Abstract—Recommendation system has been used by enormous users. It used commonly in recent years, and are used in a variable areas in many popular applications which comprise of movies, songs, bulletin, files, research courses, online shopping, social networking sites, and products recommendation. Gradually, the extent of consumers, products and facts has mature rapidly, the big data scrutiny problem for examination of recommender systems. Here we are evaluating most efficient recommendation system. Conventional recommender systems frequently suffer from deficiency of scalability, efficiency and real time recommendation problems while processing or analysing documents taking place at huge scale. To get rid from these problems, a recommendation system is implemented in Apache Hadoop and Spark including MapReduce paradigm for Bigdata. Proposed Framework can have considerable enhancement in performance associated to traditional tools.

Index Terms—Bigdata, Collaborative, Hadoop, HBase, MapReduce, Product, Recommendations, Spark, SVD

I. INTRODUCTION

Product database are repetitively becoming larger, making it progressively difficult for impartial systems to process products data. The large expanse of data is available on the network in the custom of ratings, ranks, appraisals, ideas, complain, explanations, response, and comments about any product which can be any product, event, distinct and services which can be used to make correct conclusion [1, 2]. Moreover lots of blog forums are existing on the web where web users can give their judgment, examinations, and comments about the items. The recommendation based on the ranking and summary of appropriate text about the items can be used for decision making. The growth of e-commercesites and online dealings are improving the requirements of a stout recommendation system. Now a day the many of users buy products from online shopping websites. Approaches of praising innovative things take their boundaries, particularly for information discovery [3, 4]. Recommender systems have convert extremely common in current ages, and are exploited in a various areas some applications which are movies, music, news, books, research objects, search enquiries, social codes, and products. These systems are advantageous substitute to search algorithms as they benefit users to determine stuffs they might not have set up by themselves.

Recommender systems, can be divided into following categories: the content-based filtering, the collaborative filtering and hybrid recommender system [5].

A. Content-based filtering

This filtering method is centred on the item and profile of users fondness. In the content-based filtering, keywords are used to define the items and a user summary is erected to designate the type of item the user likes. The content based recommender technique is made, using the buyers profiles based on their pre-experienced item features and the consistency with other items. With the use of the customers outlines, it can imitate the results of the recommendation or customers histories of purchases with ease. There are several shortcomings in the content-based filtering. In this filtering, the attributes of recommending goods must be textual and it cannot recommend multi-media files unswervingly and the recommendation is made based only on the customers pre-experience, it is possible to recommend specific items that are fit to customers own preference and impossible to recommend unrelated products which are different from what they purchased in the past [6, 7].

B. Collaborative filtering

This filtering method is based on collecting and analysing massive amount of records on users preferences, activities and behaviour. To make up for limitations of content based filtering, the collaborative filtering is accepted in the recommender system. It predicts based on what user will like built on the resemblance among other users. The collaborative filtering can be said that the most widespread item is recommended for every customer. It is recognized as the most commercially successful recommender technique. This method is based on the assumption that the user who like an item in past will also the like the same or similar type of item in the future. Collaborative filtering practice can be partitioned into the user-based technique using the relations among customers and the item-based technique using the associations between goods [8, 9]. Examples of this filtering are Facebook and LinkedIn recommendations.


C. Hybrid filtering

This filtering is the grouping of content based and collaborative filtering approach. Hybrid approach was presented to handle with a difficulty of traditional recommendation systems. Two foremost problems that are mentioned by researchers in this field are cold-start problem and stability versus plasticity problem. Cold-start problem happens when learning based techniques like collaborative, content-based, and demographic recommendation algorithms are used. Netflix is the example of hybrid recommendation system [10, 11].

The rest of the paper is ordered as follows. Review of literature is explained in section II. Section III contains our system overview, section IV contains system design, section V contains mathematical model and Conclusion is given in section VI.

II. REVIEW OF LITERATURE

Gejianxin and Liu jiaomin proposed the recommender system for software test cases founded on collaborative filtering the paper contains value and software test case design. At the identical time it analyses the compensations and difficulties of various algorithms in changed computational complexity and under the situation of the act and suggests a combination organisation filtering algorithm and software test instance recommendation system basic frame, and lastly presents the difficulties need to be upgraded and solved in collaborative filtering recommendation algorithm [1].

Bartosz Kupisz and Olgierd Unold projected Collaborative Filtering Recommendation Algorithm based on Hadoop and Spark the purpose of this work was to mature and associate recommendation systems which practise the tem-based collaborative filtering algorithm, based on Hadoop and Spark. The Hadoop form was applied with the use of the Mahout library which was a component of the Hadoop system [2].

Riyaz P A, Surekha Mariam Varghese proposed a Scalable Product Recommendations using Collaborative Filtering in Hadoop for Bigdata. Conservative recommender service structures often grieve from absence of scalability as well as competence problems when dispensation or examination of this data on a bulky scale. To evade these difficulties, a new recommendations system using collaborative filtering algorithm is implemented in Apache Hadoop leveraging MapReduce paradigm for Bigdata [3].

Dheerajkumar Bokde, Sheetal Girase and Debajyoti Mukhopadhyay proposed an method to a university recommendation by multi-criteria collaborative filtering and dimensionality reduction techniques this paper proposes key not only to decrease the computation cost but also increases the estimate truth and efficiency of the multi-criteria filtering algorithms implemented using the Apache Mahout context [4].

Zhiyang Jia, Wei Gao, Yuting Yang, Xu Chen, proposed the system for tourist attraction based on user based collaborative filtering here system is created as an online application which is capable of producing a custom-made list of fondness attractions for the tourist. In order to govern the likenesses between each user, the cosine arrangement is adopted during the evolution of the generation of neighbours [5].

Jyoti Gupta, Jayant Gadge proposed Performance Analysis of Recommendation System Based on Collaborative Filtering and Demographics. Prediction using item based collaborative filtering is collective with prediction using demographics based employer clusters in a weighted scheme. The expected solution is scalable while magnificently lecturing user cold start [6].

Suman Datta, Joydeep Dasy, Prosenjit Gupta and Subhashis Majumder, proposed SCARS: A Scalable Context-Aware Recommendation System. Main impartial of this graft is to moderate the running interval without bargaining the recommendation excellence. This safeguards scalability, agreeing us to challenge bigger datasets using the same means [7].

FAN Lu, LI Hong, LI Changfeng, suggested the enhancement and employment of distributed item-based collaborative filtering algorithm on Hadoop in this paper adopts real data set to run the algorithm and the researchs result couriers that the developed algorithm can run efficiently on the outsized volumes of data with the improved accuracy, and at the similar time, can overwhelmed the cold-starting drawback successfully [8].

Shunmei Meng, Wanchun Dou, Xuyun Zhang, and Jijinun Chen offered KASR: A Keyword Aware Service Recommendation Method on MapReduce for Big Data Applications to expand its scalability and efficiency in big data background, KASR is instigated on Hadoop, an extensively adopted disseminated computing platform using the MapReduce parallel processing archetype [9].

Kunhui Lin, Jingjin Wang, Meihong Wang, proposed a hybrid recommendation algorithm based on Hadoop. The earlier hybrid recommendation algorithm was designed to parallel on MapReduce framework. The experimentations were applied to the MovieLens dataset to achieve the welfares of parallel algorithm [10].

Yingya Zhang, Cheng Yang, Zhixiang Niu proposed the recommendation scheme for the investigation of job created on collaborative filtering this paper user-based and item-based collaborative filtering algorithm is compared to select a better executed one. It takes background information containing students resumes and details of recruiting material into consideration, bring weights of co-apply users that is the users who had smeared the candidate jobs and weights of
student used-liked jobs into the algorithm [11].

Poonam Ghuli, Atanu Ghosh and Dr. Rajashree Shettar suggested a Collaborative Filtering Recommendation Engine in a Distributed Environment this paper focus on parallelism of both item-based and user-based CF algorithm distinctly on MapReduce by excruciating the CF algorithm computations into three Mapper and Reducer phases [12].

Xiao Peng, Shao Liangshan, Li Xiuran proposed improved collaborative filtering algorithm in the inquiry and solicitation of personalized movie references. The electronic business recommendation system, it emphases on the collaborative filtering algorithm in the solicitation of personalized movie recommendation system [13].

Suyun Wei, Ning Ye, Shuo Zhang, Xia Huang, Jian Zhu proposed Item-based Collaborative Filtering recommendation Algorithm Combining Item Category with interestingness measure a top-N recommendation algorithm that uses items classes similarity and item-item interestingness to calculate the recommendations. Results presented that proposed algorithm affords additional accurate recommendations than those given by traditional CF techniques [14].

Michael D. Ekstrand, John T. Riedl and Joseph A. Konstan., proposed Collaborative Filtering Recommender Systems, this paper deliberates a varied diversity of the selections accessible and their implications, pointing to provide both experts and investigators with an introduction to the important subjects essential for recommenders and existing superlative practices for addressing these issues [15].

Wu Yueping and Zheng Jianguo proposed a research of recommendation algorithm based on cloud model. The paper use cloud model which are knowledge demonstration in quality and bond function of the conversion between quality and quantity, provide an item cataloguing recommendation algorithm based on cloud model [16].

III. SYSTEM OVERVIEW

A. Problem Statement

Recommender provision systems often grieve from dearth of scalability and efficiency glitches when handling or scrutiny of this data on an outsized scale. Traditional filtering algorithms suffer from scalability problems, as well as many systems need to react instantly to online recommendations for all users nevertheless of their consumptions and assessments antiquity, which hassles large scalability To circumvent these problems, a proposed recommendation system using filtering algorithm are implemented in Apache Hadoop and Spark leveraging MapReduce paradigm for Bigdata. Our main objective is developing a scalable big data exploration system with recommendation related algorithms applied on top of the Hadoop and Spark which delivers scalable and real time recommendations.

B. Proposed System Architecture

Hadoop has a distributed file structure so it takes the input data from various file systems. Input data is taken from datasets. From this dataset we are taking 70 percent data as training and 30 percent data for testing. The data is collected in a structured format and transferred from local file system to Hadoop distributed file system (HDFS) as shown in Fig 1.

Here on Apache spark we are implementing our algorithms. We are using Apache spark over Hadoop system because Apache spark runs 100x faster than Hadoop MapReduce. Then by using the metadata of user, movies and ratings matrix is computed over Spark.
We will combine the collected data in matrix format. Using Singular vector decomposition algorithm matrix is factorized over spark so as to predict recommendations. To predict user ratings we are factoring our matrix which is in HDFS into vectors and values. Then we associate this factorized data with test user input data, this value which we get by combining factorized matrix and user test data should approximately match with our original training matrix data. Test user input data comes from the inputs testing data which is stored in HDFS [12].

Then recommendations are provided using singular vector with the help of test data. Then we are computing the RMSE that is root mean square error and then visualization is provided using web interface. [13].

Root Mean Square Error (RMSE) puts more emphasis on larger absolute error and the lower the RMSE is, the better the recommendation accuracy.

\[
RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (\hat{y}_{i} - y_{i})^2}
\]

Product data is loaded into the Hadoop clusters with MapReduce paradigm. Data is encumbered into clusters so that each node will be able to process data simultaneously. The data is changed in accordance to the customer rather than the product. The user perceptions are compared with other users. So the data is arranged as a customer with list of products he had purchased. Product big data influences MapReduce technique for fast court. Data abstraction problems framed as key-value pairs can be capably dispersed with Hadoop as well as HDFS [14, 15].

Here we will test our implementation with minimum 3 nodes using Apache spark for computation and HDFS for storage of movies data. We will be using Movielens as our dataset

The main modules which we are implementing in our project are item-based recommendation, SVD that is singular value decomposition, then we are implementing incremental singular value decomposition and Trust Singular value decomposition. In item based recommendation similarity between items calculated using people’s ratings of those items. SVD finds a hidden feature space where the users and items they like have feature vectors that are closely aligned it is commonly used for producing low-rank approximations. Incremental SVD is used to handle dynamic databases, where new terms and documents may arrive once the model is built it handles incremental databases. The rationale behind TrustSVD is to take into consideration user/item biases and the influence of rated items other than user/item-specific vectors on rating prediction

C. Comparison with similar systems

Here comparison of similar systems is given. Similar systems are content-based, collaborative and hybrid recommendation system.

<table>
<thead>
<tr>
<th>Sr No.</th>
<th>Content-based Recommendation system</th>
<th>Collaborative Recommendation System</th>
<th>Hybrid Recommendation System</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Items with similar properties are considered</td>
<td>Here users behaviour is taken into consideration</td>
<td>Combination of both is taken into consideration</td>
</tr>
<tr>
<td>2</td>
<td>Generate preferences based on similarity among items</td>
<td>Preferences are generated considering objects and users</td>
<td>Combination of both is used to provide recommendation.</td>
</tr>
<tr>
<td>3</td>
<td>If performs well on fresh content and needs generation of multiple features.</td>
<td>Maximum data will provide proper recommendations.</td>
<td>It provides better results than the other techniques.</td>
</tr>
<tr>
<td>4</td>
<td>This technique is more personalized.</td>
<td>This technique is not much personalized.</td>
<td>Provide recommendation using both content based and collaborative approach.</td>
</tr>
</tbody>
</table>

IV. SYSTEM DESIGN

System design contain the class diagram which specifies the classes of system. Our system contains classes

![Fig. 2. Class Diagram](image-url)
SVD computation recommendation and visualization. Class preprocessing contains metadata and ratings as attributes and performs user ratings and matrix computations. The class SVD contains attributes such as ratings from users and matrix and perform operations such as left, right singular vectors and singular vectors[12]. Recommendation class contains attributes such as user preferences, left, right singular vectors and singular vectors it perform operations such as rate predictions and give ratings. Visualization class contains attributes like user metadata, predictions and ratings and provides the recommendations.

V. Mathematical Model

Let S contains the set of attributes required for the computation of SVD based recommendations.

\[ S = \{U_r, P, R, M, U, \sum, V, T, P_r\} \]

\[ U_r = \{U_{r1}, U_{r2}, U_{r3}...U_{rm}\} \]

\[ U \rightarrow \text{set of users} \]

\[ m \rightarrow \text{Number of users} \]

\[ P = \{P_1, P_2, P_3...P_n\} \]

\[ P \rightarrow \text{set of products(i.e.Movie)} \]

\[ n \rightarrow \text{Number of movies} \]

\[ R = \{R_1, R_2, R_3...R_l\} \]

\[ R_i = \{U_i, P_j, RT_k\} \]

Where,

\[ U_i \rightarrow \text{Users} \]

\[ P_j \rightarrow \text{Movie} \]

\[ RT_k \rightarrow \text{Ratings} \]

\[ l = m \times n \]

Function \( F_1(U_r, P, R) \rightarrow M \)

Function \( F_1 \) constructs matrix from user metadata, product metadata and ratings

Function \( F_2(M) \rightarrow (U, \sum, V) \)

Function \( F_2 \) factorizes matrix \( M \) into \( U, \sum \) and \( V \)

Where,

\[ U \rightarrow \text{Left singular vector} \]

\[ \sum \rightarrow \text{Singular values} \]

\[ V \rightarrow \text{Right singular vectors} \]

Function \( F_3(U, \sum, V, T) \rightarrow (P_r) \)

For test user input data, function \( F_3 \) predicts ratings

Where,

\[ T \rightarrow \text{Test user input data} \]

VI. Result Analysis

In our project we will implement recommendation system over Hadoop and Apache Spark. Here we are storing our data over Hadoop. Data from various sources is collected at a single place which is in structured format. This structuring of data is necessary as it reduces the problem of duplicate data. Once we have all the data in one place we can use Apache Spark to do in memory computation on the data. Apache Spark is used as it provides low latency computation. It is a cluster computing system. It does in memory processing of data. It uses various machine learning libraries so as processing can be done faster. Hadoop reads and write data to disk which takes extra time to do computation. So a recommendation engine build using only Hadoop will not provide real time recommendation so we are using Spark to provide real time computation of data.

A. Performance Measure

1) Accuracy Measure:

Using Single value decomposition algorithm in recommendation system we will get accurate recommendation. Applying this algorithm to dataset which is in the form of matrix will provide ratings and recommendation for the product.

2) Robustness measure:

The robustness accuracy score (RAS) aims at measuring the ability of an algorithm to make good predictions in the presence of noisy data. This score therefore compares the results provided by the algorithm and the results obtained by an ideal algorithm.

3) Computing-time measure:

Computing time should be divided into two parts. The first part computes the time needed by a particular algorithms to provide recommendations; called computing time while the second part computes the time needed to compute new recommendations or to update old recommendations by adding new users, new items.

B. Result Analysis

Here we are implementing our project on serial and distributed processing. Using serial computation for processing
maximum time is required and as small number of nodes are used it requires large processing time. In HDFS computation as the number of datanodes increases it improves the processing performance.

### Table II

<table>
<thead>
<tr>
<th>Sr No</th>
<th>Serial Computation</th>
<th>Distributed Computation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>It requires maximum time.</td>
<td>It requires minimum time.</td>
</tr>
<tr>
<td>2</td>
<td>It performs computation on one node at a time.</td>
<td>It performs computation on multiple nodes at a time.</td>
</tr>
<tr>
<td>3</td>
<td>Using this computation recommendation of one product is computed at one time.</td>
<td>Using this computation method recommendation of multiple products for multiple users is given at a time.</td>
</tr>
<tr>
<td>4</td>
<td>Here sequential processing of products is done.</td>
<td>Here parallel processing of multiple products is done.</td>
</tr>
<tr>
<td>5</td>
<td>With small number of datanodes large processing time is required.</td>
<td>As the number of datanodes increases improves the processing time.</td>
</tr>
<tr>
<td>6</td>
<td>If number of users are 30000 then 80 seconds are required for serial computation.</td>
<td>If number of users are 30000 then 60 seconds are required for distributed computation.</td>
</tr>
</tbody>
</table>

### VII. Conclusion

An ascendable product recommendation filtering for Bigdata on Apache Hadoop and Spark can process superior than regular recommendation system based on Hadoop. A boosted HBase gives better presentation. For low expectancy applications HBase is highly preferred as of distributed architecture and leverage the influence of Apache Hadoop. Apache spark is a reckless and universal engine for large scale records processing. It runs programs up to 100x quicker than Hadoop MapReduce in memory, or 10x nearer on disk, so we are realizing our system on Spark to provide scalability and real time commendations. As the size of data escalates the Apache Spark accomplishes well by accumulating additional data nodes for processing.

### Acknowledgment

I would like to thank all the people who have been of the help and assisted me throughout my project work. First of all I would like to thank my respected guide Prof. R.S. Shirsath, Prof. A. D. Potgantwar Head of Department Computer Engineering for introducing me throughout features needed. The time-to-time guidance, encouragement, and valuable suggestions received from him are unforgettable in my life. This work would not have been possible without the enthusiastic response, insight, and new ideas from him. I am also grateful to all the faculty members of SITRC College of Engineering for their support and cooperation.

### References


