ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 4, April 2017

Book Recommendation System Using Apache Spark

Ishwari Kulkarni, Pritesh Gandhi, Pradnya Karlekar

B.E. Student, Dept. of C.S., P.E.S Modern College of Engineering, Savitribai Phule Pune University, Pune, India

ABSTRACT: There are many online shopping websites such as, Amazon, Filpcart, Snapdeal and others. Number of online buyers and traders are increasing day by day. Therefore effective business techniques are needed to handle the large amount of data generated daily. Recommendation systems filter the data and provide adequate information to the users. Therefore recommendation systems are very useful in online shopping websites, e-commerce so that user can find the desired item form many items on website. Here we have proposed recommendation system for books to improve user experience on online book store and give user more accurate recommendation. The aim of the paper is to provide faster and efficient recommendations. We are using Apache Spark to build the recommendation system. Apache spark have ALS library for matrix factorization method which is collaborative filtering method. Along with this we are doing popularity estimation of every book and consider it while recommendation. Popularity estimation is used to solve cold start problem in recommendation system.

KEYWORDS: Collaborative filtering, apache spark, ALS, popularity estimation, Matrix factorization model

I. INTRODUCTION

Many online books stores use recommendation systems to recommend the books to users. This helps user to find the book they might be interested in. This system is a win-win situation. That means it is beneficial for the user because it becomes easy to find the desired the book and it is also beneficial for the online book store company as they are attracting users by providing easy interaction. There are many approaches for building a recommendation system. They are collaborative filtering, content based filtering and hybrid. In collaborative filtering user's rating history is used for predicting the items he/she may be interested in. Content-based filtering methods are based on a description of the item and a profile of the user's preference. Hybrid approach combines collaborative filtering and content based filtering approach.

We are using apache spark for collaborative filtering. spark.mllib currently supports model-based collaborative filtering, in which users and products are described by a small set of latent factors that can be used to predict missing entries. spark.mllib uses the alternating least squares (ALS) algorithm to learn these latent factors. Popularity estimation is done by calculation number of rating given to particular user. In the end we combine the results from collaborative filtering and popularity estimation and then display the book recommendations. New user has not rated any books so it is difficult to rate the books using collaborative filtering as this method requires rating history of the user. This problem is called cold start problem.

II. RELATED WORK

In "A Distributed Hybrid Recommendation Framework to Address the New-User Cold-Start Problem" paper user classification and applying different algorithm for different user is given.

In "Collaborative Filtering Recommendation Algorithm based on Hadoop and Spark" paper comparison between collaborative algorithm executed on hadoop and spark is given. It proves that Apache spark is better for the recommendation system than apache hadoop. In-memory based RDD paradigm allows for more efficient work allocation between nodes, reducing costly read/write operations of intermediate results on hard drives.

In "A Hybrid Distributed Collaborative Filtering Recommender Engine Using Apache Spark" paper collaborative filtering using apache spark is explained.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 4, April 2017

III. PROPOSED ALGORITHM

A. *Design Considerations:*

Here collaborative filtering and popularity estimation are two major components of recommendation system. Result of these modules is combined and final result is generated. The overall architecture of the proposed approach is illustrated in Fig. 1

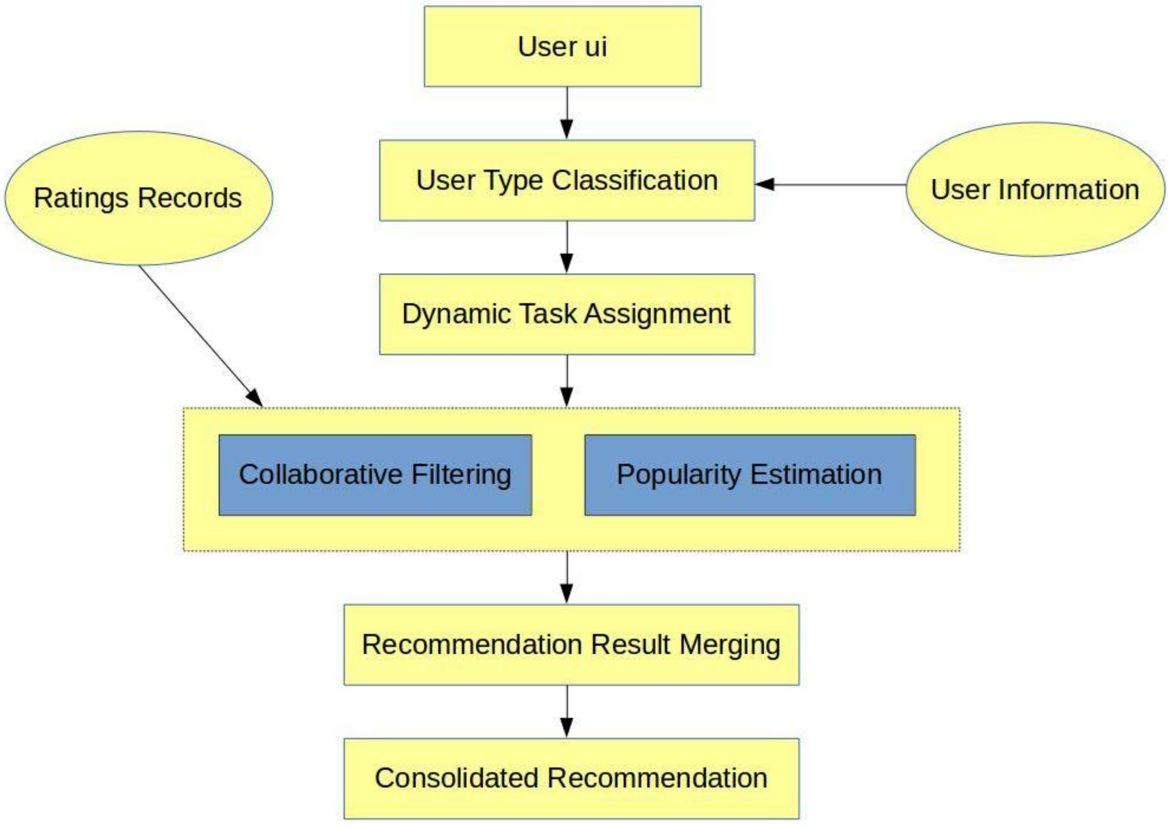

Fig.1. Architecture diagram of Book Recommendation System

As shown in architecture diagram first user type is identified. We classified users in two types, new user and ordinary user. If user has not rated any books then it is new user otherwise it is ordinary user. If user is new user then we apply only popularity estimation. Because collaborative filtering is not suitable of new user as it requires rating history. If user is ordinary user then collaborative filtering and popularity estimation both are used. After that result from both algorithm is merged.

B. *Collaborative filtering:*

ALS is Matrix Factorization Algorithm. Matrix Factorization decomposes a large matrix into products of matrices.
R = U * V

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 4, April 2017

For example in recommendation systems, let us consider R as a matrix of User (Rows) and Ratings (Columns). Matrix factorization will allow us to discover the latent features that define the interactions between User and Ratings. In other words, ALS uncovers the latent features.
R is m*n matrix
U is m*r matrix - User and Feature matrix
V is r*n matrix - Item and Feature Matrix
So the model will allow U and V giving a low dimension feature space. Also, space efficient for large data sets.
Following is how it works:
1. M is initialized with the average item ratings as its first row and random numbers for the rest row.
2. Alternate the calculation of the Cost function until we met the criteria:

- Fix V and solve U by minimization of the cost function J(R, U, V) by resolving the least square error function
- Now, Fix U and solve V by minimization of the cost function J(R, U, V) by resolving the least square error function[4]

Spark MLlib implements a collaborative filtering algorithm called Alternating Least Squares (ALS), which has been implemented in many machine learning libraries and widely studied and used in both academia and industry. ALS models the rating matrix (R) as the multiplication of low-rank user (U) and product (V) factors, and learns these factors by minimizing the reconstruction error of the observed ratings. The unknown ratings can subsequently be computed by multiplying these factors. In this way, companies can recommend products based on the predicted ratings and increase sales and customer satisfaction.

ALS is an iterative algorithm. In each iteration, the algorithm alternatively fixes one factor matrix and solves for the other, and this process continues until it converges. MLlib features a blocked implementation of the ALS algorithm that leverages Spark's efficient support for distributed, iterative computation.[7]

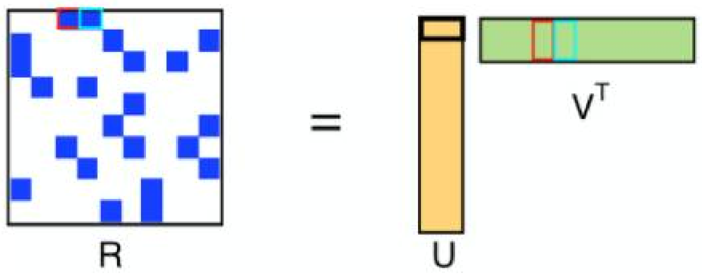

Fig.2. Matrix Factorization method

C. *Popularity Estimation*

The third type of recommendation algorithm is based on popularity. In general, items that most people are interested are usually a safe choice when you don't know her particular preferences. In the case of new user, popularity becomes one possible solution to the cold start problem. Popularity of the bookcan be defined as the number of users that have rated that book. [1]

D. *User type Classification*

We classify the user as new user and ordinary user. If user is new user then popularity estimation algorithm is applied as collaborative filtering is not suitable. If user is ordinary user then both popularity estimation and collaborative is applied. If user has not rated any book then it is considered as new user otherwise it is ordinary user.

IV. Pseudo code

```
Step 1: Identifythe user as new or ordinary.
        If new user goto step 4
Step 2: Load training and test data into (user, book, rating) tuples
        def parseRating(line):
            fields = line.split()
            return (int(fields[0]), int(fields[1]), float(fields[2]))
        training = sc.textFile("…").map(parseRating).cache()
        test = sc.textfile("…").map(parseRating)

Step 3:  Training recommendation model.
        model= ALS.train(training, rank=10, iteration=5)
Step 4:  make predictions on (user, book) pairs from the test data
        predictions = model.predictAll(test.map(lambda x: (x[0], x[1])))
Step 5:  Calculate the popularity of books.
Step 6: Display the recommended books.
Step 7: End.
```

V. SIMULATION RESULTS

For implementing this system we have used flask framework which is written in python. We created web app using this framework. Web pages are written in HTML. Front end i.e. web pages are connected to the dataset using python. In front end we have created online book store website. User can login, logout, rate the book, search the book.

We have used book-crossing dataset from site http://www2.informatik.uni-freiburg.de/~cziegler/BX/ . This dataset contains 278,858 users providing 1,149,780 ratings about 271,379 books. This dataset contains 3 tables BX-Users, BX-Books, BX-Book-Ratings. BX-User contains user id, age, location. BX-Books contains book id, book title, book author, year of publication, publisher, image link. BX-Book-Ratings contains user, book id, rating expressed on scale from 1-10.

Apache spark is fast and general engine for large scale data processing. We have implemented recommendation algorithms i.e. collaborative filtering and popularity estimation using apache spark framework. MLlib is apache spark's scalable machine learning library. Spark brings data into RAM and then process it unlike hadoop where most time goes in writing data from disk to memory.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 4, April 2017

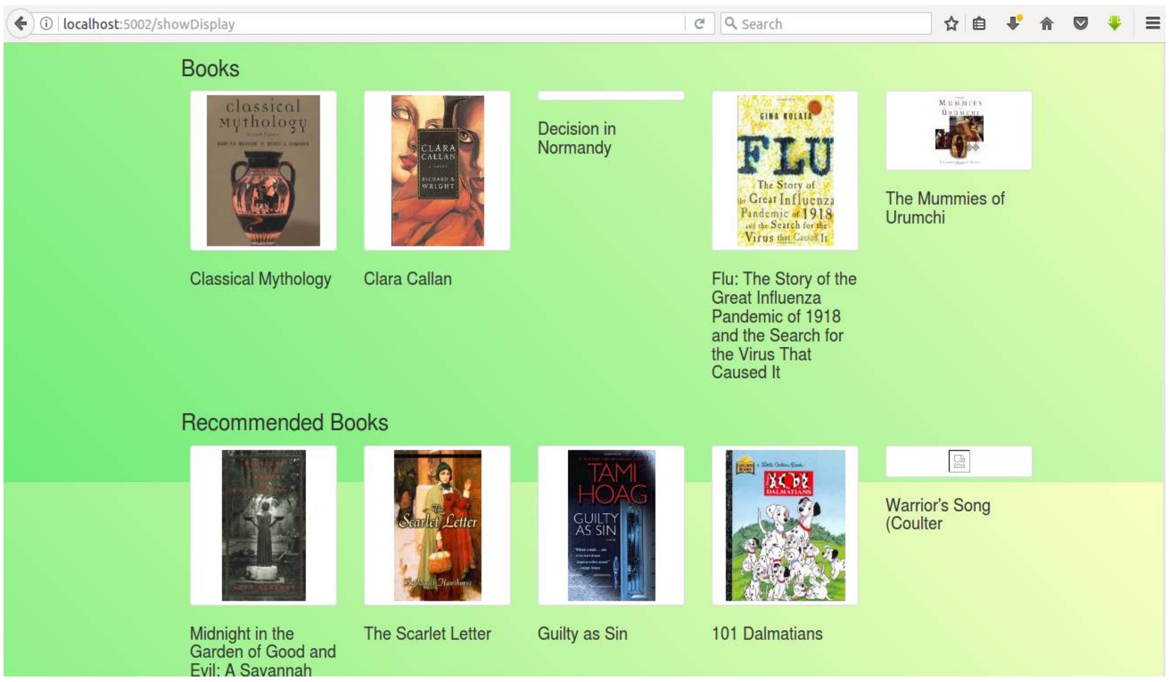
localhost:5002/showDisplay
Search
Books
Classical Mythology
Clara Callan
Decision in Normandy
Flu: The Story of the Great Influenza Pandemic of 1918 and the Search for the Virus That Caused It
The Mummies of Urumchi
Recommended Books
Midnight in the Garden of Good and Evil: A Savannah
The Scarlet Letter
Guilty as Sin
101 Dalmatians
Warrior's Song (Coulter

Fig.3.Result of recommended books

VI. Conclusion and Future Work

The experimental findings show that running time of the algorithm is improved with Spark. However few challenges we faced is Spark demands a higher RAM size for in memory computation which is expensive. This paper usually solves the cold start problem and increases the recommendation results. In-memory based RDD paradigm allows for more efficient work allocation between nodes,

reducing costly read/write operations of intermediate results on hard drives Much better scalability can be achieved if data is in the size of Terabytes. Thus using spark framework it gives more accurate results for the large data sets. But there is one limitation for apache spark that is memory requirement. For future scope we can add content based filtering. We can add Multilanguage facility. Demographic content can also be used for recommending new users.

References

1. Jenq-Hour Wang, Yi-Hao Chen, ”A Distributed Hybrid Recommendation Framework to Address the New-User Cold-Start Problem”, IEEE conference ,2015
2. Poonam Ghuli, Antanu Ghosh, Dr. Rajashree Shettar,”A Collaborative Filtering Recommendation Engine in a Distributed Environment”, International Conference on Contemporary Computing and Informatics,IEEE conference ,2014
3. Manisha Chandak Sheetal Girse Debajyoti Mukhopadhyay,”Introducing Hybrid Technique For Optimization of Book Recommender System”,International Conference on Advanced Computing Technologies and Applications,Elsevierconference , 2015.
4. Baetosz Kupisz, Olgierd Unold, ”Collaborative Filtering Recommendation Algorithm based on Hadoop and Spark”, IEEE conference , 2015.
5. Chunzi Wang, Zhou zheng,Zhuang Yang,”The Research of Recommendation System Based on Hadoop Cloud Platform”, IEEE conference , 2014.
6. Michael Armbrust, Reynold S. Xin, Cheng Lian, Yin Huai, Davies Liu, Joseph K. Bradley,Xiangrui Meng, Tomer Kaftan, Michael J. Franklin, Ali Ghodsi, Matei Zaharia, “Spark SQL: Relational Data Processing in Spark”, ACM, 2015.
7. Sasmita Panigrahia, Rakesh Ku. Lenkaa, Ananya Stitipragyana, “A Hybrid Distributed Collaborative Filtering Recommender Engine Using Apache Spark”, Elsevier, 2016.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 4, April 2017

8. Archit Verma1, Dharmendra Kumar2, “Evaluating and Enhancing Efficiency of Recommendation System using Big Data Analytics”, IRJET, 2017.
9. B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, “Item-based collaborativefiltering recommendation algorithms,” in Proceedings of the 10th international conference on World Wide Web. ACM, 2001, pp. 285– 295.
10. Z.-D. Zhao and M.-S. Shang, “User-based collaborative-filtering recommendation algorithms on hadoop,” in Knowledge Discovery and Data Mining, 2010. WKDD’10. Third International Conference on. IEEE, 2010, pp. 478–481.
11. Mr. Rajesh H. Davda1, Mr. Noor Mohammed, “ Text Detection, Removal and Region Filling Using Image Inpainting”, International Journal of Futuristic Science Engineering and Technology, vol. 1 Issue 2, ISSN 2320 – 4486, 2013