1. 요약

빅데이터 시대 도래에 따라 소비자에게 맞춤화된 서비스에 대한 관심이 크게 상승했다. 추천 시스템(Recommendation System)으로 다양한 소스들이 공개되고 있다. 이 중 활발하게 사용되고 있는 Surprise 와 Python-recsys 를 비교 및 분석하여, 어떠한 알고리즘이 가장 정확하게 소비자의 취향을 정확히 인식할 수 있는지 연구한다.

2. 서론

A. 제안 배경 및 필요성

i. 사용자 맞춤형 서비스의 부상

1. 2013 년까지 전세계적으로 4.4 제타바이트의 데이터가 형성되었으며, 2020 년까지는 그 10 배인 44 제타바이트까지 그 양이 증가할 예정이다[1]. 한 명의 사람은 수많은 데이터를 남기고, 그러한 데이터를 모아 여러 기업 및 정부가 가공하여 유의미한 정보를 읽으려고 노력하고 있다.
2. 특히 이러한 데이터는 개인의 니즈를 정확히 파악하여 개인에게 딱 맞는 서비스를 제공하기 위한 수단으로 사용되고 있다. 많은 기업들이 소비자의 변화하는 기호를 인지하기 위하여 소비자가 제품 및 서비스와 관련하여 남긴 데이터를 적극적으로 수집하고 있다.

ii. 소비자 행동특성에 기반한 접근

1. 넷플릭스(Netflix)의 부회장 토드 옐린(Todd Yellin)이 최근 인터뷰에서 “주소, 나이, 성별과 같은 정보는 모두 쓰레기다”라는 다소 과격한 표현을 통해 기존의 빅데이터 분석 방식의 틀을 비판했다[2]. 소비자의 니즈를 정확히 파악하기 위해서는, 더이상 전통적인 인구통계학적 데이터에 얽매여서는 안 된다는 의미로 풀이된다. 전체 시청되는 콘텐츠 중 75%가 추천 시스템에 의하여 소비되고 있는 넷플릭스에서 이러한 발언을 한 것은 상당한 의미가 있다[3].

2. 기존의 인구통계학적 데이터는 계속해서 수집되고 있으나, 소비자의 행동패턴에 대한 관심이 더욱 커지고 있는 것도 사실이다. 어떠한 콘텐츠에 주목하고, 소비하고, 재구매하는지에 대한 정보를 꾸준히 추적하여 소비자 의견을 반영하려는 기업들이 많아지고 있다.

iii. 추천 시스템의 비교

1. 소비자의 여러 요소를 파악하여 이들이 가장 원하는 제품 및 서비스를 제공하려는 추천 시스템은 다양한 알고리즘으로 이루어져 있다. 본 연구는 이러한 알고리즘들을 동일한 입력데이터값과 동일한 기준으로 비교하여 어떠한 알고리즘이 가장 정확한 추천 시스템인지를 밝히고자 한다.

iv. 오픈 소스의 활용

1. 개발자들 사이의 더 나은 결과물을 내기 위한 협력은 지속되어왔다. 오픈 소스는 이러한 협력을 더욱 편리하고 효과적으로 돕고 있다. 소스를 공개하여 다른 개발자들에게 조언을 받고 그에 기반한 수정사항들을 반영한다. 본 연구 역시 그렇게 공개된 오픈 소스들을 비교하고자 한다.

B. 졸업작품의 목표

i. 본 연구는 무비렌즈(http://movielens.org/)에서 수집된 정보를 입력데이터로 활용한다. 특정 영화에 대한 소비자들의 평가이력을 주축으로 하는 다양한 정보를 활용하여, 동일한 입력데이터에 대해 가장 높은 정확도를 보이는 추천 알고리즘을 밝혀내고자 한다.

ii. 비교 대상은 다음과 같다.

1. Surprise(http://surpriselib.com)
2. Python-recsys(http://ocelma.net/software/python-recsys/build/html/)

C. 졸업작품의 전체 overview

i. 서론

ii. 비교 대상 알고리즘 소개

iii. 데이터셋 소개 : 입력 데이터의 출처, 사용자 수, 영화 수, 태그 수, 최종 업데이트 일자 등을 포함한 메타데이터를 소개한다.

iv. 구현 방식 소개 : 입력 값을 어떻게 매트릭스화 하고 분석하는지 표현한다.

v. 구현 결과 : 여러 실험 조건 내에서 각 데이터셋에 대한 예측의 정확도를 표현한다.

vi. 결과 분석 : 알고리즘이 각 조건 내의 데이터셋에 대해 추측한 결과의 정확도를 분석하고, 그 차이의 이유를 파악한다.

vii. 제언 : 이러한 연구의 효용성 및 기대효과를 논한다.

3. 관련연구

A. 추천 시스템(Recommendation System)이란, 사용자가 특정 아이템(도서, 보험, 영화, 앱 등)에 부여할 것으로 기대되는 '선호도'나 '점수'를 측정하는 것을 목표로 하는 정보 필터링 시스템의 하위 개념이다.

B. 추천 시스템은 보통 세 가지 필터링 기법으로 나뉜다.

i. 내용 기반 필터링(Content-based Filtering) : 내용 기반 필터링은 사용자의 과거 구매 이력과 프로파일 정보를 기반으로 아이템을 추천하는 시스템으로[4], 'Cognitive Filtering'이라고도 불리운다. 사용자의 선호도를 명확히 반영한다는 장점이 있으나, 프로파일 정보 혹은 구매 이력 둘 중에 하나라도 없으면 추천이 불가능하며, 새로운 아이템에 대한 추천의 정확도가 낮다(Overspecialization Problem).

ii. 협업 필터링(Collaborative Filtering): 사용자들의 평가이력을 기반으로 아이템을 추천하는 협업 필터링은 유사한 사용자들 사이의 연관성에 집중하기 때문에 'Social Filtering' 이라고도 불리운다. 협업 필터링은 영화 등 콘텐츠 추천에 가장 많이 사용되는 방식으로, 사용자와 사용자 간의 선호도를 사용하는 사용자 기반(User-based) 협업 필터링 방법과 항목들 간의 선호도를 사용하는 아이템 기반(Item-based) 협업 필터링 방식이 있다[5]. 다음 항목에서 더 자세히 다룬다.

iii. 하이브리드 추천 시스템(Hybrid Recommender System): 내용 기반 필터링과 협업 필터링은 결합한 하이브리드 추천 시스템은 1) 서로 다른 필터링 기법으로 각각의 결과물을 보여주는 형태, 2) 한 필터링 기법을 거친 후 다음 필터링 기법으로 단계적으로 진행하는 형태, 3) 두 필터링 기법이 완전히 혼합되어 기술적인 구분 없이 처리하여 하나의 결과물을 얻는 형태, 세가지로 분류된다. [6]

C. 협업 필터링(Collaborative Filtering)

i. 협업 필터링은 평가점수 매트릭스 구성, 유사도 계산 및 이웃집단 형성, 추천리스트 작성 세 단계로 나뉜다. [7]

1. 1 단계 : 평가이력 매트릭스 구성

A. 사용자가 특정 영화에 대해 남긴 평가 이력을 매트릭스화 한다. Ø 는 이력을 남기지 않았음을 의미하며, 나머지는 사용자가 남긴 평점을 담는다.

	Movie 1	Movie 2	Movie 3	Movie 4	…	Movie M
User 1	$Rating_{1,1}$	$Rating_{1,2}$	$Rating_{1,3}$	$Rating_{1,4}$	…	$Rating_{1,M}$
User 2	$Rating_{2,1}$	$Rating_{2,2}$	Ø	$Rating_{2,4}$	…	$Rating_{2,M}$
User 3	Ø	Ø	$Rating_{3,3}$	Ø	…	$Rating_{3,M}$
User 4	$Rating_{4,1}$	Ø	$Rating_{4,3}$	$Rating_{4,4}$	…	$Rating_{4,M}$
User 5	$Rating_{5,1}$	$Rating_{5,2}$	$Rating_{5,3}$	Ø	…	Ø
…	…	…	…	…	…	…
User N	$Rating_{N,1}$	Ø	$Rating_{N,3}$	$Rating_{N,4}$	…	$Rating_{N,M}$

표 1 : 평가이력 매트릭스 구성

2. 2 단계 : 유사도 및 이웃집단 분석

A. 특정 영화에 대해 사용자가 남긴 평가값의 유사도에 따라서 두 사용자의 거리를 알 수 있다. 평가값이 유사할수록 두 사용자의 거리가 가깝다고 표현하고, 평가값이 상이할수록 두 사용자의 거리가 멀다고 표현할 수 있다. 거리가 가까운

사용자들끼리 이웃집단으로 분류되며, 이웃집단 내에서는 비슷한 취향을 가지고 있다고 판단한다.

B. 밑의 표 처럼 두 사용자 간의 평가 이력이 특정 범위 내의 유사도를 갖고 있다면 관계가 존재한다고 판단하며, 그 거리가 가까울수록 유사도가 높음을 의미한다. 사전에 상정해 둔 범위 내의 거리라면, 하나의 이웃집단으로 클러스터링 하여 추천의 지표로 활용할 수 있다.

C. 이웃집단으로 판단하는 기준은 크게 두가지가 있다. 하나는 전체 중 가장 유사한 n 개를 하나의 이웃으로 묶는 Top N Best Neighbors 기법이고, 다른 하나는 특정 수치를 지정해 두고 그보다 높은 유사도를 보이는 개체들을 묶는 Threshold Based Selection 기법이 있다.

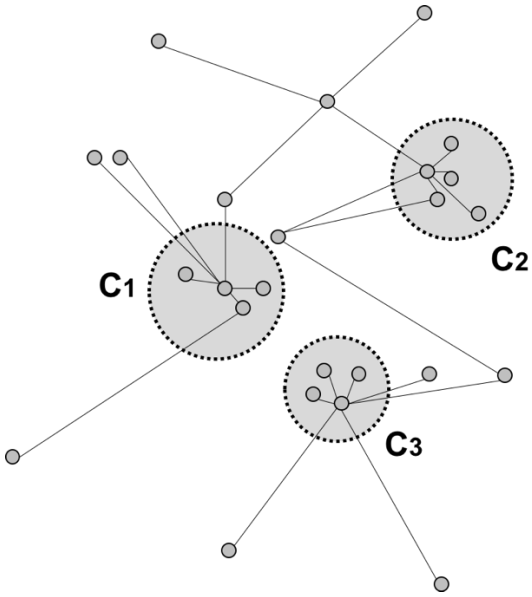


표 2 : 이웃집단의 클러스터링

3. 3 단계 : 추천 항목 산출

A. 동일 이웃집단 내에 있는 사용자들 사이에서, 특정 사용자가 아직 평가를 내리지 않은 영화에 대해서 이웃집단 내 다른 사용자가 내린 평가와 유사할 것이라고 가정한다. 이러

한 가정 하에 추천 항목을 산출하여 추천 리스트를 만든다.

ii. 문제

1. 희소성 (Sparsity) : 정보의 입력데이터의 부족이 야기하는 추천의 부정확성이 존재한다. 따라서, 구매이력이 존재하지 않는 새로운 아이템이 계속되서 추천리스트에서 탈락하는 현상이 발생할 수 있다. 뿐만 아니라 사용자의 수가 증가함에 따라 평가 자료의 희소성 역시 증가하며 콘텐츠 기반 필터링의 성능을 감소시키는 원인이 된다.[8]
2. 확장성 (Scalability) : 사용자 사이의 유사도를 기반으로 하기 때문에, 사용자의 수가 늘어날수록 행렬의 크기가 커져 계산하는데에 시간적 비용이 커진다. 따라서 시스템의 확장성에 문제가 생긴다. 기존의 연구에서는 이러한 문제를 해결하기 위해 사용자–아이템 간이 아닌 사용자–장르 간의 행렬로 시스템의 부하를 낮추려는 노력이 있었다[9]. 다만 음악, 영상, 시나리오 등의 다양한 매체의 복합체인 영화는 장르만으로 그 선호도를 추천하기 어렵다는 문제가 있다.
3. 예외성 (Gray Sheep) : 사용자가 남긴 다른 행동 특성으로 추측하기 어려울 정도로 매우 독특한 취향을 가지고 있는 경우를 의미한다. 이러한 경우에는 높은 정확도를 가진 알고리즘이라고 하더라도, 해당 사용자의 취향이 아웃라이어인 경우다. 따라서, 해당 경우로 분류되는 사용자는 논의 범주 내의 '정확도'를 분석하기 어렵게 만드는 요소다.

4. 제안 작품 소개

A. 데이터셋 메타데이터

i. 데이터셋은 무비렌즈(http://movielens.org)에서 수집된 실사용자들의 평가이력을 미네소타 대학 연구진이 모은 것을 활용한다. [9]

ii. 현재 확보된 데이터셋의 종류는 세 가지이며, 수집 시기 및 데이터의

사이즈에서 차이를 보인다.

iii. 세 데이터셋의 메타데이터

1. Ratings : 사용자가 한 영화에 부여한 평가점수로, 5 점 만점이다. 특정 사용자-영화-평가이력 정보에는 타임스탬프가 부여된다.
2. Users : 무비렌즈 서비스의 사용자로, 최소 20 개 영화에 평가를 남긴 사용자 중 연령, 성별, 직업에 편중되지 않고 무작위로 선정하였다. 각 사용자는 고유의 id 를 가지고 있으며, 그 외에 다른 정보는 없다.
3. Movies : 무비렌즈에서 서비스되는 영화들로, 각 영화는 고유한 id 와 복수의 장르를 가지고 있다. 장르는 어드벤처, 애니메이션, 아동, 코미디, SF, 로맨스, 드라마, 범죄, 호러, 미스테리, 스릴러 등으로 분류된다. 각 영화당 영화 데이터베이스인 imbd 와 tmbd 의 id 와도 연동이 되어있다. 해당 연결은 'links' 에서 확인할 수 있다.
4. Tags : 사용자가 영화에 부여한 free-text tagging 을 의미한다. 특정 사용자-영화-태그 정보에는 타임스탬프가 부여된다.
5. From/Until : 해당 데이터셋의 최초 평가이력과 최신 평가이력의 기간을 의미한다.
6. Generated : 해당 데이터셋이 수집 및 최초 가공된 최종 일자를 의미한다.
7. Last Update : 해당 데이터셋이 재가공 및 배포된 최종 일자를 의미한다.
8. DataSet2('ml-lastest')와 DataSet3('ml-latest-small')은 계속해서 업데이트 되는 데이터셋으로, 본 연구 이후에 데이터가 변동될 수 있으며, 그에 따라 산출값과 정확도도 변할 수 있다.

	DataSet1 ('ml-20m')	DataSet2 ('ml-latest')*	DataSet3 ('ml-latest-small')*
N(Ratings)	20000263	26024289	1000004

N(Users)	138493	270896	671
N(Movies)	27278	45843	9125
N(Tags)	465564	753170	1296
From	Jan 09, 1995	Jan 09, 1995	Jan 09, 1995
Until	Mar 31, 2015	Aug 04, 2017	Oct 16, 2016
Generated	Mar 31, 2015	Aug 04, 2017	Oct 17, 2016
Last Update	Oct 17, 2016	Aug 04 2017	Oct 17, 2016

표 3 : 데이터셋의 메타데이터

iv. 세 데이터셋의 공통적인 구조

userId	movieId	Rating	timestamp
1	2	3.5	1112486027
1	29	3.5	1112484676
1	32	3.5	1112484819
1	47	3.5	1112484727
…	…	…	…

표 4 : Ratings

userId	movieId	tag	timestamp
18	4141	Mark Waters	1240597180
65	208	dark hero	1368150078
65	353	dark hero	1368150079
65	521	noir thriller	1368149983
…	…	…	…

표 5: Tags

movieId	title	genres
1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy

2	Jumanji (1995)	Adventure|Children|Fantasy
3	Grumpier Old Men (1995)	Comedy|Romance
4	Waiting to Exhale (1995)	Comedy|Drama|Romance
...	...	...

표 6 : Movies

movieId	imdbId	tmdbId
1	114709	862
2	113497	8844
3	113228	15602
4	114885	31357
...	...	...

표 7: Links

B. 평가 방법

i. 두 데이터셋을 동일한 방식의 평가 방법을 활용하여 그 정확성을 측정한다.

ii. 현재 두 데이터셋 모두 행렬의 크기가 크고, 비어 있는 데이터 역시 많다. 그러한 특징에 맞추어 SVD (Singular Value Decomposition) 방식을 활용해 행렬의 차원을 줄여 유사 사용자를 그룹화한다.

iii. SVD 알고리즘 내에서 그 에러를 계산하는 방식은 추천 시스템 분석에 가장 많이 사용되는 MAE(Mean Absolute Error)와 RMSE(Root Mean Squared Error)를 활용한다. MAE 는 각 예상값과 실제값의 차이를 모두 더한 다음 총 아이템 수로 나누어 그 차례를 계산하는 방식이다. RMSE 는 각 에러의 평균치를 측정하는 2 차 계산식의 제곱근이다.

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i|$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}$$

5. 제안 작품 소개

A. 알고리즘 소개

i. Surprise

1. Surprise 는 Python Scikit 에 기반한 추천시스템 분석툴로, 강력한 API 가 제공된다. 머신 러닝 전문가 Nicolas Hug 가 개발했으며, 꾸준히 관련 가이드가 업데이트 되고 있다.
2. Dataset handling 에 특히 강점을 가지고 있으며, SVD(Singular-value decomposition)뿐만 아니라 PMF, NMF 와 같은 다양한 Matrix-factorzation 으로도 분석할 수 있다.
3. Baseline algorithm, neighborhood algorithm, neighborhood methods 및 여러 유사성 검정 모듈이 포함되어 있다.

ii. Python-recsys

1. Python-recsys 는 SVD 형태의 데이터를 분석하여 아이템을 추천하는 추천시스템으로, 스페인 출신 개발자인 Oscar Celma 가 개발한 라이브러리이다.
2. Divisi2 와 csc-pysparse 기반으로 제작되었으며, 두 라이브러리를 위해 Numpy 와 Scipy 역시 요구된다.
3. 일반적으로 활용되는 SVD 형태로 분석으르 진행함으로써, 다른 라이브러리에서의 명령어와 상당히 유사하게 개발되었다. 그러한 점에서 특별한 튜토리얼 없이 바로 이용할 수 있다는 장점이 있다.

B. 데이터셋 소개

i. 데이터셋 메타데이터

1. 데이터셋은 무비렌즈(http://movielens.org)에서 수집된 실사용자들의 평가이력을 미네소타 대학 연구진이 모은 것을 활용한다.
2. 현재 확보된 데이터셋의 종류는 세 가지이며, 세 가지 중 결과값이 가장 명확하게 산출되는 데이터셋을 최종보고서에서 활용한다.
3. 세 데이터셋의 메타데이터
   A. Ratings : 사용자가 한 영화에 부여한 평가점수로, 5 점 만점이다. 특정 사용자-영화-평가이력 정보에는 타임스탬프가 부여된다.
   B. Users : 무비렌즈 서비스의 사용자로, 최소 20 개 영화에 평가를 남긴 사용자 중 연령, 성별, 직업에 편중되지 않고 무작위로 선정하였다. 각 사용자는 고유의 id 를 가지고 있으며, 그 외에 다른 정보는 없다.
   C. Movies : 무비렌즈에서 서비스되는 영화들로, 각 영화는 고유한 id 와 복수의 장르를 가지고 있다. 장르는 어드벤처, 애니메이션, 아동, 코미디, SF, 로맨스, 드라마, 범죄, 호러, 미스테리, 스릴러 등으로 분류된다. 각 영화당 영화 데이터베이스인 imbd 와 tmbd 의 id 와도 연동이 되어있다. 해당 연결은 'links' 에서 확인할 수 있다.
   D. Tags : 사용자가 영화에 부여한 free-text tagging 을 의미한다. 특정 사용자-영화-태그 정보에는 타임스탬프가 부여된다.
   E. From/Until : 해당 데이터셋의 최초 평가이력과 최신 평가이력의 기간을 의미한다.
   F. Generated : 해당 데이터셋이 수집 및 최초 가공된 최종 일자를 의미한다.
   G. Last Update : 해당 데이터셋이 재가공 및 배포된 최종 일자를 의미한다.
   H. DataSet2('ml-lastest')와 DataSet3('ml-latest-small')

은 계속해서 업데이트 되는 데이터셋으로, 본 연구 이후에 데이터가 변동될 수 있으며, 그에 따라 산출값과 정확도도 변할 수 있다.

	DataSet1 (‘ml-20m’)	DataSet2 (‘ml-latest’)*	DataSet3 (‘ml-latest-small’)*
N(Ratings)	20000263	26024289	1000004
N(Users)	138493	270896	671
N(Movies)	27278	45843	9125
N(Tags)	465564	753170	1296
From	Jan 09, 1995	Jan 09, 1995	Jan 09, 1995
Until	Mar 31, 2015	Aug 04, 2017	Oct 16, 2016
Generated	Mar 31, 2015	Aug 04, 2017	Oct 17, 2016
Last Update	Oct 17, 2016	Aug 04 2017	Oct 17, 2016

표 8 : 데이터셋의 메타데이터

ii. 세 데이터셋의 공통적인 구조

userId	movieId	Rating	timestamp
1	2	3.5	1112486027
1	29	3.5	1112484676
1	32	3.5	1112484819
1	47	3.5	1112484727
…	…	…	…

표 9 : Ratings

userId	movieId	tag	timestamp
18	4141	Mark Waters	1240597180
65	208	dark hero	1368150078
65	353	dark hero	1368150079
65	521	noir thriller	1368149983

…	…	…	…

표 10: Tags

movieId	title	genres
1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
2	Jumanji (1995)	Adventure|Children|Fantasy
3	Grumpier Old Men (1995)	Comedy|Romance
4	Waiting to Exhale (1995)	Comedy|Drama|Romance
…	…	…

표 11 : Movies

movieId	imdbId	tmdbId
1	114709	862
2	113497	8844
3	113228	15602
4	114885	31357
…	…	…

표 12: Links

6. 구현 및 결과 분석

A. 구현 과정

i. 분석 기준

1. 입력데이터를 어떠한 방식으로 매트릭스화 할 것인지에 대해서는 여러 방법론이 있으나, 해당 연구에서는 가장 일반적인 SVD(Singular-value decomposition) 방식을 활용한다.

2. 데이터셋 전체를 동일한 규모로 나누어 서로 그 정확도를 검정하는 Cross-validation 방식을 활용하며, 일괄적으로 2 개, 3 개, 5 개, 10 개, 20 개의 데이터셋으로 나누어 검정하고 그 값을 비교한다.
3. 정확도를 분석하는 방식으로는 RMSE 와 MAE 를 활용하며, 이에 대한 자세한 설명은 다음 항목에서 논한다.

ii. RMSE 와 MAE : RMSE 와 MAE 는 Continuous variable 의 정확도를 분석할때 가장 일반적으로 활용되는 메트릭스 구조이다. 많은 연구 결과물에서 그 정확도를 RMSE 나 R-squared 형식으로 표현한다.

1. RMSE(Root Mean Squared Error)
   A. RMSE 의 정의는 다음과 같다.

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

   B. 평균적인 에러값들을 계산하는 2 차 방정식이다.
2. MAE(Mean Absolute Error)
   A. MAE 의 정의는 다음과 같다.

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

   B. 예측값과 실제값의 차에 대한 절대값들을 평균낸 값이다.
3. RMSE 와 MAE 비교
   A. 분석하려고 하는 값들의 예측 에러를 평균 낸다는 점에서 두 수치는 매우 유사하다.
   B. 둘 다 0 에서 무한 사이의 값을 가지고 있으며, 에러값이 양인지 음인지는 영향으르 주지 않는다.
   C. RMSE 는 평균값을 내기 전에 각 에러를 제곱하기 때문에 큰 에러값이 전체적으로 큰 영향을 미치게 된다.
   D. 그에 따라, 분산값이 증가할수록 RMSE 는 그 값이 더욱 커지는 반면, MAE 는 보다 안정적인 수치를 보여준다.

B. 구현 결과

i. Surprise 의 구현 결과 예시

1. Surprise 의 두번째 데이터셋(ml-latest) 분석 결과

	RMSE	MAE
FOLD 1	0.8184	0.6223
FOLD 2	0.8254	0.6285
FOLD 3	0.8189	0.6239
FOLD 4	0.8296	0.6299
FOLD 5	0.8226	0.6243
FOLD 6	0.8231	0.6257
FOLD 7	0.8195	0.625
FOLD 8	0.8209	0.6243
FOLD 9	0.824	0.6269
FOLD 10	0.818	0.6227
FOLD 11	0.8215	0.6251
FOLD 12	0.8211	0.6249
FOLD 13	0.8223	0.624
FOLD 14	0.8215	0.624
FOLD 15	0.8236	0.6264
FOLD 16	0.8241	0.6281
FOLD 17	0.8224	0.6236
FOLD 18	0.818	0.6237
FOLD 19	0.8211	0.6234
FOLD 20	0.8223	0.6262
MEAN	0.8219	0.6252

ii. Python-recsys 의 구현 결과 예시

1. Python-recsys 의 두번째 데이터셋(ml-latest) 분석 결과

	RMSE	MAE
FOLD 1	0.8566	0.6535
FOLD 2	0.8560	0.6527
FOLD 3	0.8512	0.6494
FOLD 4	0.8523	0.6521
FOLD 5	0.8574	0.6534
FOLD 6	0.8520	0.6494
FOLD 7	0.8513	0.6508
FOLD 8	0.8479	0.6480
FOLD 9	0.8526	0.6503
FOLD 10	0.8603	0.6559
FOLD 11	0.8555	0.6517
FOLD 12	0.8514	0.6509
FOLD 13	0.8576	0.6548
FOLD 14	0.8523	0.6495
FOLD 15	0.8554	0.6528
FOLD 16	0.8544	0.6535
FOLD 17	0.853	0.6515
FOLD 18	0.8529	0.6498
FOLD 19	0.8556	0.6539
FOLD 20	0.8478	0.649
MEAN	0.8537	0.6516

iii. RMSE MAE 평균값

1. Surprise

7. CROSS VALIDATION		ML-20M	ML-LATEST	ML-LATEST-SMALL
NFOLDS=2	RMSE	0.8619	0.8617	0.9111

2013312498 정수민

NFOLDS=3	MAE	0.6616	0.6584	0.7038
	RMSE	0.8457	0.8448	0.9034
NFOLDS=5	MAE	0.6478	0.6442	0.6967
	RMSE	0.8336	0.8332	0.8977
NFOODS=10	MAE	0.6378	0.6347	0.6905
	RMSE	0.8256	0.8252	0.8932
NFOLDS=20	MAE	0.6311	0.6281	0.6874
	RMSE	0.8220	0.8219	0.8996
	MAE	0.6285	0.6252	0.6840

표 13 : Surprise 의 정확도 비교

1. Python-Recsys

CROSS VALIDATION		ML-20M	ML-LATEST	ML-LATEST-SMALL
NFOLDS=2	RMSE	0.8662	0.8651	0.9023
	MAE	0.6657	0.6617	0.6938
NFOLDS=3	RMSE	0.8608	0.8597	0.9023
	MAE	0.6613	0.6571	0.6954
NFOLDS=5	RMSE	0.8579	0.8563	0.8984
	MAE	0.6588	0.6540	0.6917
NFOODS=10	RMSE	0.8559	0.8546	0.8966
	MAE	0.6574	0.6528	0.6899
NFOLDS=20	RMSE	0.8552	0.8946	0.8946
	MAE	0.6567	0.6891	0.6891

A. 결과값 비교

ii. RMSE MAE 평균값

A. Surprise

CROSS VALIDATION		ML-20M	ML-LATEST	ML-LATEST-SMALL
NFOLDS=2	RMSE	0.8619	0.8617	0.9111
	MAE	0.6616	0.6584	0.7038
NFOLDS=3	RMSE	0.8457	0.8448	0.9034
	MAE	0.6478	0.6442	0.6967
NFOLDS=5	RMSE	0.8336	0.8332	0.8977
	MAE	0.6378	0.6347	0.6905
NFOODS=10	RMSE	0.8256	0.8252	0.8932
	MAE	0.6311	0.6281	0.6874
NFOLDS=20	RMSE	0.8220	0.8219	0.8996
	MAE	0.6285	0.6252	0.6840

표 14 : Surprise 의 정확도 비교

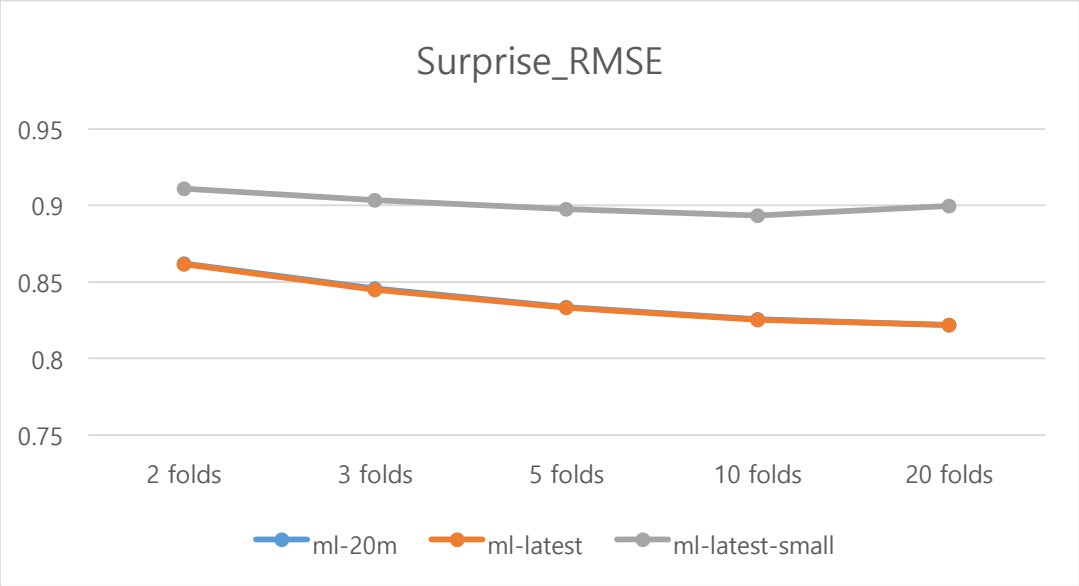
iii. Python-recsys

CROSS VALIDATION		ML-20M	ML-LATEST	ML-LATEST-SMALL
NFOLDS=2	RMSE	0.8662	0.8651	0.9090
	MAE	0.6657	0.6617	0.7017
NFOLDS=3	RMSE	0.8608	0.8597	0.9023
	MAE	0.6613	0.6571	0.6954
NFOLDS=5	RMSE	0.8579	0.8563	0.8984
	MAE	0.6588	0.6540	0.6917
NFOODS=10	RMSE	0.8559	0.8546	0.8966
	MAE	0.6574	0.6528	0.6899
NFOLDS=20	RMSE	0.8552	0.8537	0.8946
	MAE	0.6567	0.6516	0.6891

표 15 : Python-recsys 의 정확도 비교

i. Surprise 의 결과치 분석

1. RMSE



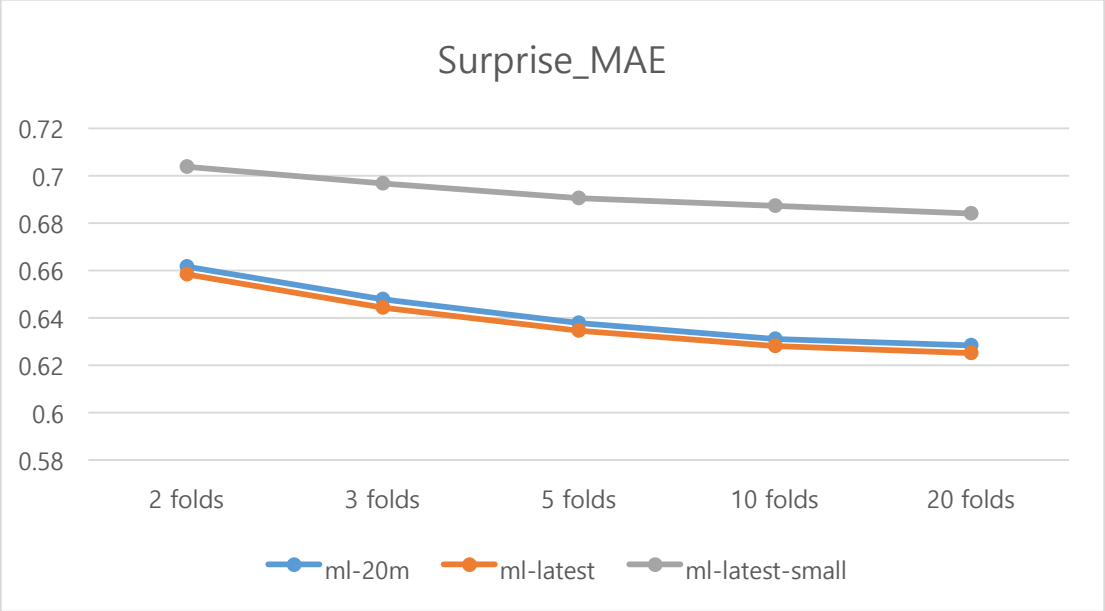
A. 동일 데이터 내에서 fold 값이 커질수록 그 정확도가 높아진다.

B. 데이터셋의 규모가 커질수록 그 정확도가 높아진다.

C. 에러가 0.8 에서 0.95 사이로 분포했다.

D. 데이터셋의 규모가 유사한 'ml-20m'과 'ml-latest'의 정확도가 각 데이터 분할 값에 따라 거의 유사하게 나타난 것으로 보아, Surprise 의 RMSE 는 분석 데이터 규모와 강력한 상관관계가 있음을 알 수 있다.

2. MAE



A. 동일 데이터 내에서 Cross Validation 을 위한 데이터 분할이 많아질수록 그 정확도가 높아진다.

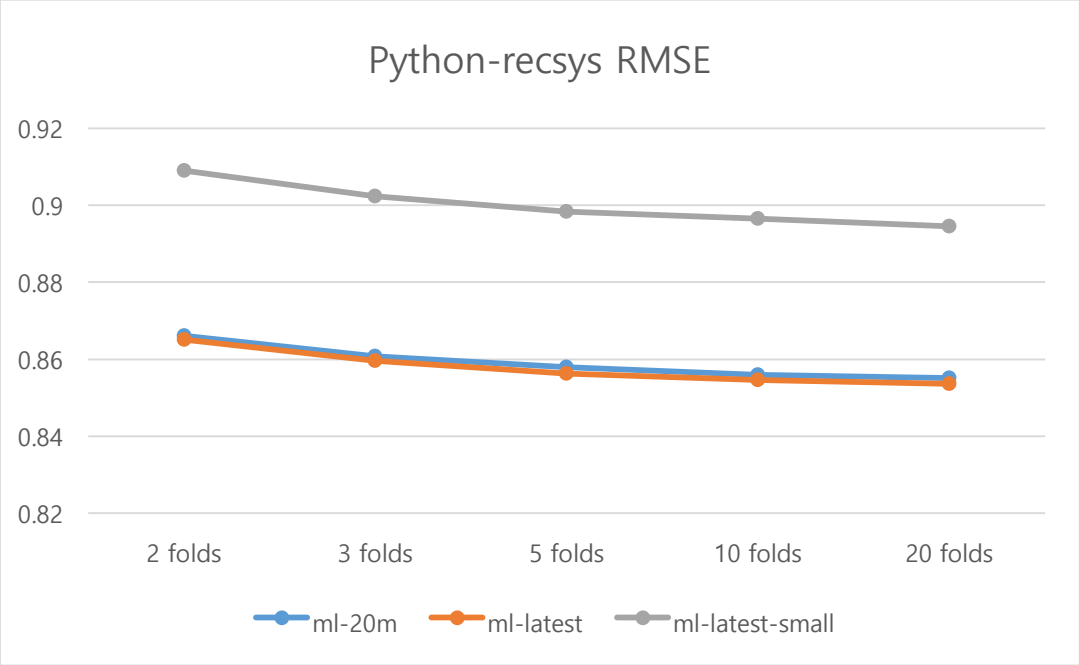
B. 데이터셋의 규모가 커질수록 그 정확도가 높아진다.

C. 정확도는 0.64 에서 0.72 사이에서 분포했다.

D. MAE 역시 RMSE 와 마찬가지로 분석 데이터의 사이즈가 유사하다면 그 정확도 역시 매우 유사하게 산출된다.

ii. Python-recsys 의 결과치 분석

3. RMSE



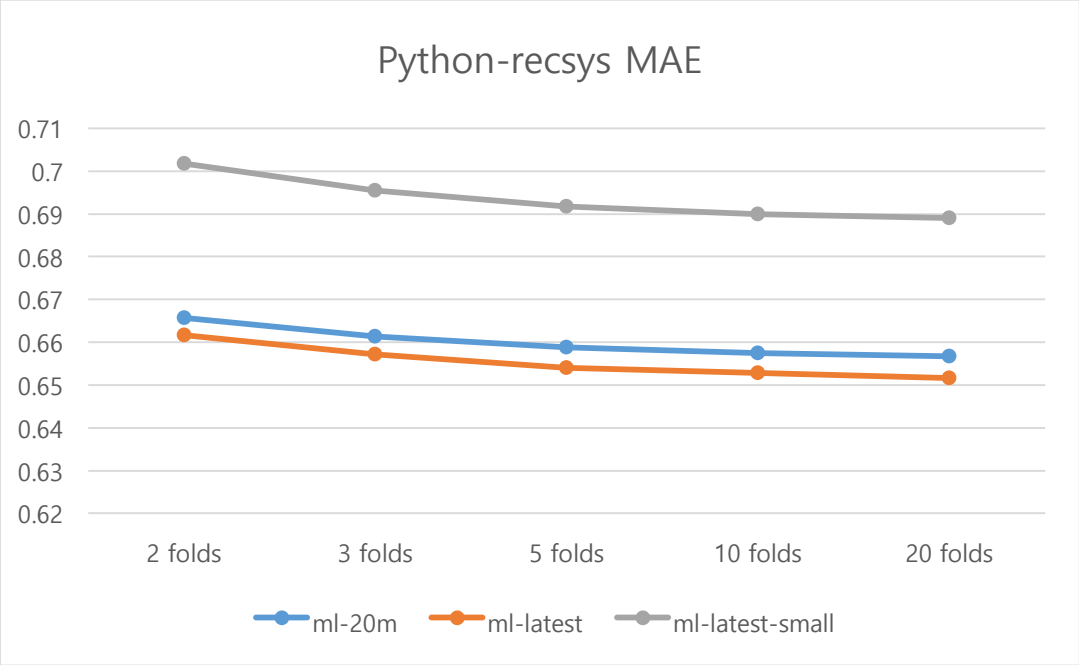
A. 동일 데이터 내에서 fold 값이 커질수록 그 정확도가 높아진다.

B. 데이터셋의 규모가 커질수록 그 정확도가 높아진다.

C. 에러가 0.84 에서 0.92 사이로 분포했다.

D. 데이터셋의 규모가 유사한 'ml-20m'과 'ml-latest'의 정확도가 각 데이터 분할 값에 따라 거의 유사하게 나타난 것으로 보아, Python-recsys 의 RMSE 는 분석 데이터 규모와 강력한 상관관계가 있음을 알 수 있다.

4. MAE



A. 동일 데이터 내에서 Cross Validation 을 위한 데이터 분할이 많아질수록 그 정확도가 높아진다.

B. 데이터셋의 규모가 커질수록 그 정확도가 높아진다.

C. 에러는 0.65 에서 0.71 사이에서 분포했다.

D. Surprise 의 MAE 에 비해 데이터 사이즈의 유사도에 따른 정확도 유사도의 관계성이 낮다.

iii. 동일 Cross-validation 기준 Surprise 와 Python-recsys 양자비교

5. 데이터셋 규모의 차이

A. 규모가 큰 'ml-20m'이나 'ml-latest' 데이터 셋에서 항상 Surprise 가 Python-recsys 에 비하여 더 좋은 정확도를 나타냈다. 따라서, 데이터 셋의 규모가 20 만개 이상으로 큰 경우에는, Surprise 를 활용하는 것이 더 바람직하다.

B. Cross-validation 을 위해 데이터를 여러 사이즈로 분할했을 때, Surprise 는 분할의 영향을 상대적으로 더 많이 받았다. 그러나 Python-recsys 는 그 차이가 적어 보다 안정적인 정확도를 나타냈다.

6. RMSE 와 MAE 에 따른 차이

A. 모든 경우에 RMSE 가 MAE 보다 그 값이 컸다. 이는 RMSE 와 MAE 의 기본적 특성에서 기반하는것으로, 항상 RMSE 는 MAE 보다 크며, 에러들이 모두 같은 수준의 양을 가졌을때만 두 값이 동일하다.

8. 결론 및 소감

A. 결론

i. 본 연구는 추천 시스템 오픈 소스를 비교하고, 데이터 셋에 따른 그 정확도를 파악한다.

ii. Python-recsys 는 Surprise 에 비하여 Cross-validation 의 영향을 더 적게 받는다.

iii. Surprise 는 보다 더 작은 규모의 데이터셋에서 높은 정확도를 보였으며, 데이터셋의 규모가 20 만개 이상으로 넘어갔을 때 Python-recsys 는 더 높은 정확도를 보였다. 따라서, 큰 데이터셋을 활용하는 경우에는 Python-recsys 를 활용하는 것이 더 바람직하다.

iv. 알고리즘의 외적으로는, Surprise 가 관련 업데이트가 더 활발하게 이루어지고 있어, 사용자의 입장에서는 더 쉽게 다가갈 수 있는 알고리즘이다.

B. 소감

i. 오픈 소스 활용의 어려움 : 파이썬은 대표적인 high level language 로 활발한 오픈 소스 공유가 일어나고 있다. 특히 과학 통계 방면에서 연구를 위해 자주 이용되고 있으며, 컴퓨터공학 뿐만 아니라 범학제적인 연구자들이 오픈소스 개발에 기여하고 있다. 다만, 해당 연구 영역에 대한 오픈 소스의 경우 유사한 모듈이나 알고리즘을 이해하고 있어야 이해할 수 있는 API 가 많았다. Surprise 같은 경우 이미 활용이 활발하게 이루어지고 있으며 API 역시 상세히 구술되어 있고 관련 포럼이 잘 구성되어 있다. 그러나 상대적으로 그 이용이 적은 Python-recsys 같은 경우, 포럼이 없으며 관련한 정보 역시 매우 파편화되어 있는 실

정이다. 특정 오픈 소스에 대해 보다 쉽게 정보를 얻을 수 있는 플랫폼이 형성되면 오픈 소스 사회에 더욱 도움이 될 것이라 기대한다.

ii. 연구를 위한 통계적 기반의 필요성 : 추천 시스템은 다양한 통계적 기법을 활용한 것이며, 해당 연구의 대상이 되었던 두 알고리즘 모두 통계적 기법을 모아둔 패키지이다. 그러한 점에서 그 정확도 및 그 정확도의 차이를 야기하느느 원인을 이해하기 위해서는, 통계적 기반이 필수적이다. 이와 같은 연구를 진행하기에 앞서 기존의 추천시스템에서 활용하는 통계적 기법을 참조한 것이 큰 도움이 되었다. 그러나 알고리즘의 개선을 위해서는 더욱 심층적인 통계적 기반이 필요함을 느꼈다.

iii. 데이터 분석의 기대되는 전망 : 빅데이터 시대의 도래에 따라 서비스의 개인화는 그 기세에 더욱 힘이 실렸다. 수집된 데이터를 기반으로 지금까지 이루지 못했던 높은 수준의 개인화가 실현될 것이다. 그 과정에서 여러 통계적 분석기법이 개발되고, 오류를 줄여나갈 것이다. 그러나 협업 필터링에서도 한 번 언급되었듯, 행동특성으로도 완벽히 추측할 수 없다. 몇 십년간 마케팅을 지배하고 있던 인구통계학적 데이터가 행동기반 데이터로 대체되었듯, 행동기반을 넘어서는 새로운 소비자 지표가 등장할 것으로 기대한다.

9. 참고문헌

[1] IDC, The Digital Universe of Opportunities: Rich Data and the Increasing Value of the Internet of Things. Retrieved September 20, 2017, from https://www.emc.com/leadership/digital-universe/2014iview/executive-summary.htm

[2] Morris, D. Z. (n.d.). Netflix says Geography, Age, and Gender Are 'Garbage' for Predicting Taste. Retrieved September 20, 2017, from http://fortune.com/2016/03/27/netflix-predicts-taste/

[3] Albanesius, C., & Chloe Albanesius (2012, April 09). 75 Percent of Netflix Viewing Based on Recommendations. Retrieved September 20, 2017, from https://www.pcmag.com/article2/0,2817,2402739,00.asp

[4] Mooney, R. J., & Roy, L. (2000). Content-based book recommending using learning for text categorization. Proceedings of the fifth ACM conference on Digital libraries - DL 00. doi:10.1145/336597.336662

[5] P. Mathew, B. Kuriakose and V. Hegde, "Book Recommendation System through content based and collaborative filtering method," 2016 International Conference on Data Mining and Advanced Computing (SAPIENCE), Ernakulam, 2016, pp. 47-52. doi: 10.1109/SAPIENCE.2016.7684166

[6] 이희정, "클러스터링 기반 사례기반추론을 이용한 추천시스템 개발", 대한산업공학회/한국경영과학회, 2004.

[7] Sarwar, B., Karypis, G., Konstan, J., &Riedl, J. (2000, October). Analysis of recommendation algorithms for e-commerce. In Proceedings of the 2nd ACM conference on Electronic commerce (pp. 158-167). ACM.

2013312498 정수민

[8] 정승윤, 김형중. (2017). Factorization Machine을 이용한 추천 시스템 설계. 한국디지털콘텐츠학회 논문지, 18(4), 707-712.

[9] 김충일 (Chung Il Kim) , 최남규 (Nam Gyu Choi) , 허유진 (Yu Jin Heo) , 신지훈 (Ji Hoon Sin) , 윤장혁 (Jang Hyeok Yoon), (주)엘지씨엔에스(구 LGCNS 엔트루정보기술연구소), 〈Entrue Journal of Information Technology〉 14 권 2 호 (2015), pp.71-82

[10] F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. ACM Transactions on Interactive Intelligent Systems (TiiS) 5, 4, Article 19 (December 2015), 19 pages. DOI=http://dx.doi.org/10.1145/2827872