

2019 January 19

조선대학교 IT융합대학 컴퓨터공학과

20144800 천승현

**2019동계 계절학기 OSSP   
최종 보고서**

영화 추천 시스템 구현

내용

[**개요** 2](#_Toc535675741)

[**기존 연구** 3](#_Toc535675742)

[**제안방법** 6](#_Toc535675744)

[- 문제 파악 7](#_Toc535675745)

[- 데이터베이스 다운로드 7](#_Toc535675746)

[- 필요 데이터 선별 9](#_Toc535675747)

[- 예측 모델에 맞게 프로그램 구현 9](#_Toc535675748)

[**인구통계학적 필터링** 9](#_Toc535675749)

[**컨텐츠기반 필터링** 10](#_Toc535675750)

[**협업필터링 – SVD** 10](#_Toc535675751)

[**실험 및 결과** 11](#_Toc535675752)

[- 실험 데이터에 대한 설명 (코드해석) 11](#_Toc535675753)

[**인구통계학적 필터링 구현** 11](#_Toc535675754)

[**컨텐츠 기반 필터링 구현** 14](#_Toc535675755)

[- 모델 개선 18](#_Toc535675756)

[**협업 필터링 – SVD 구현** 18](#_Toc535675757)

[- 개발환경(H/W 스펙, SW 버전) 23](#_Toc535675758)

[**느낀점** 23](#_Toc535675759)

[**참고** 24](#_Toc535675760)

[\*깃허브 히스토리 25](#_Toc535675761)

# **개요**

**영화의 데이터를 사용하여 사용자에게 영화를 추천해주는 프로그램 구현**

온라인 컨텐츠 시장이 성장함에 따라 소비자들은 더욱 많은 선택권을 가지게 되었다. 하지만 그만큼 모든 컨텐츠를 소모하기엔 시간적 여유가 따르지 않기 때문에 소비자들은 제한된 시간을 효율적으로 소모하기 위해 제한적으로 컨텐츠를 고를 수 밖에 없다. 대부분의 멀티미디어 서비스 웹사이트들은 이런 소비심리를 전략적으로 이용하기 위해 대부분 컨텐츠 추천 알고리즘을 사용하고 있다.

최근 가장 높은 성장률을 보이고 있는 넷플릭스(Netflix)는 대표적인 멀티미디어 컨텐츠 서비스 사이트이다. 넷플릭스는 강력한 컨텐츠 추천알고리즘을 사용하여 사용자(User)가 시청한 영상물들을 분석하여 트렌드를 파악하고, 취향에 맞는 컨텐츠를 추천해준다. 또는 다른 유저들이 많이 즐긴 컨텐츠들을 리스트로 상단에 노출시키는 마케팅을 하기도 한다.

* **Netflix : 대여되는 영화의 2/3가 추천으로부터 발생**
* **Google News : 38% 이상의 조회가 추천에 의해 발생**
* **Amazon : 판매의 35% 가 추천으로부터 발생**
* **Netflix Prize (~2009) Netflix에서 주관하는 경연대회로, 영화 선호도를 가장 잘 예측하는 협업 필터링 알고리즘에 수상 (US$1,000,000)**

이런 컨텐츠 추천 알고리즘은 소비산업에서 가장 필수적이면서 기본적인 마케팅 기법이 되었다. 미디어 산업뿐만 아니라 쇼핑, 스포츠, 부동산등 여러 분야에서 활발히 사용되고 있고 소비산업 뿐만 아니라 사용자들의 니즈를 분석하여 새로운 데이터를 수집할 수 있다는 점에서 다양하게 사용될 수 있다. 앞으로 이런 추천 알고리즘을 개량 시키고 발전시킴으로써 궁극적으로 소비자의 마음을 읽어 완벽하게 필요로 하는 요소들만 추천하는 인공지능 분야에도 사용될 수 있다.

# **기존 연구**

# ìí ì¶ì² ìê³ ë¦¬ì¦ì ëí ì´ë¯¸ì§ ê²ìê²°ê³¼

현재 가장 활발히 사용되고 있는 추천 알고리즘은 협업 필터링(Collaborative filtering)이다. 협업 필터링은 사용자들로부터 정보를 받아 사용자들의 관심사들을 자동적으로 예측하여 추천해주는 필터링 기법이다. 협업 필터링은 크게 유저 기반 협업 필터링(User-based-CF), 아이템 기반 필터링(Item-based-CF)로 나누어지고, 유저 기반 협업 필터링은 다시 능동적, 수동적 필터링으로 분류 된다. 이런 필터링 알고리즘들은 데이터를 분석하여 유사도 계산을 통해 추천하게 된다.

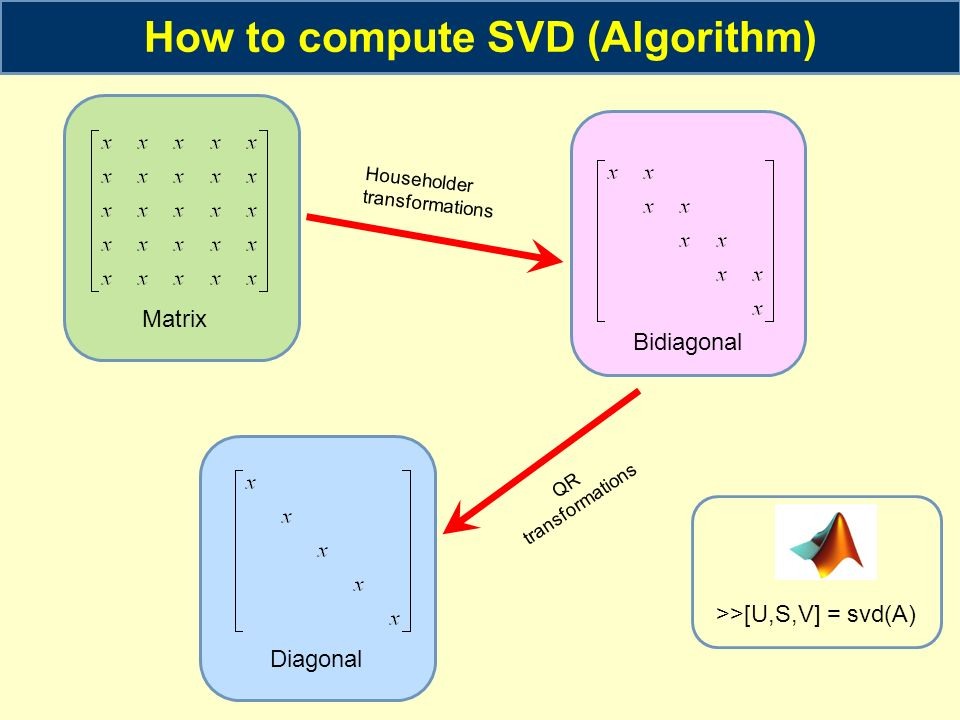
유저 기반 필터링(UBCF)는 사용자들간 공통된 아이템을 기준으로 얼마나 선호도가 일치하는지를 수치화 하여 그 점수를 사용한다. 예를 들어 A가 Z라는 블록버스터 영화에 평점5점을 주고, B가 5점을 준다면 둘의 선호도도 점수의 거리는 0이라고 할 수 있다. 만약 B가 3점을 줬다면, 두 사용자의 거리는 증가하게 된다. UBCF는 이것을 수치화 하여 더 가까운 유사도를 활용한 추천 방식이라고 볼 수 있다. 그러나 UBCF는 단점을 가지고 있는데, 만약 사용자가 최초로 시스템을 이용하는 거라면, 필요한 초기 데이터가 없기 때문에 추천을 하기가 어렵고, 또한 이런 데이터를 수집하는데 많은 시간이 소요 될 수 있다. 이러한 단점을 보완하는 것이 아이템 기반 필터링 이다.

아이템 기반 필터링(IBCF)는 추천 알고리즘에 사용자 데이터 대신 평가된 아이템 객체들이 데이터로 사용된다. 오늘날 대부분의 추천 시스템은 IBCF를 사용하고 있는데, 가장 대표적인 곳이 넷플릭스와 아마존이다. 아마존에서는 사용자가 구매한 상품들과 유사한 상품들을 선호할 것이라는 예측을 기반으로 하고 있다. IBCF는 UBCF와 달리 초기에 등록한 객체들의 데이터에 의존하기 때문에, 처음 시스템을 접하는 사용자라도 원하는 상품과 유사한 상품들을 쉽게 추천 받을 수 있다. 그러나 IBCF역시 상품들과의 유사도를 고려하였지만, 사용자들간의 선호도가 전혀 고려되지 않았기 때문에 특정 고객과 선호도가 매우 다르다면 상품들간의 추천 정확도가 떨어지고, 추천 시스템의 추천 능력이 떨어질 수 있다.

이외에도 최근에 인기가 증가하고 있는 능동적 필터링 방법이나, 미래에 가장 잠재력이 있다고 여겨지는 수동적 필터링 방법들도 존재한다. 이러한 필터링 방법들은 모두 각각 장단점이 있고, 완전하다고 볼 수 없다. 해당 필터링 방법론들의 장단점을 보기 쉽게 아래 표로 정리했다.

|  |  |  |
| --- | --- | --- |
|  | 장점 | 단점 |
| 유저 기반 필터링 | 알고리즘이 간단함, 아이템 정보 없이 추천이 가능 | 데이터가 많아질수록 연산이 복잡해짐, 신규 사용자의 데이터 공백 |
| 아이템 기반 필터링 | 아이템 자체의 정보 없이 추천 가능, 신규 사용자 추천 가능 | 데이터가 커질수록 연산이 복잡해짐, 초기 서비스 추천 정확도 떨어짐 |
| 능동적 필터링 | 신뢰성이 높음 | 편협 적인 유사도, 적은 피드백 |
| 수동적 필터링 | 능동적 필터링의 변수 제거 | 사용자 의존적 |

그렇다면 이런 협업 필터링들의 단점들을 어떻게 보완할 것인가? 가장 각광받는 것은 차원 축소 알고리즘이다. 차원 축소 알고리즘은 대표적으로 PCA(Principal component analysis), SVD(Singular Value Decomposition), NMF(Non-negative Matrix Factorization) 3가지가 있다. 이중에서 SVD(특이값 분해)라는 알고리즘을 사용하여 추천 시스템을 보완한다. SVD는 추천 시스템에 쓰이는 가장 대표적인 차원 축소 알고리즘으로, 매트릭스(행렬)을 이용하여 차원 축소를 수행한다.



SVD를 매우 핵심적이고 간략하게 설명하자면, 사용자와 상품간의 선호도를 수많은 데이터셋으로 정리하고 매트릭스로 표현한다. 이때 SVD를 사용하여, 대각 행렬을 축소하여 전체 처리해야 하는 데이터의 양을 줄일 수 있고, 아직 평가하지 않은 데이터에 대해서 평균 값을 이용하여 예상점수를 구할 수 있다. 추천 시스템은 이 점수를 이용하여 추천 점수가 높은 아이템을 사용자에게 추천하게 된다. 이런 차원 축소 알고리즘들은 다양한 상품에 대해 차원 축소를 적용함으로써 일반화 현상이 발생되고, 차원 축소가 많이 적용될수록 추천시스템은 더 효율적으로 작동한다.

# **제안방법**

스크린샷이(가) 표시된 사진

자동 생성된 설명

본격적으로 프로그램을 구현하기 전에 플로우 차트를 그려 전체적인 흐름을 파악한다. 프로젝트는 크게 5개의 섹션으로 진행되며, 섹션마다 각각의 결과물이 도출된다. 최종 결과물은 구현이 완료된 프로그램이며, 테스트와 개선을 마친 상태를 의미한다.

문제 파악: Kaggle(http://kaggle.com)에서 완료된 기존 kernel의 문제점을 파악하고 개선점 파악

->**데이터 베이스 다운로드**

필요 데이터 선별: 프로젝트를 진행하며 필요한 데이터만 추출하여 데이터베이스를 가공

->**가공된 데이터 셋**

데이터 변형: 코드를 작성하며 데이터구조를 필요에 따라 변형시킨다.

->**예측 모델**

프로그램 구현: 원하는 데이터구조로 변형된 데이터셋을 이용하여 코드 작성

->**완성된 초기 모델**

테스트: 모델을 테스트 및 개선하여 최종 결과물 도출

->**구현 완료된 최종 결과물**

## - 문제 파악

협업 필터링의 단점에서 문제점을 파악한다.

단점들을 종합하여 가장 큰 두가지의 문제점을 도출할 수 있다.

**1. 사용자의 취향은 항상 고정되어 있지 않고 축적된 데이터라도 일관성은 변동이 가능하다.**

**2. 사용자들의 성향을 파악할 수 없다. 단순한 1차원적 추천시스템만 구현 가능하다.**

이러한 사용자들의 자세한 성향이나 유사취향 문제는 차원축소 알고리즘을 사용하여 해결할 수 있다. 이방법은 기존의 사용자, 아이템 기반 필터링보다 많은 계산량이 요구되기 때문에 실제 적용하기까지 시간이 소요됐지만, 현재 컴퓨터의 기술력이 발전하고 사양이 상승함에 따라 적용 가능하게 되었다.

## - 데이터베이스 다운로드

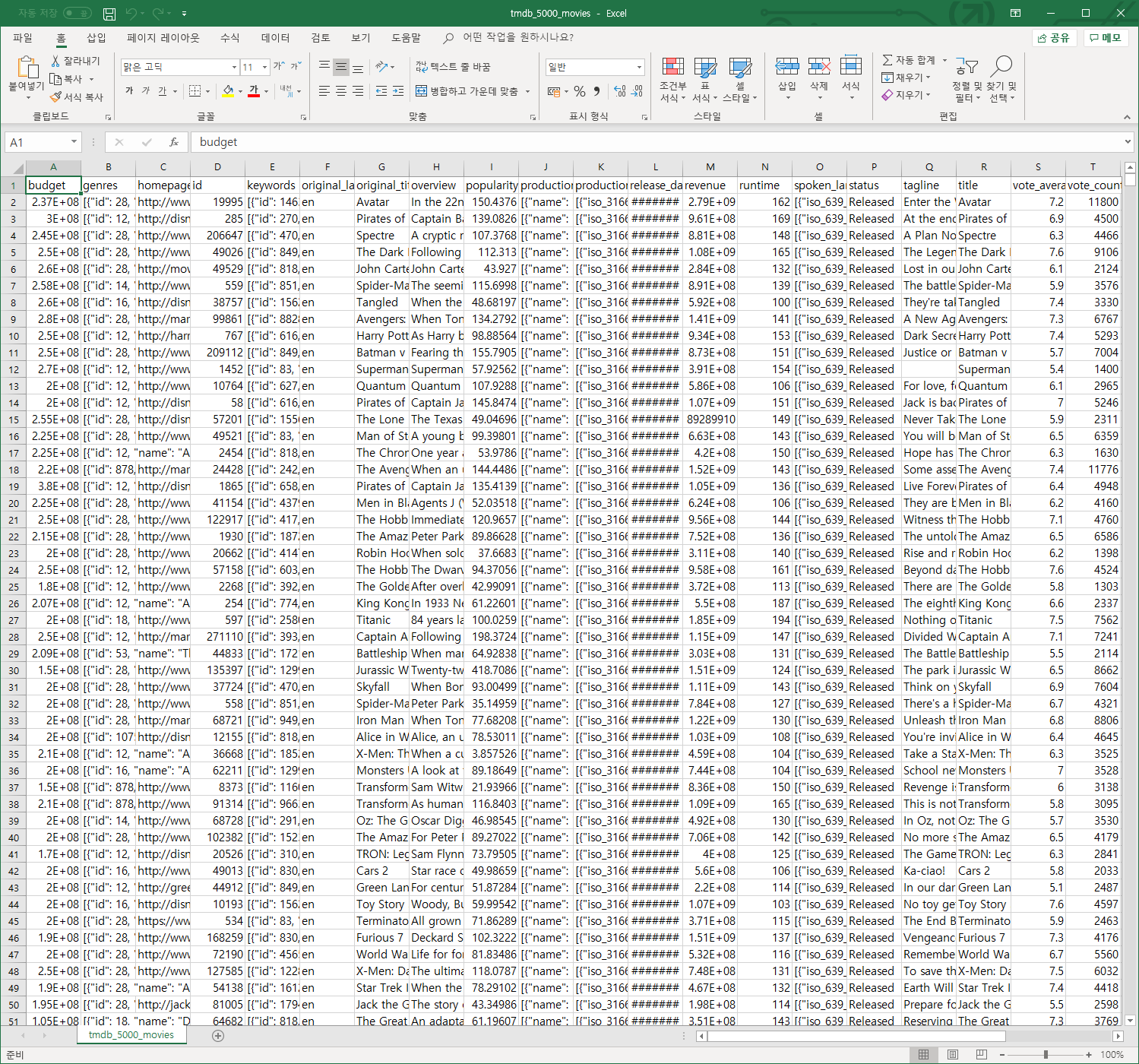
kaggle에서 제공하는 데이터 셋 다운로드

사람, 사진, 건물이(가) 표시된 사진

자동 생성된 설명

<https://www.kaggle.com/tmdb/tmdb-movie-metadata>

아마존닷컴의 자회사인 IMDB는 세계에서 가장 규모가 거대한 영화사이트를 운영한다. IMDB에서는 세계 모든 영화의 다양한 정보들을 제공하고 있다. 주연, 감독, 장르 뿐만 아니라 영화에 등장하는 장소, 음악, 수익, 심지어 모든 엑스트라 배우들의 크레딧도 제공한다. 이 데이터베이스를 선별하여 약 5000여편의 영화에 대한 데이터를 정리한 영화 데이터를 사용하기로 한다. 어떤 영화가 상업적으로 성공할 수 있는지, 어떤 영화가 높은 평가를 받는지 파악하기에 매우 좋은 자료이다.



데이터는 쉼표로 구분된 csv 텍스트 파일로 구성되어 있으며, 컬럼과 로우가 존재하는 데이터시트형태로 편집되어 있다. 컬럼들의 목록들은 다음과 같다.

* budget - 영화가 만들어진 예산.
* genre - 영화의 장르
* homepage - 영화 홈페이지의 링크
* id - movie\_id
* keywords -. 동영상과 관련된 키워드 또는 태그
* original\_language - 영화가 만들어진 언어.
* original\_title - 번역 또는 개봉하기 전의 영화 제목
* overview - 영화에 대한 간략한 설명
* popularity 영화 인기도
* production\_companies - 영화의 제작소
* production\_countries - 생산된 국가.
* release\_date - 출시 날짜.
* revenue - 창출되는 전 세계 매출액
* runtime - 영화의 상영 시간
* status - "개봉" 또는 "루머".
* tagline - 영화의 태그라인
* title 영화 제목.
* vote\_average - 영화가 받은 평균 등급
* vote\_count - 득표수

## - 필요 데이터 선별

영화를 추천해주기 위해서는 측정지표가 필요하다. 모든 영화의 점수를 계산하고 정렬한 뒤에, 사용자에게 적합한 가장 좋은 점수의 영화를 추천해준다. 하지만 현재 데이터에서 제공하는 기준치는 평균 기준치로 삼을 수 없다. 평점이 3표 밖에 없지만 8.9점인 영화와, 40표이상인 7.8점 영화영 비교할 때 3표를 받은 영화가 더 뛰어난 영화라고 볼 수 있지 않기 때문이다. 이러한 문제때문에 정확한 기준치를 계산하기 위해 IMDB 가중치를 계산법을 이용한다.

https://image.ibb.co/jYWZp9/wr.png

각각의 변수는 다음과 같다.

* v 영화가 받은 추천수
* m 차트에 적용 해야하는 최소 추천수
* R 영화의 평균 등급
* C 전체 보고서에서의 평균 추천수

위와 같은 계산법을 함수로 정의하여 데이터시트의 모든 영화별로 적용해서 컬럼에 추가한다.

## - 예측 모델에 맞게 프로그램 구현

### **인구통계학적 필터링**

스크린샷이(가) 표시된 사진

자동 생성된 설명

Deomograpic Filtering(인구통계학적 필터링)은 대다수의 사용자들이 내린 평점을 추천 시스템에 사용한다. 대다수의 사용자들이 많이 추천하고 선호하는 컨텐츠는 좋은 컨텐츠라는 전제하에 작동한다. 데이터셋에서 제공하는 평점과 투표수를 계산하여 지표를 만들고, 해당 지표로 영화를 정렬하여 우선순위를 선정하여 사용자에게 추천한다. 그러나 매우 단순한 추천시스템으로 사용자가 항상 상위권의 영화들을 선호할 것이라는 보장을 하지 않는다.

### **컨텐츠기반 필터링**

스크린샷이(가) 표시된 사진

자동 생성된 설명

Content-Based-Filtering(컨텐츠 기반 필터링)은 영화의 제목이나 개요, 감독, 장르, 주연배우등의 컬럼에서 키워드를 분석하여 각 영화별 유사도를 계산한다. 사용자가 영화를 선택하면 그 영화의 키워드를 분석하여 그와 비슷한 키워드를 가진 영화들을 리스트로 선별하여 추천한다. 컨텐츠 기반 필터링은 인구통계학적 필터링보다 더 높은 수준의 추천 알고리즘이라고 할 수 있지만 여전히 사용자의 취향이나 성향을 파악하지는 못한다.

### **협업필터링 – SVD**

스크린샷이(가) 표시된 사진

자동 생성된 설명

Collavorative Filtering(협업 필터링)은 이름 그대로 사용자와 사용자간, 아이템과 아이템간의 정보를 서로 이용한다. 사용자가 아이템들에 대해 평가를 내리면 그 아이템에 내려진 평가들을 분석하여 비슷한 아이템을 추천하거나, 사용자가 내린 다양한 평가들에 대하여 유사한 아이템을 추천해준다. 전자는 User-Based이고 후자는 Item-Based이다. 또한 협업필터링에는 다양한 축소알고리즘들을 사용하여 그 결과의 신뢰성을 높이는데, 여기서는 SVD(특이값분해)를 이용하여 알고리즘을 개선시키도록 하겠다.

# **실험 및 결과**

## - 실험 데이터에 대한 설명 (코드해석)

### **인구통계학적 필터링 구현**

스크린샷이(가) 표시된 사진

자동 생성된 설명

필요한 모듈들을 import해주고 데이터셋을 변수에 바인딩한다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

데이터프레임을 id를 기준으로 합병한다. 이제 df2에는 영화의 모든 정보 컬럼을 가지고있다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

df2를 출력해본 모습

스크린샷이(가) 표시된 사진

자동 생성된 설명

IDMB 가중치를 계산하기위해 변수에 컬럼값들을 계산하여 바인딩한다.

스크린샷이(가) 표시된 사진

자동 생성된 설명스크린샷이(가) 표시된 사진

자동 생성된 설명

메타스코어를 score 컬럼으로 이름 짓고 데이터셋에 추가하여 내림차순 출력을 한다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

해당 데이터 프레임을 matplotlib로 그래프 표현을 해보았다. 위 그래프를 볼 때 사용자에게 가장 선호 하는 영화는 ‘Minions’, ‘Interstellar’… 등등 이며 해당 영화들을 사용자에게 추천해 줄 수 있을 것이다.

### **컨텐츠 기반 필터링 구현**

스크린샷이(가) 표시된 사진

자동 생성된 설명

df2의 ‘overview’컬럼을 확인하자. 해당 영화의 개략적인 개요가 적혀있다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

데이터셋의 모든 영화의 개요를 각각의 문자열로 분할한다. 총 4803건의 영화에 20978개의 단어들로 이루어진 새로운 행렬을 생성한다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

코사인유사도와 유사한 계산결과를 리턴하는 linear\_kernel을 import하고 df의 유사도를 계산한다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

이제 영화를 추천받기 위한 시스템을 구현하기 위해 함수를 정의한다. 추천 프로그램은 영화제목을 입력 받고, 유사성 점수를 이용하여 영화 10편을 추천한다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

함수가 동작한 결과. 하지만 ‘개요’만을 키워드로 사용하면 1차원적이고 단순한 추천밖에 할 수 없다. 여기에 ‘감독’ ‘장르’ ‘주연배우’..등의 키워드를 추가하여 더 신뢰성 있는 추천리스트를 얻도록 하겠다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

‘특징’이라는 변수를 선언하고 각 객체의 키워드들을 바인딩 한다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

감독의 이름을 가져온다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

주연배우 3인의 이름을 가져온다.

스크린샷이(가) 표시된 사진

자동 생성된 설명스크린샷이(가) 표시된 사진

자동 생성된 설명

데이터프레임에 특징 변수를 추가하여 확인한다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

이제 유사도를 계산하기위해 모든 항목의 문자열들을 변환하고 공백을 제거한다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

스크린샷이(가) 표시된 사진

자동 생성된 설명

스크린샷이(가) 표시된 사진

자동 생성된 설명

코사인 유사도를 적용한 함수를 사용하여 새로운 데이터셋의 유사도를 계산한다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

이제 다시 개선된 코사인 유사도 점수를 사용하여 함수에 사용한다. ‘개요’만을 키워드로 사용하여 유사도를 측정한 추천영화 리스트보다 ‘감독’ ‘주연배우3인’ ‘장르’의 키워드를 추가적으로 사용하여 유사도를 계산하여 추천한 리스트가 훨씬 신뢰성이 높을 것이다.

## - 모델 개선

### **협업 필터링 – SVD 구현**

#해당 코드부터 추가적으로 직접 작성한 개선 코드이다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

SVD모듈을 사용하기위해 Surprise를 프롬프트에 install한다. 그리고 새로운 데이터셋을 바인딩한다.



해당 데이터셋을 reader클래스에 로드하고 교차검증을 5회 실시하기 위해 분할한다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

evaluate클래스를 사용하여 RMSE와 MSE를 계산한다. 평균 RMSE가 0.89로 사용하기 적합한 결측치를 가지고 있다고 정의한다. 이제 데이터셋을 학습시킨다.

스크린샷이(가) 표시된 사진

자동 생성된 설명스크린샷이(가) 표시된 사진

자동 생성된 설명

userId가 1인 사용자가 평점을 내린 항목들을 확인해본다.

스크린샷이(가) 표시된 사진

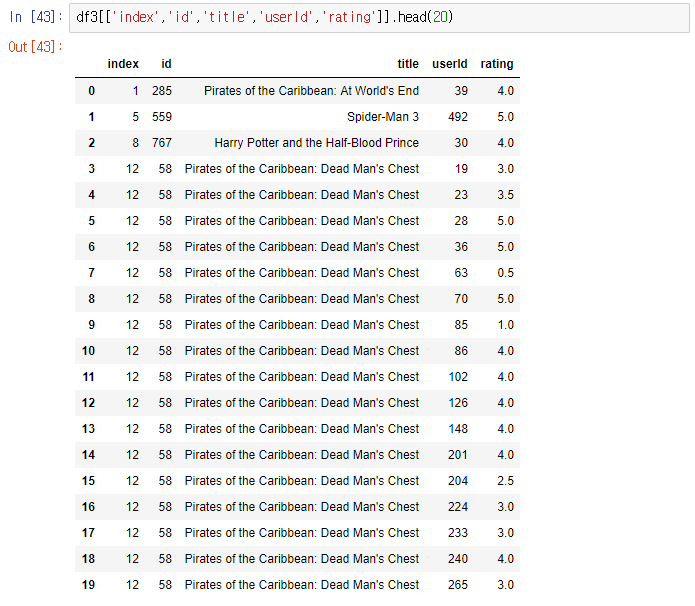
자동 생성된 설명

predict클래스를 사용하여 사용자의 평가 예측 점수를 확인한다. 사용자가 movieId가 302인 영화에 대해 약 2.7점의 점수를 매길 것이라고 예측할 수 있다. 해당점수를 이용하여 영화 제목 리스트를 추천하는 프로그램을 작성하자.

스크린샷이(가) 표시된 사진

자동 생성된 설명

기존 데이터셋에서는 유저들이 내린 평가에 movieId별로 제목(title)이 없으므로 기존 데이터셋에서 가져와서 합병한뒤 새로운 데이터셋을 생성한다.



스크린샷이(가) 표시된 사진

자동 생성된 설명

선택한 영화의 예측 평가점수를 추출한다. 소수점 둘째자리에서 반올림하여 첫째자리 까지만 출력한다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

이제 추천 시스템을 함수로 정의한다. 먼저 사용자와 영화를 입력 받고 예측점수를 svd클래스에서 받아와서 유사한 평가점수를 받아 오기 위해 소수점 자리를 0 or 5로 바꿔준다. 그 뒤에 예측점수와 똑 같은 점수들을 가지고 있는 영화들을 데이터프레임에서 가져와서 새로운 데이터셋을 생성하고 중복을 제거한다. 그 뒤에 함수를 실행시킬 때마다 예측점수와 같은 점수를 가진 영화제목들을 리스트에서 무작위로 5개 출력하도록 한다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

## - 개발환경(H/W 스펙, SW 버전)

H/W

|  |  |
| --- | --- |
| 하드웨어 | 스펙 |
| CPU | Intel i7-8700K |
| GPU | GTX-1060C |
| RAM | 8G |
| STORAGE | SSD 250GB, HDD 2TB |

SW

|  |  |
| --- | --- |
| 소프트웨어 | 버전 |
| 윈도우10 | Edu 1709 |
| Anaconda Navigator | 1.9.6 |
| Jupyter Notebook | 5.5.0 |
| Python | 3.6.5 |

# 

# **느낀점**

kaggle에서 데이터셋을 살펴보던 도중 평소에 취미로도 즐기고 있고 친근하게 다가갈 수 있는 영화라는 주제로 프로젝트를 진행하겠다고 마음먹을 때만 해도 아무것도 모르고 있었다. 추천시스템이라는 단순하게 보이는 서비스 안에는 내가 모르는 수많은 알고리즘과 수학적 요소들로 가득차 있었다. 처음 추천 알고리즘을 이해하기 위해 적용되는 유사도 계산을 공부하게 되고, 다음으로 차원축소 알고리즘에 대해 알게 되고, 그 다음에 개선시키기 위해 특이값분해라는 선형대수학 개념까지 공부해야 했다. 물론 완벽하게 이해하지 못하였지만, 최대한 최종발표에서 강의를 같이 수강 하고 있는 학우들이 쉽게 이해하기 위해 최대한 수학적 개념을 지양하고 직관적으로 이해할 수 있는 발표를 준비하였다.

해당 프로젝트를 진행하면서, 알고리즘과 컴퓨터과학, 머신러닝, 데이터분석등의 학문들은 전부 깊게 연결 되있다는 걸 알게 되었고, 그 중심에는 기본적인 자료구조와 알고리즘, 수학적개념이 필수적이라는 것을 알게 되었다. 또한 깃허브를 중심적으로 활용한점이 개인적으로 큰 경험이 되었다고 생각했다. git저장소를 활용할 줄 알고, 어디서든 편하게 자료를 받아서 서로 공유하고 개발할 수 있다는 점이 굉장히 매력적이었다. 앞으로도 꾸준히 내 github를 관리하여 나만의 경쟁력 있는 포트폴리오로 가꾸고 싶다는 생각을 했다.

# **참고**

http://darkpgmr.tistory.com/106

＃SVD(Singular value Decomposition) 설명

https://www.kaggle.com/ibtesama/getting-started-with-a-movie-recommendation-system/notebook

＃영화 추천 시스템 시작하기 kernel

https://ko.wikipedia.org/wiki/%EC%BD%94%EC%82%AC%EC%9D%B8\_%EC%9C%A0%EC%82%AC%EB%8F%84

＃코사인 유사도 위키백과

http://docs.likejazz.com/cosine-sim/

#코사인 유사도의 의미

https://ko.wikipedia.org/wiki/%ED%98%91%EC%97%85\_%ED%95%84%ED%84%B0%EB%A7%81

＃협업 필터링(collaborative filtering) 위키백과

https://m.blog.naver.com/a2zygote/220922530433

＃선형성과 비선형성의 개념

http://blog.naver.com/PostView.nhn?blogId=tjdudwo93&logNo=221051481147

＃SVD와 Kernel의 의미

https://proinlab.com/archives/2103

＃User-based CF, Item-based CF

http://leebaro.tistory.com/entry/SVD%EB%A5%BC-%EC%9D%B4%EC%9A%A9%ED%95%9C-%EC%B6%94%EC%B2%9C-%EC%8B%9C%EC%8A%A4%ED%85%9C-%EA%B5%AC%ED%98%84%ED%95%98%EA%B8%B0

＃SVD를 이용한 추천 시스템 구현

https://www.draw.io/

＃플로우 차트 그리기

http://blog.kthdaisy.com:8080/recommendation\_system\_kthdaisy/

＃협업 필터링 알고리즘과 SVD

conda install -c conda-forge scikit-surprise

conda install -c conda-forge/label/gcc7 scikit-surprise

conda install -c conda-forge/label/cf201901 scikit-surprise

＃surprise 모듈 설치

https://kutar37.tistory.com/37

＃파이썬 협업필터링 추천 알고리즘

https://datascienceschool.net/view-notebook/fcd3550f11ac4537acec8d18136f2066/

＃추천 시스템 알고리즘

http://mjdeeplearning.tistory.com/30

＃RMSE(평균 제곱근 오차), MAE(평균 절대 오차)

https://surprise.readthedocs.io/en/stable/index.html

＃surprise 라이브러리

## \*깃허브 히스토리

<https://github.com/CheonSeounghyun>

스크린샷이(가) 표시된 사진

자동 생성된 설명스크린샷, 컴퓨터, 노트북이(가) 표시된 사진

자동 생성된 설명스크린샷이(가) 표시된 사진

자동 생성된 설명