Journal of The Korea Society of Computer and Information
Vol. 23 No. 4, pp. 89-95, April 2018

www.ksci.re.kr
https://doi.org/10.9708/jksci.2018.23.04.089

Design and Implementation of a User-based Collaborative Filtering Application using Apache Mahout - based on MongoDB -

Junho Lee*, Kyungsoo Joo**

Abstract

It is not easy for the user to find the information that is appropriate for the user among the suddenly increasing information in recent years. One of the ways to help individuals make decisions in such a lot of information is the recommendation system. Although there are many recommendation methods for such recommendation systems, a representative method is collaborative filtering. In this paper, we design and implement the movie recommendation system on user-based collaborative filtering of apache mahout based on mongoDB. In addition, Pearson correlation coefficient is used as a method of measuring the similarity between users. We evaluate Precision and Recall using the MovieLens 100k dataset for performance evaluation.

▸Keyword: Mahout, User-based Collaborative Filtering, Movie Recommender System, MongoDB

I. Introduction

최근 인터넷의 급격한 발달로 인해 '정보의 바다'라고 불릴 정도로 넘쳐나는 정보가 생성되고 있다. 이러한 정보 속에서 사용자는 자신이 원하는 것을 선택하거나, 정보를 찾기 위한 의사결정에 많은 노력을 하고 있다. 하지만 너무 많은 양의 정보로 인해 혼자서는 의사결정을 하기 매우 어려운 실정이다. 따라서 사용자가 원하는 정보를 쉽게 추천해주는 추천 시스템(Recommendation System)이 등장하게 되었다. 추천 시스템이란 사용자들의 관심사 및 선호도를 기반으로 사용자 개개인에 알맞은 상품이나 서비스를 제공하는 방법이다.

이렇게 증가한 데이터로 인해 생겨난 분석 데이터는 기존의 관계형 데이터베이스로는 처리하기 어려운 크기이다[1]. 또한, 기존의 관계형 데이터베이스는 확장성이 좋지 않아 빅 데이터를 처리하기 위한 데이터베이스로 적당하지 않다[2, 3, 4].

그러한 문제점을 해결하기 위해 NoSQL(Not Only SQL)이 등장하였고, 이는 새로운 가능성을 제시한다[5]. NosSQL 데이터베이스 중 MongoDB는 분산 확장을 지원하여 빅 데이터 처리에 용이한 데이터베이스이다. 따라서 빅 데이터를 위한 시스템은 관계형 데이터베이스가 아닌 NoSQL 중 하나인 MongoDB 기반으로 구축되어야 바람직하다.

추천 시스템에 널리 사용되고 있는 알고리즘이 협업 필터링(Collaboration Filtering) 알고리즘이다. 협업 필터링 기법은 다른 사람들의 의견을 통하여 항목들을 추천하거나 예측하는 것이다[2].

한편, 추천 시스템에 대한 알고리즘 연구 및 성능 개선을 위한 연구는 많이 이루어지고 있다[3][4]. 하지만 실제 어플리케이션 시스템을 구현한 연구는 매우 드문 실정이고, 더욱이 빅 데이터 데이터베이스인 MongoDB를 활용한 협업 필터링 시스템을 구현한 국내 연구는 전무하다. 알고리즘의 연구 및 성능 향상을 위한 연구도 중요하지만 실제로 협업 필터링을 적용해 어플리케이션 시스템을 만들고, 그것을 효과적으로 개발하기

• First Author: Junho Lee, Corresponding Author: Kyungsoo Joo
*Junho Lee (wnsgh461@naver.com), Dept. of Computer Science, Soonchunhyang University
**Kyungsoo Joo (gsoojoo@naver.com), Dept. of Computer Software Engineering, Soonchunhyang University
• Received: 2018. 02. 19, Revised: 2018. 03. 22, Accepted: 2018. 04. 03.
• This research was supported by the Soonchunhyang University Research Fund.

위한 연구도 필수로 수행해야 할 작업 중 하나이다.

본 연구에서는 많은 문화 콘텐츠 중 영화를 사용자에게 쉽게 추천하기 위한 영화 추천 시스템을 사용자 기반의 협업 필터링과 MVC(Model-View-Controller)패턴, NoSQL 데이터베이스 중 MongoDB를 사용하여 설계 및 구현하였다. 데이터 셋은 미네소타 대학의 GroupLens Research Project에서 수집된 MovieLens 100k 데이터 셋을 사용하였고, 각 이웃 간의 유사도를 구하는 방법으로는 피어슨 상관계수(Pearson Correlation Coefficient)를 채택하였다.

본 연구는 Mahout에서 제공하는 기본적인 협업 필터링 알고리즘 중 사용자 기반의 협업 필터링 알고리즘을 사용하여 영화 추천 시스템을 구현하였다. 개발 및 실행 환경은 Windows7 64bit 플랫폼에서 Java, JSP를 사용하여 구현하였고, 데이터베이스는 MongoDB 3.2.1을 웹 어플리케이션 서버는 Apache Tomcat8.5을 사용하였다.

본 연구의 구성은 다음과 같다. 2장에서는 구현한 영화 추천 시스템의 주요 기술인 Mahout 협업 필터링, MVC 패턴, MongoDB의 관련 연구를 설명하고, 3장에서는 영화 추천 시스템의 설계, 4장에서는 MovieLens 데이터 셋을 이용해 구현한 영화 추천 시스템에 대한 성능평가를 진행하고, 5장에서는 4장의 평가 결과와 본 연구에서 구현한 영화 추천 시스템에 대한 결론을 기술한다.

II. Related works

2.1 Collaborative Filtering

협업 필터링 기법은 목표 사용자와 유사한 구매이력을 보이는 이웃 사용자들이 보여준 아이템에 대한 선호도를 바탕으로 목표 사용자에게 유용한 아이템을 추천하는 방법이다. 즉, 협업 필터링 기법은 그룹이나 사람들의 취향 사이에는 일반적인 트렌드와 패턴이 있다는 가정에서 출발하는 것이다[5]. 협업 필터링의 대표적인 추천 방법 중 하나인 사용자 기반의 협업 필터링의 실행 과정은 Fig.1과 같다.

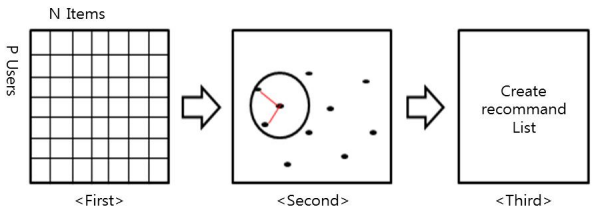

Fig. 1. Process of user-based collaborative filtering

사용자들의 평가치를 이용하여 사용자들 간의 유사도를 계산한 후 k-NH를 구성한다. 유사도를 계산하기 위해서는 일반적으로 피어슨 상관 계수가 사용된다.

피어슨 상관 계수 $w(A,B)$는 두 사용자 A, B에 의해 공통적으로 평가된 아이템들의 평가치를 이용하여 식(1)과 같이 계산한다[6].

$$w(A,B)=\frac{\sum_{i=1}^{q}(R_{A,i}-\overline{R_A})(R_{B,i}-\overline{R_B})}{\sqrt{\sum_{i=1}^{q}(R_{A,i}-\overline{R_A})^2\sum_{i=1}^{q}(R_{B,i}-\overline{R_B})^2}} \quad (1)$$

여기서 $R_{A,i}$와 $R_{B,i}$는 사용자 A와 B가 공통으로 평가한 아이템 i의 평가치를 뜻한다. 그리고 $\overline{R_A}$, $\overline{R_B}$는 사용자 A와 B의 이용 가능한 평가치 들의 평균값을 뜻한다.

협업 필터링 시스템에서 유사도를 평가하는 방법 중 대표적인 방법은 k-최 근접 이웃 모델이다. 해당 모델은 추천받을 사용자와 다른 사용자들의 거리를 계산하여 가까운 이웃 k명을 최 근접 이웃으로 선정하는 방법을 말한다.

협업 필터링에는 여러 가지 종류가 존재하는데 대표적으로 아이템 기반의 협업 필터링[7, 8], 사용자 기반의 협업 필터링[9, 10], 메모리 기반의 협업 필터링[11] 등이 있다. 이러한 연구 외에도 또 다른 분야의 다양한 연구들이 존재하는데 Ding[12]은 협업 필터링 방식에 Time Weight를 추가하는 연구를 진행하였다.

2.2 Mahout

Mahout의 라이브러리는 실질적으로 세 가지 영역에 집중하고 있다. 그 세 가지 영역은 추천엔진, 군집, 분류이다. 그 중 추천 엔진은 아이템 기반 협업 필터링 추천기법, 사용자 기반 협업 필터링 추천 기법이 많이 사용되고 있다[13, 14].

Table 1. Algorithm of Item-based Collaborative Filtering of Mahout

① for every item **i** that **u** has no preference for yet ② for every item **j** that **u** has a preference for ③ compute a similarity s between **i** and **j** ④ add **u**'s preference for **j**, weighted by s, to a running average ⑤ return the top items, ranked by weighted average
* u : Target User * i : All items that the target user has not rated * j : All items rated by the target user * s : Similarity between i and j

Table 2. Algorithm of User-based Collaborative Filtering of Mahout

① for every item **i** that **u** has no preference for yet ② for every other user **v** that has a preference for **i** ③ compute a similarity **s** between u and **v** ④ incorporate **v**'s preference for **i**, weighted by **s**, into a running average ⑤ return the top items, ranked by weighted average
* u : Target User * i : All items that the target user has not rated * v : All the people who evaluated i * s : Similarity between u and v

위의 Table 1, Table 2는 협업 필터링의 대표적인 두 가지 알고리즘을 나타낸 것으로 각각 아이템 기반 협업 필터링, 사용자 기반 협업 필터링의 알고리즘 이다. 이러한 알고리즘을 사용하여 Fig.2와 같은 기본적인 Mahout 추천기의 컴포넌트 관계도를 얻을 수 있다[13].

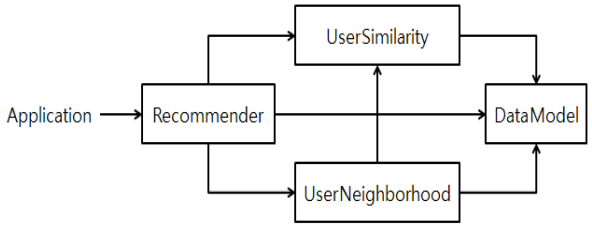

Fig. 2. Simplified component relationships of user-based recommender

위의 Fig. 2.의 컴포넌트들의 역할을 다음과 같이 정의할 수 있다[13].

DataModel : 데이터 연산을 위해 모던 선호, 사용자, 아이템 정보를 저장하거나 접근하도록 한다.

UserSimilarity : 하나 혹은 여러 개의 가능한 측정 혹은 연산을 통해 얼마나 두 사용자가 유사한지 구한다.

UserNeighborhood : 주어진 사용자와 가장 유사한 사용자 그룹을 알려준다.

Recommender : 모든 컴포넌트를 활용하여 사용자에게 아이템을 추천한다.

Mahout은 이러한 단순화된 추천기를 기반으로 다양한 추천기로 확장 가능한 추천 엔진을 제공한다[13, 14].

2.3 MVC Pattern

MVC(Model-View-Controller) 패턴은 널리 알려진 아키텍처 패턴 중 하나로, 웹 어플리케이션 개발에 많이 사용되고 있다. 이러한 MVC 패턴의 가장 큰 특징은 Fig.3과 같은 구조라고 볼 수 있다.

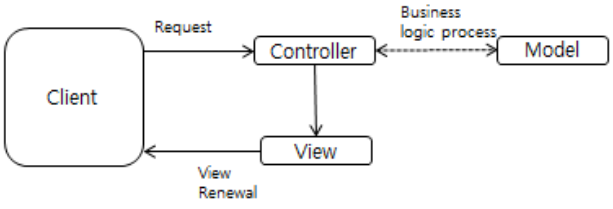

Fig. 3. MVC Pattern Architecture

위의 Fig.3과 같은 구조를 MVC 패턴 중에서도 Model 2의 구조라고 한다. 이런 구조는 모델, 뷰, 컨트롤러가 각각의 역할을 나눠 작업한다는 특징을 가지고 있기 때문에, 하나의 기능을 수행하기 위해 어플리케이션이 여러 계층으로 나누어 구현된다[15].

2.4 MongoDB

MongoDB는 문서 지향형(Document Oriented) 데이터베이스이다. 관계형 모델을 사용하지 않고 문서 모델(Document Model)을 채택하는 이유는 분산 확장을 쉽게 하기 위한 것이지만, 다른 이점도 가지고 있다. 내장문서와 배열을 허용함으로써 문서 지향 모델은 복잡한 계층 관계를 하나의 레코드로 표현할 수 있다. 또한 MongoDB에서는 문서의 키와 값을 미리 정의하지 않는다. 따라서 고정된 형태의 스키마가 존재하지 않는다. 고정된 스키마가 존재하지 않기 때문에 필요에 따라 필드를 추가하고 삭제하는 것이 쉬워졌고, 개발과정 또한 빠르게 이루어질 수 있다[16].

또한, MongoDB는 범용 데이터베이스 목적으로 만들어져 데이터의 생성, 읽기, 변경, 삭제 외에도 특별한 기능을 제공한다. 그러나 MongoDB에서는 관계형 데이터베이스에서 일반적으로 제공하는 기능 중 몇몇 기능이 없는데 그 중 대표적인 기능은 조인과 다중 트랜잭션이다. 이런 기능들은 분산 시스템에서 효율적으로 제공하기 어렵기 때문에 제외되었고, 높은 확장성을 제공하는 아키텍처를 위한 결정이다[16].

III. Design of Movie Recommendation System

일반적인 협업 필터링은 사용자에게 무엇인가를 추천하기 위해 이전에 다른 사용자가 평가하여 점수를 남긴 아이템을 기반으로 추천을 하게 된다. 이러한 협업 필터링을 추천 시스템의 System Architecture는 아래의 Fig.4와 같다.

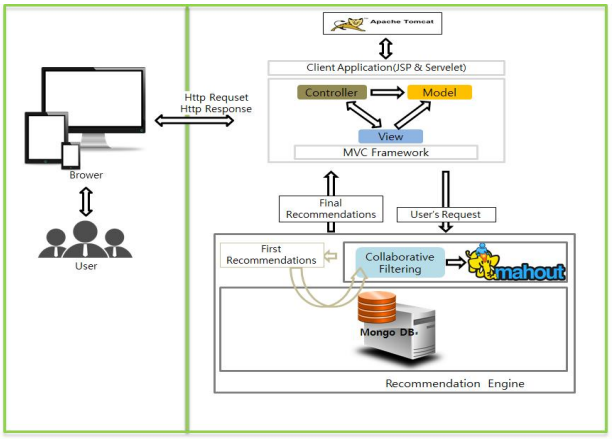

Fig. 4. System Architecture

본 연구에서 구현한 영화 추천 시스템은 Fig.4 System Architecture와 같이 추천 엔진과 GUI를 사용하여 사용자들이 시스템을 쉽게 사용할 수 있게 하는 인터페이스를 제공한다.

인터페이스는 MVC 패턴을 기반으로 구현되었고, 이는 웹을

통해 사용자가 쉽게 본 시스템을 이용 할 수 있는데 큰 역할을 할 수 있으며, 온라인을 통해 실시간으로 사용자가 영화를 추천받을 수 있다는 것을 의미한다.

본 연구의 핵심이라고 할 수 있는, 실제로 추천 처리가 이루어지는 추천엔진에서는 MongoDB 데이터베이스에 저장된 데이터를 가지고 다음과 같은 협업 필터링 알고리즘을 통해 사용자에게 영화를 추천하게 된다.

첫째, 해당 데이터 셋에 미리 기록되어있는 여러 사용자들의 세 가지 기준정보를 토대로 사용자-영화 매트릭스를 생성한다.

두 번째, 해당 매트릭스를 기반으로 사용자의 유사도를 측정하고, 유사도를 기반으로 최 근접 이웃 k명을 선정한다.

마지막으로, 최 근접 이웃이 평가한 영화 제목, 영화 평점으로 사용자에게 추천하게 될 영화 예상 목록을 생성한다.

협업 필터링 알고리즘을 사용해 사용자에게 추천을 하기 위해서는 사용자의 이전의 구매기록이나 다른 사용자가 이전에 평가하여 점수를 남긴 아이템을 기반으로 평가하며, 본 연구에서는 미네소타 대학의 MovieLens 100k 데이터 셋을 사용하였다.

3.1 Class Diagram

본 연구에서 구현한 시스템은 MVC 패턴을 적용하여, 다음과 같은 클래스 다이어그램을 얻었다.

Fig.5의 클래스 다이어그램은 각각의 스테레오 타입에 따라 다음과 같은 조건으로 구현할 수 있다[17].

1. <<entity>> 타입을 사용한 클래스는 Model의 역할을 하도록 구현한다.
2. <<boundary>> 타입을 사용한 클래스는 View로서 JSP 등으로 구현한다.
3. <<control>> 타입을 사용한 클래스는 Controller로서 서블릿 등으로 구현한다.

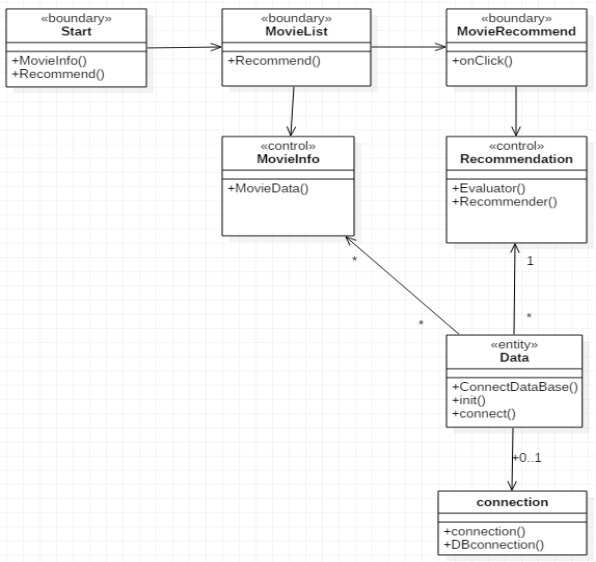

Fig. 5. Class Diagram

3.2 DataBase Design

본 연구에서 사용하는 MongoDB 데이터베이스 설계는, 관계형 데이터베이스 설계를 위한 정보공학 방법론을 확장한, '빅데이터 응용을 위한 MongoDB 기반의 데이터베이스 설계방법론'을 적용하여, Fig.5의 클래스 다이어그램 중 Model을 담당하는 부분에 해당하는 MongoDB 데이터베이스를 설계한다[18]. 이에 따라 MongoDB 데이터베이스를 위한 E-R Diagram이 Fig.6과 같이 도출된다.

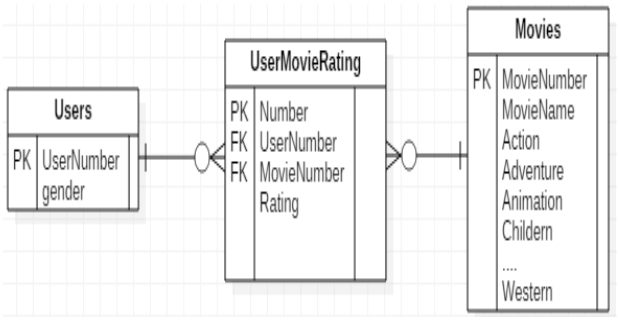

Fig. 6. MongoDB Database E-R Diagram

IV. Implementation and Evaluation

본 연구에서 구현한 영화 추천 시스템은 Java, JSP(Java Server Pages), MongoDB 데이터베이스 및 Mahout 라이브러리를 사용하여 구현하였다. 이번 절에서는 구현한 영화 추천 시스템에 데이터 셋을 적용하여 성능을 평가하는 실험을 한다. 평가는 정확도(Precision)와 재현율(Recall), F1을 기준으로 수행한다.

4.1 Implementation

본 연구에서 구현한 시스템은 위 Fig.4의 System Architecture와 같고, 사용자의 접근 편의성을 높이기 위해 웹 기반의 응용 시스템으로 개발하였다. 따라서 사용자가 웹 브라우저를 통해 쉽게 접근이 가능하다.

Table 3. Development environment

Language	Java, JSP
Platform	Windows7 64bit
Database	MongoDB v3.2.1

본 연구에서는 Table 3과 같은 개발 환경을 기반으로 Apache Mahout 0.9 라이브러리를 사용하여 구현하였고, 구현된 영화 추천 시스템은 Fig.7 과 같다.

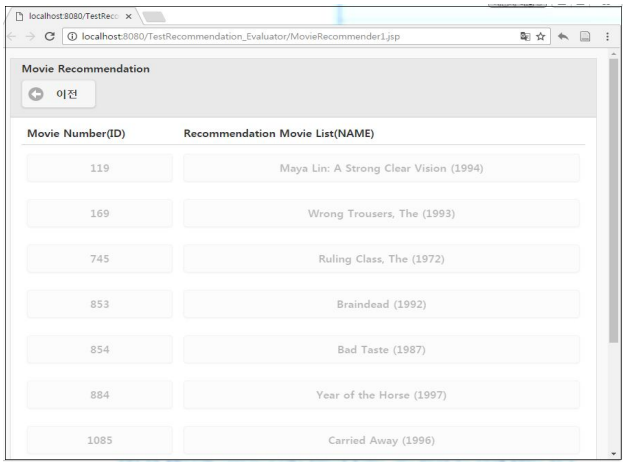

Fig. 7. Implemented web-based movie recommendation system

본 연구의 MongoDB 데이터베이스는 Fig.6의 E-R Diagram을 토대로 '빅 데이터 응용을 위한 MongoDB 기반의 데이터베이스 설계방법론'을 이용하여 확정하였다[18]. MongoDB 데이터베이스에는 사용자, 사용자별 영화감상 평점, 영화정보 세 가지 컬렉션으로 크게 구분 할 수 있고, 각각은 Table 4, Table 5, Table 6과 같이 MongoDB 컬렉션이 확정되었다.

Table 4. 'Users' Collection

```
Users :
{
    UserNumber : usernumber ,
    gender : gender
}
```

Table 5. 'UserMovieRating' Collection

```
UserMovieRating :
{
    Number : number ,
    UserNumber : getUserNumber ,
    MovieNumber : getMovieNumber ,
    Rating : Rating
}
```

Table 6. 'Movies' Collection

```
Movies :
{
    MovieNumber : getMovieNumber ,
    MovieName : getMovieName ,
    Action : 0 or 1 ,
    Adventure : 0 or 1 ,
    Animation : 0 or 1 ,
    ...
    Sci-Fi : 0 or 1 ,
    Thriller : 0 or 1 ,
    War : 0 or 1 ,
    Western : 0 or 1 ,
}
```

4.2 Evaluation

본 연구에서 사용한 미네소타 대학의 MovieLens 100k 데이터 셋은 사용자의 수가 1000명, 영화의 수는 1700개 이고, 선호도 평가 데이터 수가 약 100,000개 이다.

k-최 근접 이웃 기법으로 생성된 추천목록의 평가엔 정확도와 재현율 두 가지가 사용된다.

여기서 재현율은 특정 사용자(Target User)가 실제로 경험하여 평가한 아이템들 중 구현한 시스템이 생성한 추천목록에 속하게 된 아이템의 비율로 다음 식(2)와 같이 계산 한다[19].

$$Recall = \frac{|T \cap R|}{|T|} \quad (2)$$

정확도는 구현한 시스템이 생성한 추천목록의 아이템들 중 특정 사용자가 실제로 경험하여 평가한 아이템의 비율로 다음 식(3)과 같이 계산 한다[17].

$$\mathrm{Precision} = \frac{|T \cap R|}{|R|} \quad (3)$$

일반적으로 정확도와 재현율 두 가지는 추천목록의 개수에 따라 상충관계를 가지게 된다. 이러한 상충관계 때문에 두 가지를 동시에 고려하는 F1이 식(4)와 같이 계산되어 사용된다[19, 20].

$$F1 = \frac{2 \times Recall \times \mathrm{Precision}}{Recall + \mathrm{Precision}} \quad (4)$$

해당 식에 의거하여 최 근접 이웃에 수에 따라 본 연구에서 구현한 영화 추천 시스템의 결과를 측정하였다.

Table 7. Depending on the number of nearest neighbors Recall / Precision value

	number of nearest neighbors				
	50	100	150	200	250
Recall	0.29	0.30	0.28	0.27	0.27
Precision	0.15	0.17	0.16	0.15	0.15

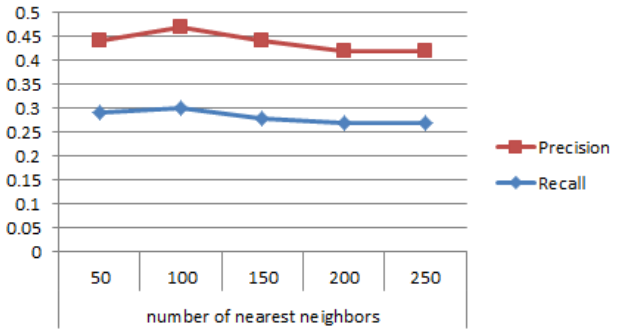

Fig. 8. Depending on the number of nearest neighbors Recall / Precision value

본 연구에서 구현한 영화 추천 시스템을 미네소타 대학의 MovieLens 100k데이터 셋을 이용한 측정결과로 Table 7, Fig.8과 같은 정확도와 재현율 값을 얻을 수 있었다. 해당 측정에 사용한 데이터 셋은 선호도 평가 100,000개와 사용자 1000명을 기준으로 측정 하였고, 최 근접 이웃에 따른 결과의 정확도, 재현율 확인을 위해 최 근접 이웃의 수를 기준으로 측정을 진행하였다.

최 근접 이웃을 50명, 100명, 150명, 200명, 250명 단위로 측정하였고, 측정결과 최 근접 이웃이 100명일 때, 가장 좋은 정확도와 재현율을 보인 것을 확인할 수 있었다.

아래의 Table 8은 측정한 정확도와 재현율을 기반으로 두 가지의 상충관계를 고려한 식(4) F1의 결과이다.

Table 8. Depending on the number of nearest neighbors F1 value

	number of nearest neighbors				
	50	100	150	200	250
F1	0.20	0.22	0.20	0.19	0.19

Table.8 F1의 결과역시 Table 7, Fig.8의 결과와 동일하게 최 근접 이웃이 100명 일 때 가장 좋은 성능을 보였다. 따라서 본 연구에서 사용한 데이터 셋에 가장 적합한 최 근접 이웃의 수는 100명이라고 할 수 있다.

V. Conclusions

본 연구에서는 빅 데이터기반 분석 알고리즘 중 하나인 협업 필터링을 활용해 많은 문화 콘텐츠 중 영화를 추천 할 수 있는 MongoDB 기반의 웹 응용 시스템을 구현하였다. 이를 구현하기 위해 Mahout의 협업 필터링 라이브러리를 사용하였다. 그로인해 사용자 간의 관계에 따라 서로 다른 사용자들로부터 유사도를 측정하고, 다른 사용자들의 선호도 평가치를 바탕으로 사용자에게 알맞은 영화를 추천 할 수 있었다. 또한, 빅 데이터를 위한 시스템 구축을 위해 기존의 관계형 데이터베이스 대신 확장성이 좋은 NoSQL 데이터베이스 중 MongoDB를 사용하여, 기존의 관계형 데이터베이스 기반의 추천시스템의 한계점인 확장성을 개선하였다.

결과측정 시 최 근접 이웃은 100명으로 설정한 뒤의 Recall, Precision, F1값들이 각각 0.29, 0.16, 0.21로 나쁘지 않은 성능을 보였다.

REFERENCES

[1] Jacobs, A., “The pathologies of big data,” Communications of the ACM, 52(8), pp. 36-44, 2009.

[2] Xiaoyuan Su and Taghi M. Khoshgoftaar, “A Survey of Collaborative Filtering Techniques,” Advances in Artificial Intelligence, pp. 1-19, 2009.

[3] Mingun Kim, and Kyoung-Jae Kim, "Recommender Systems using Structural Hole and Collaborative Filtering." Journal of Intelligence and Information Systems 20.4, pp. 107-120, 2014.

[4] Karydi, Efthalia, and Konstantinos Margaritis. "Parallel and distributed collaborative filtering: A survey." ACM Computing Surveys(CSUR), 49.2, 37, 2016.

[5] Su-Mi Shin, Kyung-Chang Kim, “Addressing the New User Problem of Recommender Systems Based on Word Embedding Learning and Skip-gram Modelling,” Journal of The Korea Society of Computer and Information, Vol. 21, No. 7, pp. 9-16, 2016.

[6] Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J., "GroupLens: an open architecture for collaborative filtering of netnews," In Proceedings of the 1994 ACM conference on Computer supported cooperative work, pp. 175-186, October, 1994.

[7] Sarwar, B., Karypis, G., Konstan, J., and Riedl, J., "Item-based collaborative filtering recommendation algorithms," In Proceedings of the 10th international conference on World Wide Web, ACM, pp. 285-295, April, 2001.

[8] Jun-Ho Lee, and Kyung-Soo Joo., "Implementation and Comparison of Moive Recommendation System using Item-Based Collaborative Filtering.", Soonchunhyang J. Instit. Technol.23(1), pp. 013-020, 2017.

[9] Jun-Ho Lee, and Kyung-Soo Joo., "Design and Implementation of Collaborative Filtering Application System using Apache Mahout." Journal of The Korea Society of Computer and Information, Vol.22, No.7, pp. 125-131, 2017.

[10] ZHAO, Zhi-Dan, SHANG, Ming-Sheng. "User-based collaborative-filtering recommendation algorithms on hadoop," In: Knowledge Discovery and Data Mining, 2010. WKDD'10. Third International Conference on. IEEE, pp. 478-481, 2010.

[11] Yu, K., Schwaighofer, A., Tresp, V., Xu, X., & Kriegel, H. P., "Probabilistic memory-based collaborative filtering," IEEE Transactions on Knowledge and Data Engineering, pp. 56-69, 2004.

[12] Y. Ding, “Time weight collaborative filtering,” Proceedings of the 14th ACM international conference, pp.485-492, 2005.

[13] S. Owen, R. Anil, T. Dunning and E. Friedman, "Mahout in Action" ManningPublications, 2014.

[14] The Apache Foundation, https://github.com /apache/mahout/blob/trunk/mrlegacy/src/main/java/org/apache/mahout/cf/taste/impl/recommender/

[15] Freeman Eric, Freeman Elisabeth, Sierra Kathy and Bates Bert, "Head First Design Patterns" Oreilly & Associates Inc, pp. 529-558, 2004.

[16] Michael Dirolf, "MongoDB The Definitive Guide." O'ReillyMedia, 2013.
[17] Jun-Ho Lee, Jung-Woong Woo, Cheong-An Lee and Kyung-Soo Joo, "A Unified Object-Oriented Analysis and Design Methodology for Secure Web ," The 2nd International Conference on Interdisciplinary research on Computer science, Psychology, and Education(ICICPE), pp. 22-24, 2018.
[18] Jun-Ho Lee, and Kyung-Soo Joo., "Development of the design methodology for large-scale database based on MongoDB." Journal of The Korea Society of Computer and Information, Vol.22, No.11 pp. 57-63, 2017.
[19] Sarwar, B., Karypis, G., Konstan, J., and Riedl, J., “Application of dimensionality reduction in recommender system-a case study,” Minnesota Univ Minneapolis Dept of Computer Science, No. TR-00-043, 2000.
[20] Yang, Y. and Liu, X., "A re-examination of text categorization methods," In Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, ACM, pp. 42-49, August, 1999.

Authors

Junho Lee received the B.S. degrees in Computer Software Engineering from Soonchunhyang University, Korea, in 2015 respectively Lee joined the faculty of the Department of Computer Software Engineering at Soonchunhyang University, Asan, Korea, in 2015. He is currently a M. S. course in the Department of Computer Science, Soonchunhyang University. He is interested in database and big data database.

Kyungsoo Joo received the Ph.D. degrees in Computer Science from Korea University, Korea, in 1993 respectively Dr. Joo joined the faculty of the Department of Computer Science at Korea University, Seoul, Korea, in 1993. He is currently a Professor in the Department of Computer Software Engineering, Soonchunhyang University. He is interested in database and big data database.