2.1 모델링

본 논문에서의 추천 시스템은 콘텐츠 기반의 모델을 이용했다. 콘텐츠 기반 추천 시스템이란 사용자가 이전에 선택한 상품 중 좋은 평가를 한 상품과 유사도가 높은 상품을 추천하는 시스템이다. 해당 모델에서는 soup 변수에 포함되는 메타데이터의 종류가 다양해질수록 모델의 정확도가 향상될 것이라고 가정한다. 이에 따라 단계별로 메타데이터를 추가해가며 모델의 정확도를 측정했다. 전체적인 알고리즘은 다음과 같으며, 모든 영화에 대한 카운트 벡터 행렬과, 그에 따른 코사인 유사도를 미리 구해 count_sim이라는 새로운 변수에 저장해 놓았다.

1. 사용자의 아이디를 입력받는다.
2. 해당 사용자가 평가한 영화 중 가장 높은 평점을 준 영화의 제목을 찾는다. 이때, 최고점을 준 영화가 여러 개라면 가장 처음 나온 영화를 이용한다.
3. 이 영화의 인덱스를 가지고 count_sim 변수에서 해당하는 행을 뽑는다.
4. 코사인 유사도를 내림차순으로 정렬하여 유사도가 높은 상위 열 개의 영화를 사용자에게 추천한다.

2.2 모델 평가

해당 모델의 평가는 평점을 이용해 진행한다. 각 사용자에게 추천된 상위 열 개의 영화 리스트를 이용해 평점을 예측하고, 이를 실제 평점과 비교하여 예측력이 높을수록 사용자에게 잘 맞는 영화 리스트를 추천했다고 평가한다. 학습 데이터를 이용하여 모델을 학습하고, 평가 데이터를 이용해 예측에 관한 결과를 나타내었다. 평가 지표는 RMSE(Root Mean Squared Error)를 사용하였으며 이는 평균 제곱근 오차이다. RMSE가 낮을수록 예측력이 좋은 모델이라고 평가하였으며, 모델별 RMSE를 구한 결과는 다음과 같다.

-키워드, 장르를 이용한 모델

```
evaluatedValue
1.3075521134134243
```

-키워드, 장르, 배우, 감독을 이용한 모델

```
evaluatedValue
1.2100318041248186
```

-키워드, 장르, 배우, 감독, 나라를 이용한 모델

```
evaluatedValue
1.2039543765013112
```

-키워드, 장르, 배우, 감독, 제작사를 이용한 모델

```
evaluatedValue
1.197896284231859
```

-키워드, 장르, 배우, 감독, 나라, 제작사 이용한 모델

```
evaluatedValue
1.1869795414975377
```

soup 변수에 포함된 데이터의 수가 늘어날수록 RMSE가 적어지며, 이는 곧 모델의 예측력이 높아짐을 의미한다. 따라서 키워드, 장르, 배우, 감독, 나라, 제작사 데이터를 모두 포함한 모델을 최종 모델로 채택한다.

2.3 최종 모델을 이용한 영화 추천 시스템

수립된 모델을 가지고 무작위로 뽑은 한 명의 사용자에 대한 추천 영화 리스트를 뽑아보고자 한다.

```
indices=pd.Series(df.index, index=df['title']).drop_duplicates()

def content_recommender(user_id,count_sim,dataset,indices):
    m_list=dataset[dataset['user_id']==user_id]
    r_max=m_list['rating'].max()
    title=m_list[m_list['rating']==r_max].iloc[0]['title']
    idx=indices[title]

    sim_scores=count_sim[idx]
    sim_scores=enumerate(sim_scores)
    sim_scores=list(sim_scores)
    sim_scores=sorted(sim_scores,key=lambda x:x[1],reverse=True)
    sim_scores=sim_scores[1:11]
    movies_indices=[i[0] for i in sim_scores]
    return df['title'].iloc[movies_indices]

print("Input Your id:")
user_id=int(input())
p_list=content_recommender(user_id,count_sim,df2,indices)
p_list
```

Fig. 4. 모델링 구현한 소스 코드

위의 코드는 모델링 단계를 실제로 구현한 것으로, 직접 사용자의 아이디를 입력받았다. 하지만 지금은 무작위로 뽑은 한 명의 사용자에

대한 추천이 필요하므로 아래와 같이 사용자의 아이디를 입력받는 부분의 소스 코드를 변경하였다.

```
user_list=total_data['user_id'].drop_duplicates()
user_id=random.choice(user_list)
```

Fig. 5. 아이디 입력 부분에 대해 변경된 소스 코드

무작위 추출의 결과로 303번의 사용자가 뽑혔으며, 그 결과 다음 열 개의 영화들이 추천되었다.

```
In [43]: p_list
Out[43]:
403                          Aladdin
473          Oliver & Company
283      Addams Family Values
911           Cats Don't Dance
662        That Thing You Do!
177                  Houseguest
223                  Rent-a-Kid
320     Getting Even with Dad
520          Larger Than Life
651       So Dear to My Heart
```

Fig. 6. 303번 사용자를 위한 영화 추천 리스트

303번 사용자가 평가한 최고 평점은 5점이었으며, 이에 해당하는 영화 Toy Story가 기준으로 사용되었다.

III. Conclusions

본 연구에서는 모델 성능 평가를 통해 영화와 관련된 데이터를 추가할수록 모델의 성능이 높아지는 것을 확인하였으며 특정 사용자에 대한 추천도 진행했다. 이를 통해 콘텐츠 기반 추천 시스템에 메타데이터가 미치는 영향과 그 중요성에 대해 생각해볼 수 있는 연구가 되었다. 이에 대한 연장선으로 추후 연구에서는 이 메타데이터의 영향력이 어느 정도인지, 얼마나 중요한지에 대한 연구를 위해 딥러닝을 적용한 메타데이터 추출 연구를 수행하고자 한다.

ACKNOWLEDGEMENT

이 논문은 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. NRF-2021R1F1A1064073).

REFERENCES

[1] Kaggle,https://www.kaggle.com/rounakbanik/the-movies-dataset
[2] MovieLens, https://grouplens.org/datasets/movielens/
[3] http://ibomalkoc.com/movies-dataset/