Chapter 1

INTRODUCTION

1.1 An overview

In the recent years WWW has grown on exponential rate and that resulted into huge amount of data. This has become an opportunity as well as a problem for user because finding the right information has become difficult. E- commerce has been gone through challenges of managing this huge explosion of information and utilized it in a smarter way by using recommendation systems. By the rise in online shopping, recommender systems has become basic need of every e-commerce portal. Recommendation systems comes with the capability of predicting the suggestions for its users on the bases of user's past behavior or the by the behavior of similar users. Amazon, Netflix and other such portal use recommender systems extensively for suggesting content to their users.

There is an extensive class of Web applications that involve predicting user responses to options. Such a facility is called a recommendation system. Recommendation plays an increasingly important role in our daily lives. Recommender system (RS) automatically suggests to a user items that might be of interest to her. Recent studies demonstrate that information from social networks can be exploited to improve accuracy of recommendations. In this dissertation, we look specifically at how Hadoop can be used to enhance the scalabilty of recommender. Most large-scale commercial and social websites recommend options, such as products or people to connect with, to users. Recommendation engines sort through massive amounts of data to identify potential user preferences.

1.1 General concepts

In the recent years, the Web has undergone a tremendous growth regarding both content and users. This has lead to an information overload problem in which people are finding it increasingly difficult to locate the right information at the right time. But the large amount of data has improved the decision making. Recommender systems have been developed to help user to make better decisions, by guiding users through the big ocean of information. Until now, recommender systems have been extensively used within e-commerce and communities where items like movies, music and articles are recommended. The World Wide Web contains an enormous amount of information. Recent statistics also show that the number of Internet users is high and rapidly growing. The tremendous growth of both information and usage has lead to a so-called information overload problem in which users are finding it increasingly difficult to locate the right information at the right time. As a response to this problem, much research has been done with the goal of providing users with more proactive and personalized information services. Recommender systems have proved to help achieving this goal by using the opinions of a community of users to help individuals in the community more effectively identify content of interest from a potentially overwhelming set of choices. Two recommendation strategies that have come to dominate are content based and collaborative filtering. Content-based filtering rely on rich content descriptions of the items that are being recommended , while collaborative filtering recommendations are motivated by the observation that we often look to our friends for recommendations. Systems using recommendations have been developed in various research projects. Later, several research projects have focused on recommender systems, either by introducing new concepts, or by combining old concepts to make better systems. Recommender systems have also been deployed within commercial domains, for example in e-commerce applications.

A well-known example is Amazon, where a recommender system is used to help people find items they would like to purchase. Many online communities within the movie domain use recommender systems to gather user opinions on movies, and then produce recommendations based on these opinions. Examples are MovieFinder and Movielens. New popular music services like Pandora and Last.fm also make use of recommendations to configure personalized music players.

1.3 Motivation

Communication networks facilitate easy access of information. Meanwhile, the richness of online information also brings forth the “information overload” problem. For example, if one wants to buy a digital camera, it would be a frustrating experience for her to read through and compare all online reviews about digital cameras before making the purchase decision. Recommender systems deal with information overload by automatically suggesting to users items that may fit their interests. Accurate recommendations enable users to quickly locate desirable items without being overwhelmed by irrelevant information. It is also of great interest for vendors to recommend those products that match the interests of each of the visitors of their websites, and hopefully turn them into satisfied and returning customers.

Recommender System (RS) roots back to several related research disciplines, such as cognitive science, approximation theory and information retrieval, etc. Due to the increasing importance of recommendation, it has become an independent research field since the mid 1990s [1]. Broadly speaking, a RS suggests to a user those items that might be of her interest. Generally, there are two variants of recommendation approaches:

- content-based approaches
- collaborative-filtering based approaches

CF approaches can be further grouped into model-based CF and neighbourhood-based CF [2]. Model-based approaches use user-item ratings to learn a predictive model. The general idea is to model the user-item interactions with factors representing latent features of users and items in the system, such as the preference class of users and the category class of items. In contrast, neighbourhood-based CF approaches use user-item ratings stored in the system to directly predict ratings for new items.

Online social networks (OSN) present new opportunities as to further improve the accuracy of RSs with the huge data it provides. Due to stable and long-lasting social bindings, people are more willing to share their personal opinions with their friends, and typically trust recommendations from their friends more than those from strangers and vendors. Popular online social networks, such as Facebook [39], Twitter [38], and Youtube [35],provide novel ways for people to communicate and build virtual communities. Online social networks not only make it easier for users to share their opinions with each other, but also serve as a platform for developing new RS algorithms to automate the otherwise manual and anecdotal social recommendations in real-life social networks. Internet data is huge and processing this data for recommendations is a crucial job so there should be some approach to handle large data sets for recommendation systems. Apache Hadoop can be an answer to scalability problem.

1.4 Related Work

Area of recommendation has been became interest of researcher from past two decades. Because of its close relation to e-commerce business it has fascinated researchers as well as industries to work in the field of recommendation system. Research is going in the field of content recommendation and friend recommendation. With the popularity of online social networks, potential friend recommendation has turned into area of interest for many researchers. Here is some related work listed. Deuk Hee Park et.al. have presented a classification and literature review in their paper and gives insight about past work and future scope of area[4].

X. Yang et.al. have compared recommender systems based on collaborative filtering [5]. Ruzhi Xu et.al. implemented CF and made predictions using singular ratings for large scale recommendation using Hadoop MapReduce distributed framework for improving efficiency [7]. J. Bobadilla et.al. provided an overview of recommender systems and algorithms used in this system [8]. Saikat Bagchi has studied the performance and quality of similarity measures for CF using Mahout and observed that Euclidean Distance Measure perform better than other measures provided in Mahout [9]. Xiwei Wang et.al. has studied different algorithm and models for recommender systems [10]. Juha Leino and Kari-Jouko Räihä tried to understand the user strategies used in complex e-commerc system by taking the case study of Amazon [11]. Carlos E. Seminario and David C. Wilson has given a case study on Mahout and evaluated it as a recommender system platform[12]. Xing Xie has designed a friend recommendation framework on the basis of user interest characterization in online social network [13]. G. Adomavicius and A. Tuzhilin reviewed different recommendation methods and studied their limitations and discussed extensions to methods to provide better results in recommendations.

1.5 Problem Statement

From the foregoing section we can conclude that there are not concrete proposals for suggesting recommendation from large scale online social network data set. Existing approaches do not have good scalabilities to process huge volumes of data. Due to huge size of data, performance may degrade, and we cannot find an efficient solution. Hence we require distributed environment so that computation can be increased and performance of recommendation system gets improve.

Hence the problem of thesis is

"To customize a recommendation system using Collaborative Filtering (CF) on the top of Hadoop platform and Apache Mahout for handling the scability problem for large scale dataset."

1.6 Organization of Thesis

The rest of this thesis is organized as follows:

Chapter 2 gives a brief idea about recommendation systems. It includes its definition, models for recommendation systems and algorithms that broadly used in recommendations, applications and challenges of recommendation systems.

Chapter 3 provides basics of relation between Big Data and recommendation systems. Its gives an idea about scalability problem , introduction to Apache Hadoop and Apache Mahout.

Chapter 4 , in this section we have done a case study on famous recommendation system for online shopping portal Amazon.com and tried to understand the functionality and expectations from recommender system.

Chapter 5 includes our implementation work. It includes proposed idea and then implementation.

Chapter 6 shows the result and conclusion of work.

Chapter 2

RECOMMENDATION SYSTEM OVERVIEW

In this chapter we first discuss the basic definition and idea behind Recommender systems. Then the various models of RS with their merits and demerits are given for better understanding, after that we discuss algorithms and finally we discuss available challenges for RS.

2.1 Definition

There is an extensive class of Web applications that involve predicting user responses to options. Such a facility is called a recommendation system.

"*Recommender Systems (RSs) are software tools and techniques providing suggestions for items to be of use to a user. The suggestions relate to various decision-making processes, such as what items to buy, what music to listen to, or what online news to read.*"

We shall begin this chapter with a survey of the most important examples of these systems. However, to bring the problem into focus, two good examples of recommendation systems are:

- Offering news articles to on-line newspaper readers, based on a prediction of reader interests.
- Offering customers of an on-line retailer suggestions about what they might like to buy, based on their past history of purchases and/or product searches.

Recommendation systems use a number of different technologies. We can classify these systems into two broad groups.

- Content-based systems examine properties of the items recommended. For instance, if a Netflix user has watched many cowboy movies, then recommend a movie classified in the database as having the “cowboy” genre.
- Collaborative filtering systems recommend items based on similarity measures between users and/or items. The items recommended to a user are those preferred by similar users. on clustering. However, these technologies by themselves are not sufficient, and there are some new algorithms that have proven effective for recommendation systems.

We can find recommender systems in many of the websites we use every day. including these well-known examples:

- **LinkedIn**, the business-oriented social networking site, forms recommendations for people you might know, jobs you might like, groups you might want to follow, or companies you might be interested in. LinkedIn uses Apache Hadoop to build its specialized collaborative-filtering capabilities.
- **Amazon**, the popular e-commerce site, uses content-based recommendation. When we select an item to purchase, Amazon recommends other items other users purchased based on that original item (as a matrix of item-to-likelihood-of-next-item purchase). Amazon patented this behavior, called *item-to-item collaborative filtering*.
- **Hulu**, a streaming-video website, uses a recommendation engine to identify content that might be of interest to users. It also uses (offline) item-based collaborative filtering with Hadoop to scale the processing of massive amounts of data. Details of Hulu's online and offline ItemCF architecture are publicly available.
- **Netflix**, the video rental and streaming service, is a famous example. In 2006, Netflix held a competition to improve its recommendation

system, Cinematch. In 2009, three teams combined to build an ensemble of 107 recommendation algorithms that resulted in a single prediction. This ensemble proved to be the key to improving predictive accuracy, and the combined team won the prize.

2.2 Idea behind Recommender System

As the World Wide Web continues to grow at an exponential rate, the size and complexity of web pages grow along with it. Different techniques have been applied to develop systems that help users find the information they seek. These techniques belong to the fields in software technology called *information retrieval* and *information filtering*.

2.2.1 Information retrieval and filtering

The rapidly expanding Internet has given the users the ability to choose among a vast variety of information [27], whether it is information concerning their profession, events in their world, or information that allows them to maintain their lifestyle. The information that is needed to fulfil these continuously increasing demands can come from different sources. Examples are web pages, emails, articles, news, consumer journals, shopping sites, online auctions and multimedia sites. Even though the users profit from the enormous amount of information that the sources provide, they are not able to handle it. This information overload problem [48] is the reason why several techniques for information retrieval and information filtering have been developed. Although the goal of both information retrieval and information filtering is to deal with the information overload problem by examining and filtering big amounts of data, there is often made a distinction between the two [7].

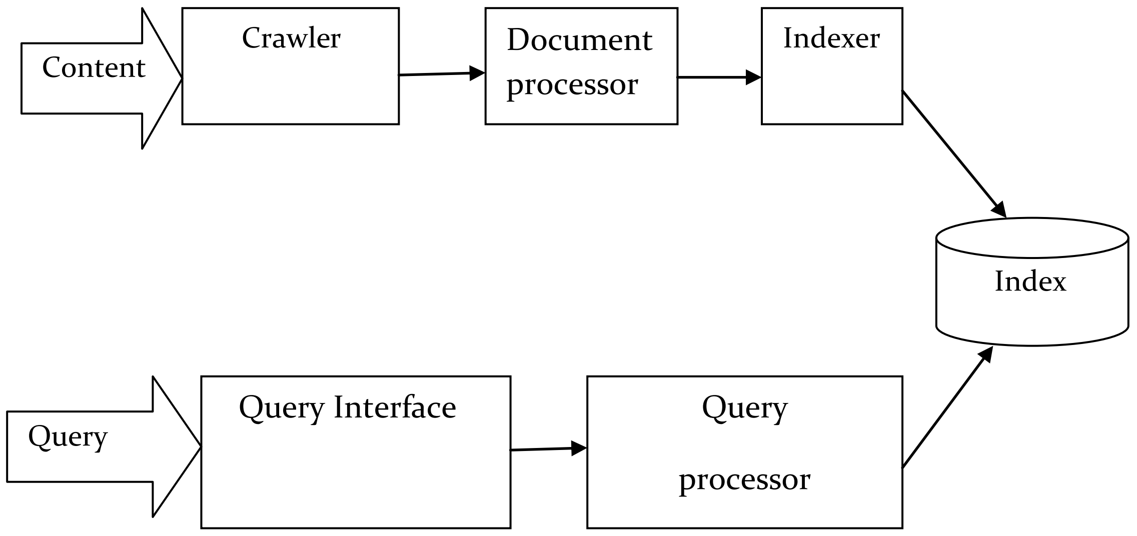


Figure 2.1: Information retrieval.

- **Information retrieval (IR),** often associated with data search, is a technology that may include *crawling*, *processing* and *indexing* of content, and *querying* for content. The normal process of IR is showed in figure 2.1. Crawling is the act of accessing web servers and/or file systems in order to fetch information. By following links, a crawler is able to traverse web content hierarchies based on a single start URL. The document-processing stage may add, delete or modify information to a document, such as adding new meta information for linguistic processing, or extracting information about the language that the document is written in. Indexing is a process that examines content that has been processed and makes a searchable data structure, called Index, that contains references to the content. Queries are requests for information. IR systems let a user write a query in form of keywords describing the information needed. The user can interact with the IR system through a Query interface. A Query-processor will use the index to find information references based on the keywords and then display the references. The goal is to analyze and identify the essence of the user's intent from the query, and to return the most relevant set of results. Filtering of information in IR systems is done by letting the user specify what information is needed by manually typing keywords describing the information. IR is very successful at supporting users who know how to describe exactly what they are looking for in a manner that is compatible with the descriptions of the content that were created during the indexing.

- **Information filtering (IF)** systems focus on filtering information based on a user's *profile*. The profile can be maintained by letting the user specify and combine interests explicitly, or by letting the system implicitly monitor the user's behavior. Filtering within IF systems is done when the user automatically receives the information needed based on the user's profile. The advantage of IF is its ability to adapt to the user's long-term interest, and bring the information to the user. The latter can be done by giving a notice to the user, or by letting the system use the Information to take action on behalf of the user.

Closely related to IF is the idea of having a system that acts as a personalized decision guide for users, aiding them in decision making about matters related to personal taste. Systems that realize this idea are called *recommender systems*.

Recommender systems (RS) [49] are used in a variety of applications. Examples are web stores, online communities, and music players. Currently, people mostly tend to associate recommender systems with e-commerce sites, where recommender systems are extensively used to suggest products to the customers and to provide customers with information to help them decide which products to purchase. Products can be based on the top overall sellers on a site, on the demographics of the consumers, or on an analysis of the past buying behaviour of the consumers as a prediction for future buying behaviour [53]. This is shown in figure 2.2.

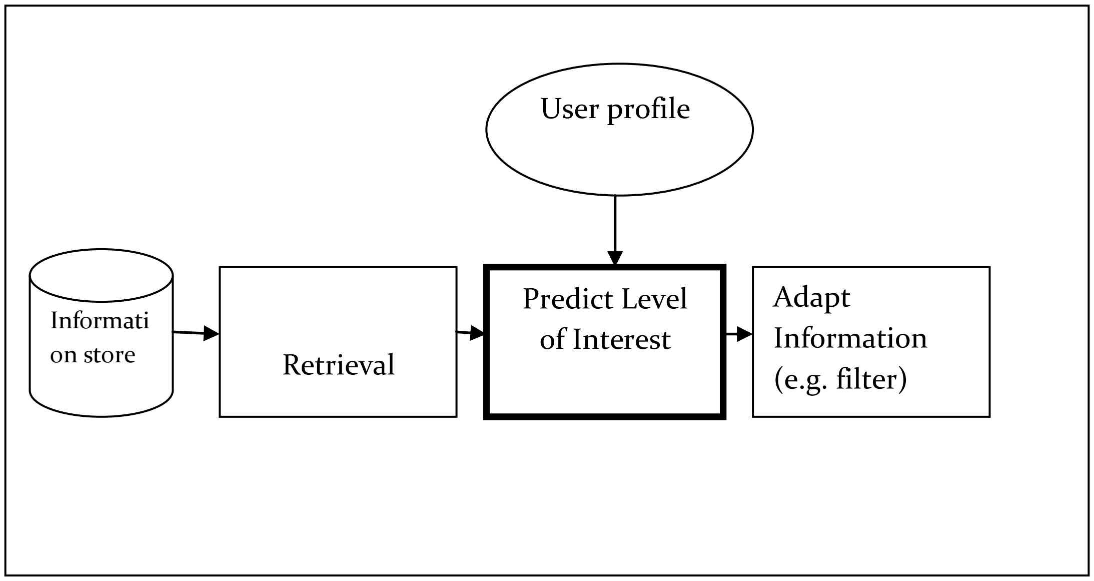


Figure 2.2: Information filtering in recommender systems.

2.3 Models for Recommender System

Most recommender systems take either of two basic approaches: collaborative filtering or content-based filtering. Other approaches (such as hybrid approaches) also exist.

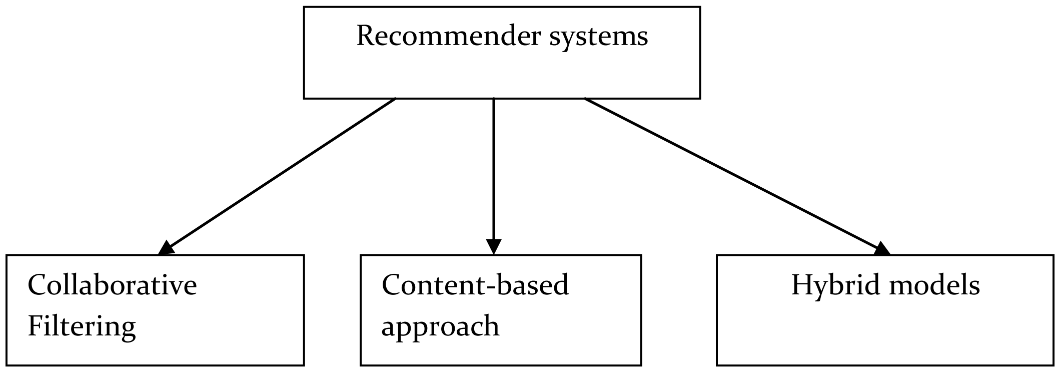


Figure 2.3 : Models for Recommender systems

2.3.1 Collaborative filtering (CF) approach

Collaborative filtering arrives at a recommendation that's based on a model of prior user behavior. The model can be constructed solely from a single user's behavior or more effectively also from the behavior of other users who have similar traits. When it takes other users' behavior into account, collaborative filtering uses group knowledge to form a recommendation based on like users. In essence, recommendations are based on an automatic collaboration of multiple users and filtered on those who exhibit similar preferences or behaviors.

2.3.2 Content-based filtering

Content-based filtering constructs a recommendation on the basis of a user's behavior. For example, this approach might use historical browsing information, such as which blogs the user reads and the characteristics of those blogs. If a user commonly reads articles about Linux or is likely to leave comments on blogs about software engineering, content-based filtering can use this history to identify and recommend similar content (articles on Linux or other blogs about software engineering). This content can be manually defined or automatically extracted based on other similarity methods.

2.3.3 Hybrid Model

Hybrid approaches that combine collaborative and content-based filtering are also increasing the efficiency (and complexity) of recommender systems. Incorporating the results of collaborative and content-based filtering creates the potential for a more accurate recommendation. The hybrid approach could also be used to address collaborative filtering that starts with sparse data known as cold start by enabling the results to be weighted initially toward content-based filtering, then shifting the weight toward collaborative filtering as the available user data set matures.

2.4 Algorithms

Many algorithmic approaches are available for recommendation engines. Results can differ based on the problem the algorithm is designed to solve or the relationships that are present in the data.

Many of the algorithms come from the field of machine learning, a subfield of artificial intelligence that produces algorithms for learning, prediction, and decision-making.

2.4.1 Pearson correlation

Similarity between two users (and their attributes, such as articles read from a collection of blogs) can be accurately calculated with the *Pearson correlation.* This algorithm measures the linear dependence between two variables (or users) as a function of their attributes. But it doesn't calculate this measure over the entire population of users. Instead, the population must be filtered down to *neighborhoods* based on a higher-level similarity metric, such as reading similar blogs. The Pearson correlation, which is widely used in research, is a popular algorithm for collaborative filtering.

2.4.2 Clustering algorithms

Clustering algorithms are a form of unsupervised learning that can find structure in a set of seemingly random (or unlabeled) data. In general, they work by identifying similarities among items, such as blog readers, by calculating their distance from other items in a *feature space.* (Features in a feature space could represent the number of articles read in a set of blogs.) The number of independent features defines the dimensionality of the space. If items are "close" together, they can be joined in a cluster.

Many clustering algorithms exist. The simplest one is *k*-means, which partitions items into *k* clusters. Initially, the items are randomly placed into clusters. Then, a *centroid* (or *center*) is calculated for each cluster as a function of its members. Each item's distance from the centroids is then checked. If an item is found to be closer to another cluster, it's moved to that cluster. Centroids are recalculated each time all item distances are checked. When stability is reached (that is, when no items move during an iteration), the set is properly clustered, and the algorithm ends. Calculating the distance between two objects can be difficult to visualize. One common method is to treat each item as a multidimensional vector and calculate the distance by using the Euclidean algorithm.

Other clustering variants include the Adaptive Resonance Theory (ART) family, Fuzzy C-means, and Expectation-Maximization (probabilistic clustering), to name a few.

2.4.3 Other algorithms

Many algorithms, larger set of variations of those algorithms exist for recommendation engines. Some that have been used successfully include:

- **Bayesian Belief Nets**, which can be visualized as a directed acyclic graph, with arcs representing the associated probabilities among the variables.
- **Markov chains**, which take a similar approach to Bayesian Belief Nets but treat the recommendation problem as sequential optimization instead of simply prediction.
- **Rocchio classification** (developed with the Vector Space Model), which exploits feedback of the item relevance to improve recommendation accuracy.

2.5 Challenges with Recommender Systems

- **Large scale data**: The massive amounts of available data also complicate the opportunity of making recommendation easily. For example, although some users' behavior can be modeled, other users do not exhibit typical behavior. These users can skew the results of a recommender system and decrease its efficiency. Further, users can exploit a recommender system to favor one product over another based on positive feedback on a product and negative feedback on competitive products, for example. A good recommender system must manage these issues.

- **Privacy:** Privacy-protection considerations are also a challenge. Recommender algorithms can identify patterns individuals might not even know exist. A recent example is the case of a large company that could calculate a pregnancy-prediction score based on purchasing habits. Through the use of targeted ads, a father was systems is scalability. Traditional algorithms work well with smaller amounts of data, but when the data sets grow, the traditional algorithms can have difficulty keeping up. Although this might not be a problem for offline processing, more-specialized approaches are needed for real-time scenarios. We are going to work on this problem.

2.6 Applications of Recommendation System

Recommender system research is being conducted with a strong emphasis on practice and commercial applications, since, aside from its theoretical contribution, is generally aimed at practically improving commercial RSs. Thus, RS research involves practical aspects that apply to the implementation of these systems. These aspects are relevant to different stages in the life cycle of a RS, namely, the design of the system, its implementation and its maintenance and enhancement during system operation. The aspects that apply to the design stage include factors that might affect the choice of the algorithm. The first factor to consider, the application's domain, has a major effect on the algorithmic approach that should be taken. [72] provide a taxonomy of RSs and classify existing RS applications to specific application domains. Based on these specific application domains, we define more general classes of domains

for the most common recommender systems applications:

- Entertainment - recommendations for movies, music, and books.
- Content - personalized newspapers, recommendation for documents, recommendations of Web pages, e-learning applications, and e-mail filters.
- E-commerce - recommendations for consumers of products to buy such as cloths, accessories , household items, books, electronic items like mobile phones, cameras, PCs etc.

- Services - recommendations of travel services, recommendation of experts for consultation, recommendation of houses to rent or buy, or matchmaking services.

As recommender systems become more popular, interest is aroused in the potential advantages in new applications, such as recommending friends or tweets to follow as in www.tweeter.com. Hence, the above list cannot cover all the application domains that are now being addressed by RS techniques. It gives only an initial description of the various types of application domains.

Chapter 3

RECOMMENDATION SYSTEM AND BIG DATA

3.1 Big Data : large scale user generated data

We live in the Information Era - the Age of BIG DATA. Big Data has become a Buzzword these days, data in every possible form, whether through social media, structured, unstructured, text, images, audio, video, log files, emails, simulations, 3D models, military surveillance, e-commerce and so on, it amounts to around some zettabytes of data. This huge data is what we call as BIG DATA. Big data is nothing but a synonym of a huge and complex data that it becomes very tiresome or slow to capture, store, process, retrieve and analyze it with the help of any relational database management tools or traditional data processing techniques. Data on web is increasing in size every second. Big Data has been very helpful for users and companies to make better decisions as shown in given figure 3.1

according to TCS Big Data Global Trend Study [22]. 2013[].

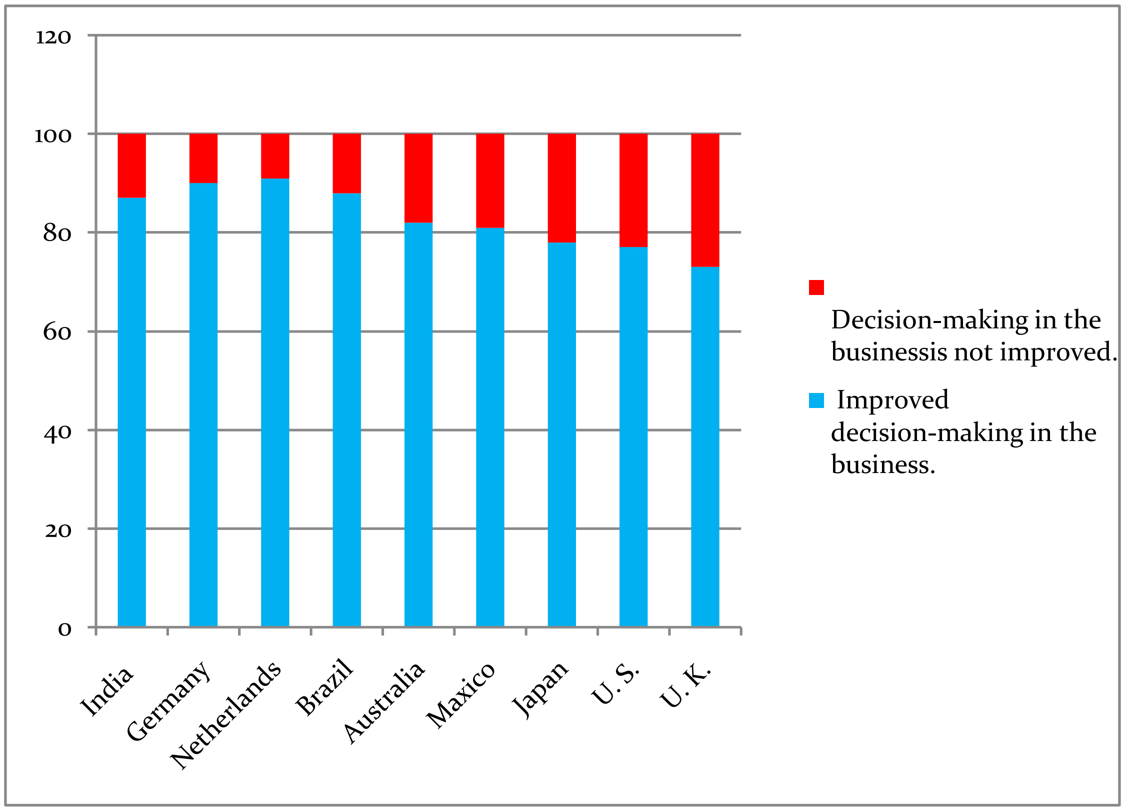


Figure 3.1 : By Country, Percentage of Companies Where Big Data Improved Decision-Making (according to TCS Big Data Global Trend Study 2013)

Online Social Networks (OSNs), such as Facebook, Tweeter and LinkedIn, have become tremendously popular in recent years. Millions of users are active daily in these sites and create a large amount of data online that has not been available before, including user links data and other user-generated content, such as blogs, photos, videos, *etc.* However, the abundance and popularity of OSNs flood users with huge volumes of information and hence bring users the problem of information overload. For example, the specific user is difficult to find potential friends he wants to know and is hard to discover content he wants to consume. Aiming to alleviate information overload over OSNs users, Social Recommender System (SRS) is proposed. This kind of system is different from traditional recommender systems using content-based methods or collaborative filtering methods separately and it can incorporate new techniques that take advantage of explicit links information between users in OSNs to make more effective recommendation. Existing SRSs above have been proved that they have better recommendation precision than traditional recommender systems which do not consider social information between users, they all face the problem of poor scalability. Realistic OSNs often possess a large amount of data, including the large-scale complex social graph data and user-generated content. All of these need more effective computing method than before. If SRS does not scale well enough to process large datasets existing in OSNs, the performance of SRS will be poor.

3.2 Problem of scalability and prospective solutions

One problem that's endemic to large-scale recommendation systems is scalability. Traditional algorithms work well with smaller amounts of data, but when the data sets grow, the traditional algorithms can have difficulty keeping up. Although this might not be a problem for offline processing, more-specialized approaches are needed for real-time scenarios. Platforms like Apache Hadoop and Apache mahout can provide a means to scalable recommendations. We are going to work on this problem and see how well it works.

3.3 Apache Hadoop

Hadoop is an open-source software framework for storing and processing big data in a distributed fashion on large clusters of commodity hardware. Essentially, it accomplishes two tasks: massive data storage and faster processing.

The Apache Hadoop project develops open-source software for reliable, scalable, distributed computing. This framework allows for the storing large data sets which are distributed across clusters of computers using simple programming models and written in Java to run on a single computer to large clusters of commodity hardware computers and it derived from papers published by Google and incorporated the features of the Google File System (GFS) and MapReduce paradigm and named it as Hadoop Distributed File System and Hadoop MapReduce.

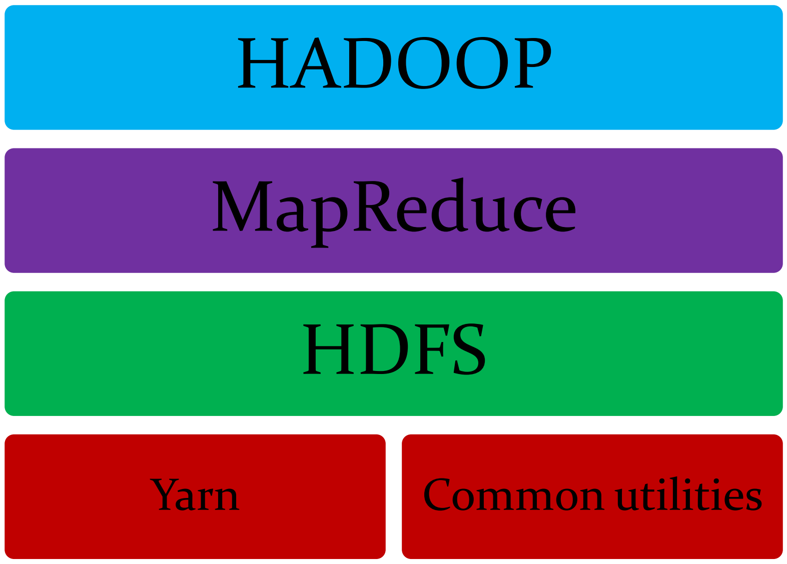


Figure 3.2: The base Hadoop framework.

- **Open-source software.** Open source software differs from commercial software due to the broad and open network of developers that create and manage the programs. Traditionally, it's free to download, use and contribute to, though more and more commercial versions of Hadoop are becoming available.
- **Framework.** In this case, it means everything you need to develop and run your software applications is provided – programs, tool sets, connections, etc.

- **Distributed.** Data is divided and stored across multiple computers, and computations can be run in parallel across multiple connected machines.
- **Massive storage.** The Hadoop framework can store huge amounts of data by breaking the data into blocks and storing it on clusters of lower-cost commodity hardware.
- **Faster processing.** How? Hadoop processes large amounts of data in parallel across clusters of tightly connected low-cost computers for quick results.

With the ability to economically store and process any kind of data (not just numerical or structured data), organizations of all sizes are taking cues from the corporate web giants that have used Hadoop to their advantage.

Some key factors that make Hadoop popular :

- **Low cost.** The open-source framework is free and uses commodity hardware to store large quantities of data.
- **Computing power.** Its distributed computing model can quickly process very large volumes of data. The more computing nodes you use, the more processing power you have.

- **Scalability.** You can easily grow your system simply by adding more nodes. Little administration is required.
- **Storage flexibility.** Unlike traditional relational databases, you don’t have to preprocess data before storing it. And that includes unstructured data like text, images and videos. You can store as much data as you want and decide how to use it later.
- **Inherent data protection and self-healing capabilities.** Data and application processing are protected against hardware failure. If a node goes down, jobs are automatically redirected to other nodes to make sure the distributed computing does not fail. And it automatically stores multiple copies of all data.

3.4 Apache Mahout

Apache Mahout is a new open source project by the Apache Software Foundation (ASF) with the primary goal of creating scalable machine-learning algorithms that are free to use under the Apache license. The project is entering its second year, with one public release under its belt. Mahout contains implementations for clustering, categorization, CF, and evolutionary programming. Furthermore, where prudent, it uses the Apache Hadoop library to enable Mahout to scale effectively in the cloud. While Mahout's core algorithms for clustering, classification and batch based collaborative filtering are implemented on top of Apache Hadoop using the map/reduce paradigm, it does not restrict contributions to Hadoop based implementations. Contributions that run on a single node or on a non-Hadoop cluster are also welcomed. According to scope of our work we will look into collaborative filtering using Mahout.

Chapter 4

CASE STUDY ON RECOMMENDATION SYSTEM : AMAZON

Recommendation algorithms are best known for their use on e-commerce Websites, where they use input about a customer’s interests to generate a list of recommended items. Many applications use only the items that customers purchase and explicitly rate to represent their interests, but they can also use other attributes, including items viewed, demographic data, subject interests, and favorite artists.

At Amazon.com, they use recommendation algorithms to personalize the online store for each customer. The store radically changes based on customer interests, showing programming titles to a software engineer and baby toys to a new mother. The click-through and conversion rates ,two important measures of Web-based and email advertising effectiveness ,vastly exceed those of untargeted content such as banner advertisements and top-seller lists.

E-commerce recommendation algorithms often operate in a challenging environment. For example:

- A large retailer might have huge amounts of data, tens of millions of customers and millions of distinct catalog items.
- Many applications require the results set to be returned in realtime, in no more than half a second, while still producing high-quality recommendations.
- New customers typically have extremely limited information, based on only a few purchases or product ratings.
- Older customers can have a glut of information, based on thousands of purchases and ratings.
- Customer data is volatile: Each interaction provides valuable customer data, and the algorithm must respond immediately to new information.

4.1 Item-to-Item Collaborative Filtering

Amazon.com uses recommendations as a targeted marketing tool in many email campaigns and on most of its Web sites' pages, including the high traffic Amazon.com homepage. Clicking on the "Your Recommendations" link leads customers to an area where they can filter their recommendations by product line and subject area, rate the recommended products, rate their previous purchases, and see why items are recommended (see Figure 4.2).

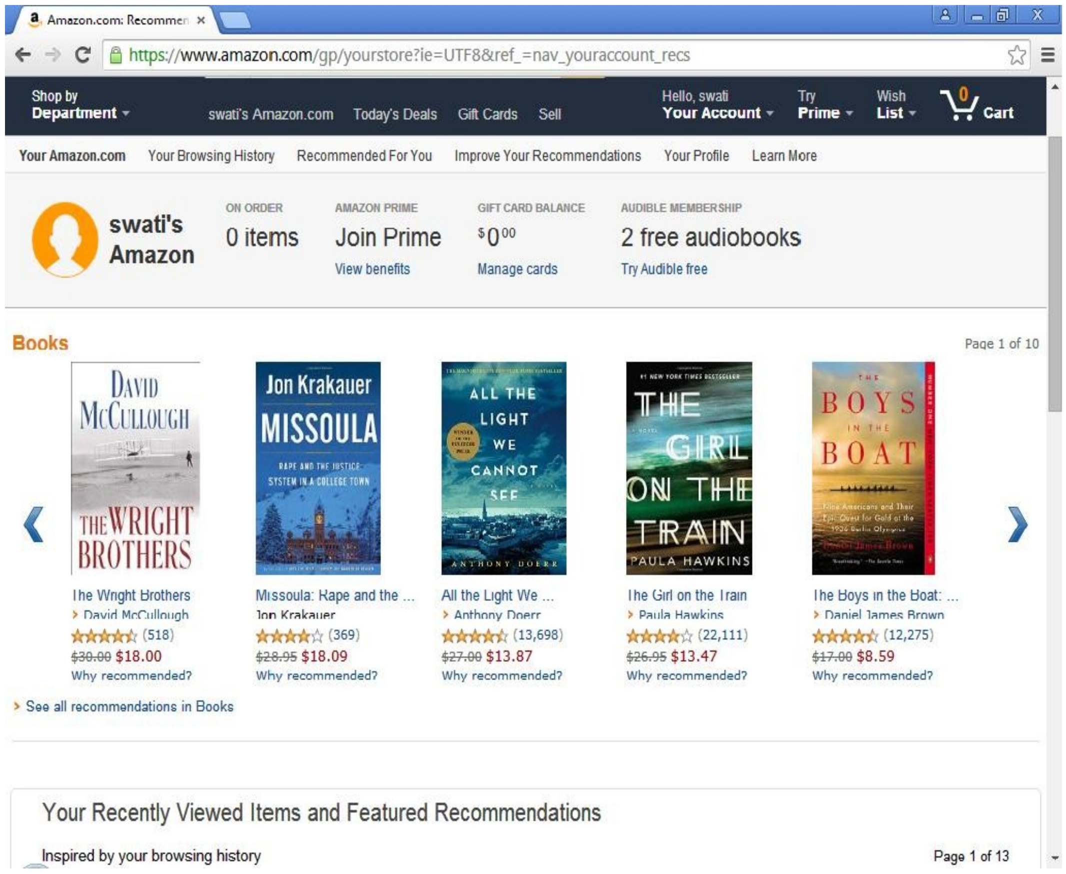
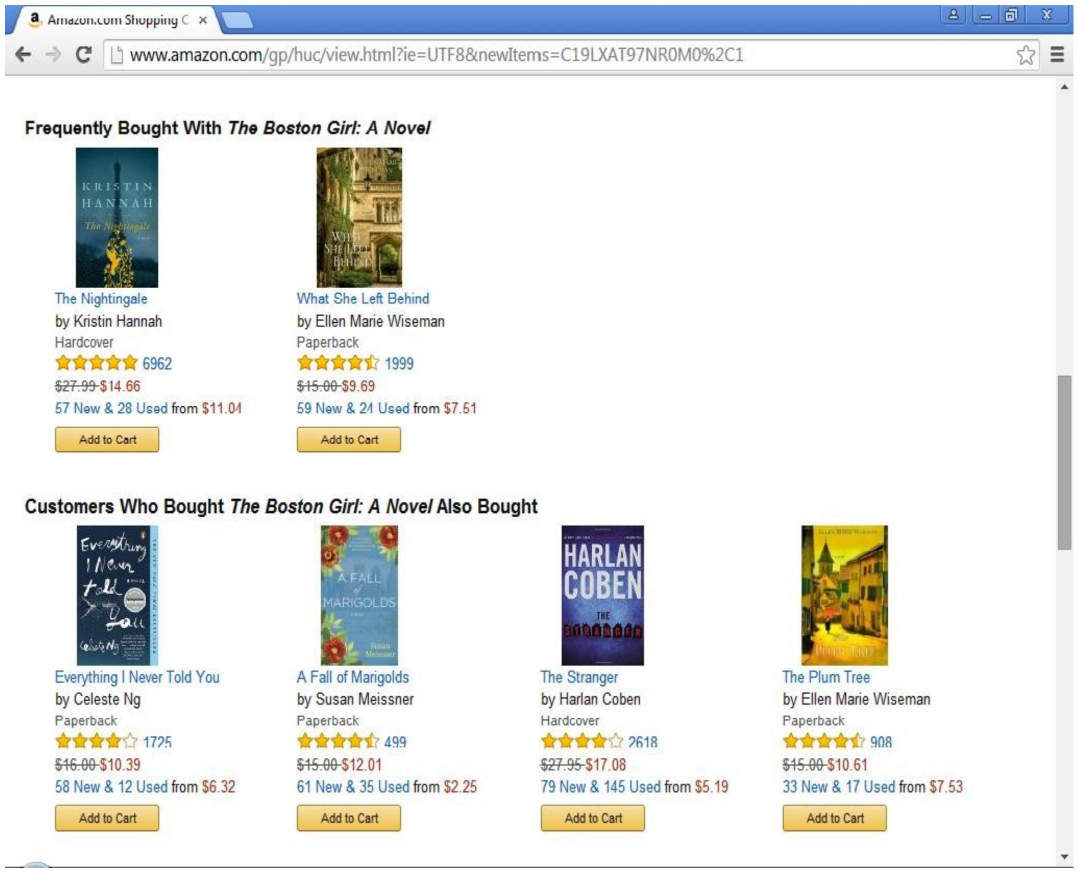

Amazon.com: Recommen
https://www.amazon.com/gp/yourstore?ie=UTF8&ref_=nav_youraccount_recs
Shop by Department
swati's Amazon.com
Today's Deals
Gift Cards
Sell
Hello, swati Your Account
Try Prime
Wish List
0 Cart
Your Amazon.com
Your Browsing History
Recommended For You
Improve Your Recommendations
Your Profile
Learn More
swati's Amazon
ON ORDER 0 items
AMAZON PRIME Join Prime View benefits
GIFT CARD BALANCE $0.00 Manage cards
AUDIBLE MEMBERSHIP 2 free audiobooks Try Audible free
Books
Page 1 of 10
The Wright Brothers › David McCullough (518) $30.00 $18.00 Why recommended?
Missoula: Rape and the ... Jon Krakauer (369) $28.95 $18.09 Why recommended?
All the Light We ... › Anthony Doerr (13,698) $27.00 $13.87 Why recommended?
The Girl on the Train › Paula Hawkins (22,111) $26.95 $13.47 Why recommended?
The Boys in the Boat: ... › Daniel James Brown (12,275) $17.00 $8.59 Why recommended?
› See all recommendations in Books
Your Recently Viewed Items and Featured Recommendations
Inspired by your browsing history
Page 1 of 13

Figure 4.1. The "Your Recommendations" feature on the Amazon.com homepage. Using this feature, customers can sort recommendations and add their own product ratings.

As Figure 4.2 shows, shopping cart recommendations, which offer customers product suggestions based on the items in their shopping cart. The feature is similar to the impulse items in a supermarket checkout line, but our impulse items are targeted to each customer. Amazon.com extensively uses recommendation algorithms to personalize its Web site to each customer's interests. Because existing recommendation algorithms cannot scale to Amazon.com's tens of millions of customers and products, we developed their own, item-to-item collaborative filtering, scales to massive data sets and produces high-quality recommendations in real time.


Amazon.com Shopping C
www.amazon.com/gp/huc/view.html?ie=UTF8&newItems=C19LXAT97NR0M0%2C1

Frequently Bought With *The Boston Girl: A Novel*

The Nightingale
by Kristin Hannah
Hardcover
6962
~~$27.99~~ $14.66
57 New & 28 Used from $11.04
Add to Cart

What She Left Behind
by Ellen Marie Wiseman
Paperback
1999
~~$15.00~~ $9.69
59 New & 24 Used from $7.51
Add to Cart

Customers Who Bought *The Boston Girl: A Novel* Also Bought

Everything I Never Told You
by Celeste Ng
Paperback
1725
~~$16.00~~ $10.39
58 New & 12 Used from $6.32
Add to Cart

A Fall of Marigolds
by Susan Meissner
Paperback
499
~~$15.00~~ $12.01
61 New & 35 Used from $2.25
Add to Cart

The Stranger
by Harlan Coben
Hardcover
2618
~~$27.95~~ $17.08
79 New & 145 Used from $5.19
Add to Cart

The Plum Tree
by Ellen Marie Wiseman
Paperback
908
~~$15.00~~ $10.61
33 New & 17 Used from $7.53
Add to Cart

Figure 4.2. Amazon.com shopping cart recommendations. The recommendations are based on the items in the customer's cart and buying history.

4.2 How It Works

Rather than matching the user to similar customers, item-to-item collaborative filtering matches each of the user's purchased and rated items to similar items, then combines those similar items into a recommendation list. To determine the most-similar match for a given item, the algorithm builds a similar-items table by finding items that customers tend to purchase together. We could build a product-to-product matrix by iterating through all item pairs and computing a similarity metric for each pair. However, many product pairs have no common customers, and thus the approach is inefficient in terms of processing time and memory usage. The following iterative algorithm provides a better approach by calculating the similarity between a single product and all related products:

```
For each item in product catalog, I1
    For each customer C who purchased I1
        For each item I2 purchased by
        customer C
    Record that a customer purchased I1
    and I2
For each item I2
```

Compute the similarity between I1 and I2

It's possible to compute the similarity between two items in various ways, but a common method is to use the cosine measure we described earlier, in which each vector corresponds to an item rather than a customer, and the vector's M dimensions correspond to customers who have purchased that item. This offline computation of the similar-items table is extremely time intensive, with $O(N2M)$ as worst case. In practice, however, it's closer to $O(NM)$, as most customers have very few purchases. Sampling customers who purchase best-selling titles reduces runtime even further, with little reduction in quality.

Given a similar-items table, the algorithm finds items similar to each of the user's purchases and ratings, aggregates those items, and then recommends the most popular or correlated items. This computation is very quick, depending only on the number of items the user purchased or rated.

4.3 Scalability: A Comparison

Amazon.com has more than 29 million customers and several million catalog items. Other major retailers have comparably large data sources. While all this data offers opportunity, it's also a curse, breaking the backs of algorithms designed for data sets three orders of magnitude smaller. For very large data sets, a scalable recommendation algorithm must perform the most expensive calculations offline. As a brief comparison shows, existing methods fall short:

- Traditional collaborative filtering does little or no offline computation, and its online computation scales with the number of customers and catalogue items. The algorithm is impractical on large data sets, unless it uses dimensionality reduction, sampling, or partitioning — all of which reduce recommendation quality.
- Cluster models can perform much of the computation offline, but recommendation quality is relatively poor. To improve it, it's possible to increase the number of segments, but this makes the online user–segment classification expensive.
- Search-based models build keyword, category, and author indexes offline, but fail to provide recommendations with interesting, targeted titles. They also scale poorly for customers with numerous purchases and ratings.

The key to item-to-item collaborative filtering's scalability and performance is that it creates the expensive similar-items table offline. The algorithm's online component, looking up similar items for the user's purchases and ratings, scales independently of the catalog size or the total number of customers; it is dependent only on how many titles the user has purchased or rated. Thus, the algorithm is fast even for extremely large data sets. Because the algorithm recommends highly correlated similar items, recommendation quality is excellent. Unlike traditional collaborative filtering, the algorithm also performs well with limited user data, producing high-quality recommendations based on as few as two or three items.

Chapter 5

IMPLEMENTATION

5.1 Proposed work

For recommendation, our proposed system uses collaborative filtering machine learning algorithm. Collaborative filtering (CF) is a machine learning algorithm which is widely used for recommendation purpose. Collaborative filtering finds nearest neighbor based on the similarities. The metric of collaborative filtering is the rating given by the user on a particular item. Different users give different ratings to items. Users, who give almost same rating to items, are the nearest neighbors. In case of User based collaborative filtering, based on the ratings given by the users, nearest neighbors has been find. Item based collaborative filtering predicts the similarity among items. To recommend an item, items which are liked by the user in his past have been found. Item which is similar to those items has been recommended [5].

Internet contains a huge volume of data for recommendation purpose. Due to size of data, if recommendation computation has been done in single system, then performance may degrade, and we cannot find an efficient solution. Hence we require distributed environment so that computation can be increased and performance of recommendation system gets improve. An open source cloud environment Hadoop provides distributed environment [3]. Due to Map-Reduce programming, it provides result efficiently and effectively in less amount of time [2]. Proposed system has been modeled on Hadoop. Mahout is an open source java library which favors Collaborative Filtering. Mahout favors Hadoop for recommendation.

This work is focusing to customize a recommendation system using Collaborative Filtering (CF) and clustering techniques. For our approach, we use Apache Hadoop (a widely used open source platform which implements Map-Reduce programming model) and Apache Mahout (An Open source library of scalable data mining algorithms, where it forms a core of the distributed recommender module). The proposed algorithm is worked in two parts as shown in figure 5.1. In first phase, we obtain the results for recommendation by applying User-based CF and Item-based CF separately. In second phase, we combine the results obtain from user-based CF and item-based CF.

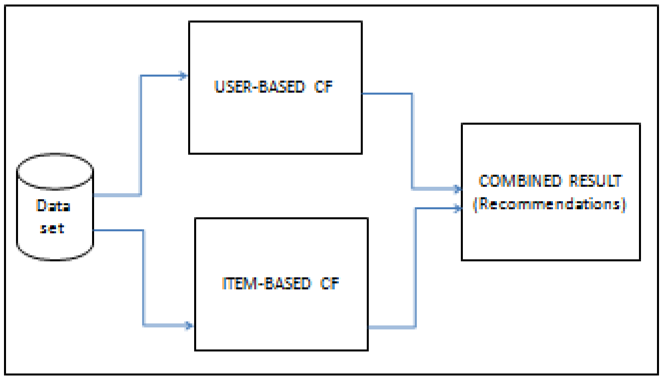


Figure 5.1: Combined approach for CF

5.2 User Based CF

The dataset is firstly loaded into Hadoop distributed file system (HDFS). Then we perform User-based CF using Mahout. We take rating matrix, in which each row represents user and column represents item, corresponding row-column value represents rating which is given by a user to an item. Absence of rating value indicates that user has not rated the item yet.

There are many similarity measurement methods to compute nearest neighbors. We have used Pearson correlation coefficient to find similarity between two users. Hadoop is used to calculate the similarity. The output of the Hadoop Map phase i.e. userid and corresponding itemid are passed to reduce phase. In reduce phase, output has been generated and sorted according to userid. Output again has been stored in HDFS. The architecture diagram for User-based CF can be shown in figure 5.2.

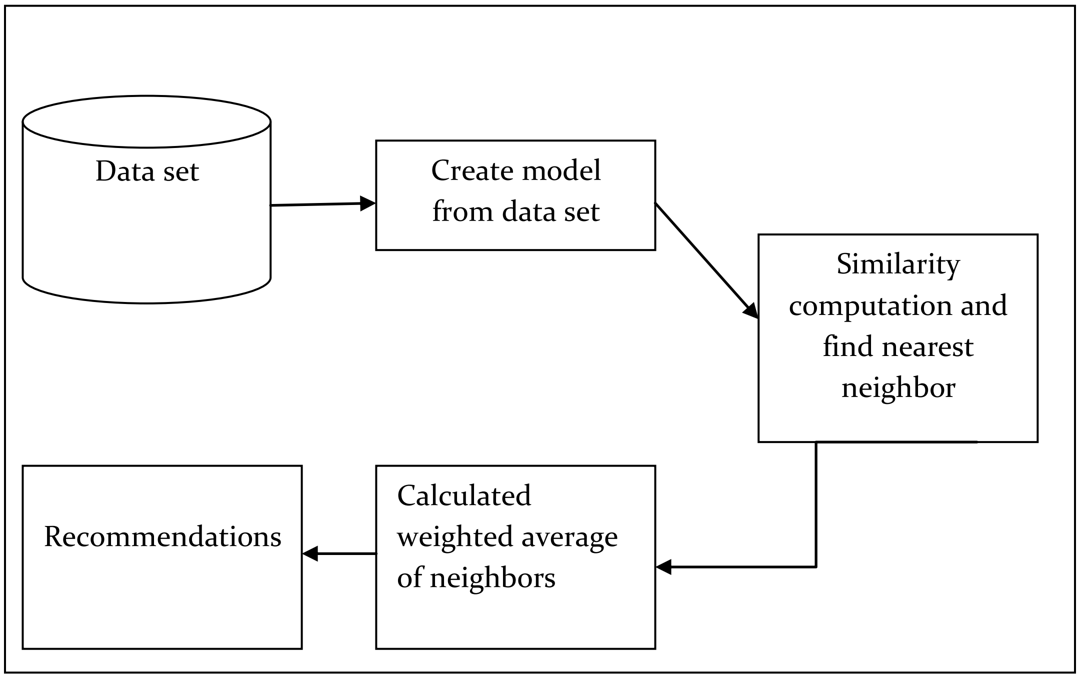

Data set
Create model from data set
Similarity computation and find nearest neighbor
Calculated weighted average of neighbors
Recommendations

Figure 5.2: User-based Collaborative Filtering

5.3 Item Based CF

Dataset is loaded into HDFS, then using Mahout we performs Item-based CF. Past information of the user, i.e. the ratings they gave to items are collected. With the help of this information the similarities between items are build and inserted into item to item matrix. Algorithm selects items which are most similar to the items rated by the user in past. In next step, based on top-N recommendation, target items are selected. The processing of Item-based CF can be described through figure 5.3.

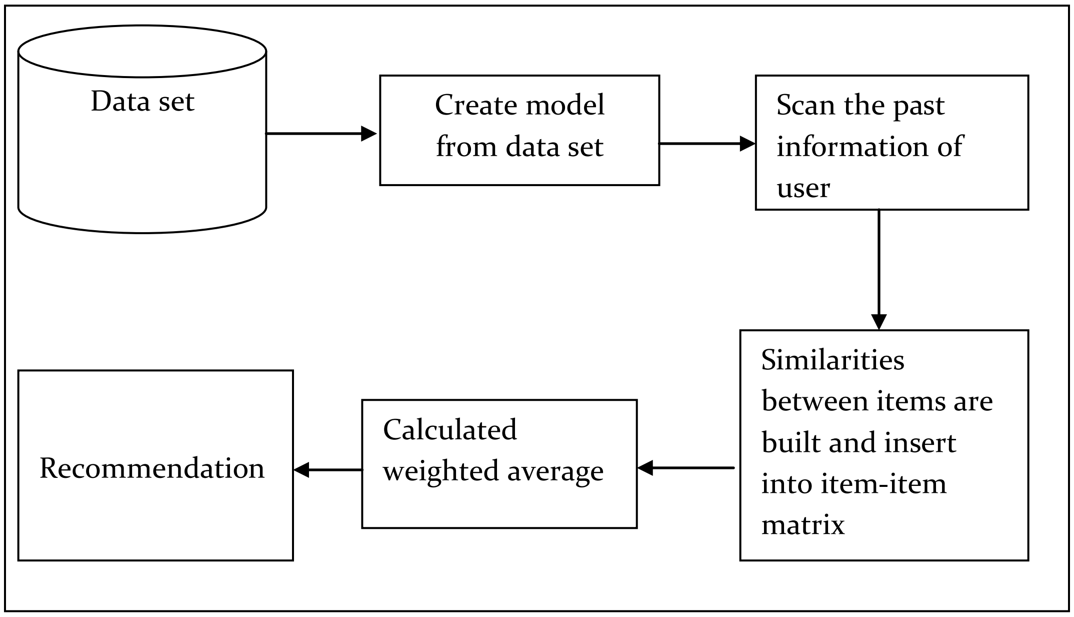

Data set
Create model from data set
Scan the past information of user
Similarities between items are built and insert into item-item matrix
Calculated weighted average
Recommendation

Figure5.3: Item-based Collaborative Filtering

5.4 Combined approach

In case of user-based CF, if nearest neighbours (similar users) are not in enough number, i.e. taste of target user is not similar to many users then the recommending an item for that particular user may be not accurate. Item-based CF is based on the past information of the user, so it works well in such cases. The user-based and item-based results that are stored in HDFS are taken and then we combine these results based on threshold value. Suppose the threshold by increasing threshold value, probability of recommending correct item get increases because it has been liked by many users. Flow can be seen in figure 5.4.

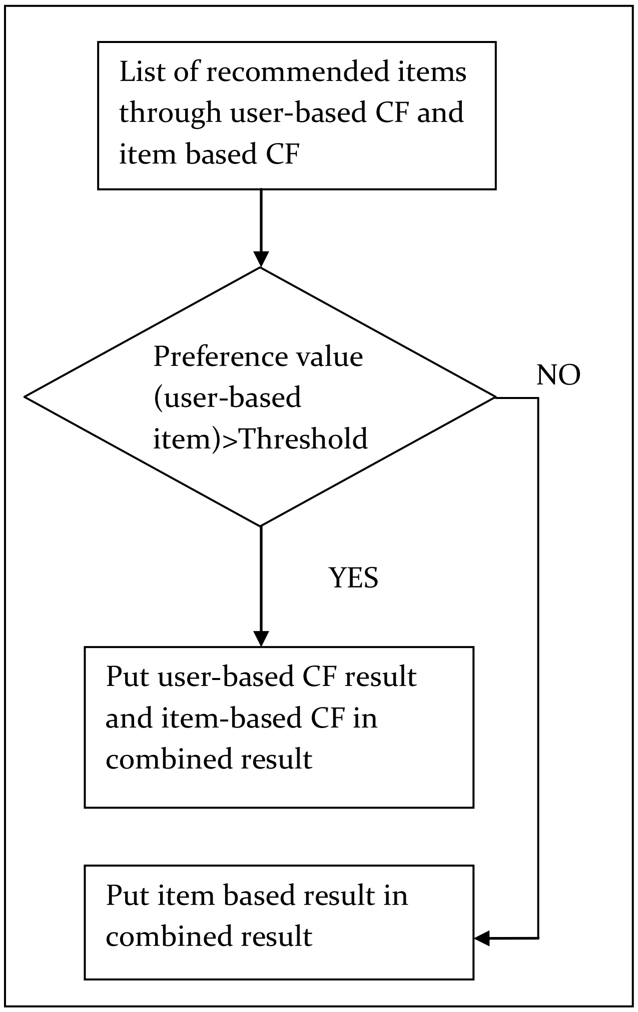

List of recommended items through user-based CF and item based CF
Preference value (user-based item)>Threshold
NO
YES
Put user-based CF result and item-based CF in combined result
Put item based result in combined result

Figure 5.4 : Combined approach for collaborative filtering

5.5 System Configuration

We implemented user-user and item based collaborative algorithm to get the recommendations on Haoop platform using Apache Mahout. Our Hadoop cluster is made up of four nodes out of which one is master and other are slave nodes.

Processor	2.00 GHz Intel Dual Core
RAM	3 GB
Operating System	Linux Ubuntu 14.04 LTS
Java	JRE 1.7
Hadoop	Apache Hadoop 2.6.0
Mahout	Apache Mahout 0.9
Dataset	Movie lens dataset

Table – 1: System Configuration

Chapter 6

RESULTS

6.1 Result analysis

For experiment we have used stable benchmark dataset with 1,000,209 ratings of approximately 3,900 movies made by 6,040 MovieLens users. In this experiment we are comparing speedup of system with increasing numbers of nodes and increase of recommendation accuracy with increase in number of users. Speedup is given by the ratio of execution time of one processor and execution time with increasing number of nodes. Given by

$$\text{Speedup} = T(1)/T(n)$$

Where n = no. of nodes and T is execution time. T(1) represents times taken by single node and T(n) represents time taken by n numbers of nodes.

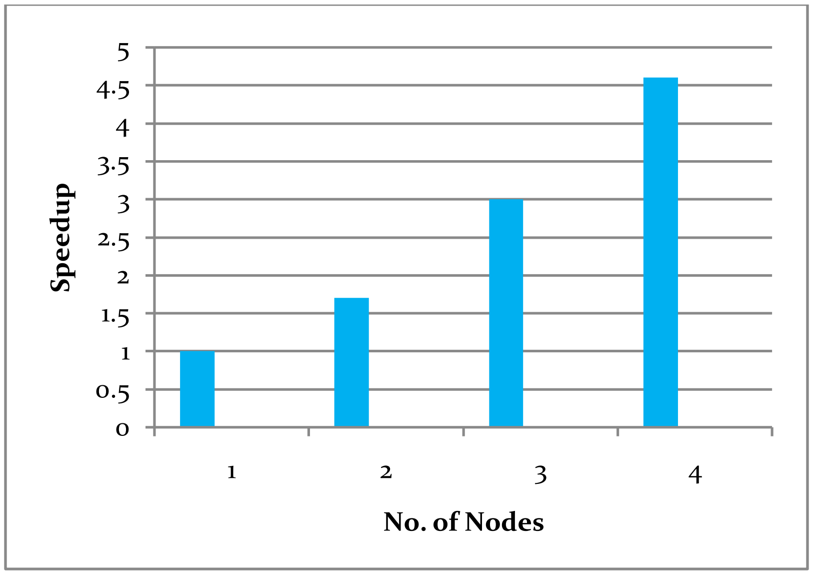


Figure 6.1 : Speedup with respect to no. of nodes

As we can see in figure 6.1, we have taken four node cluster. When we increased the number of nodes one to four and with increasing number of nodes in Hadoop cluster, speedup also increases. So this algorithm is scaling well with Hadoop platform.

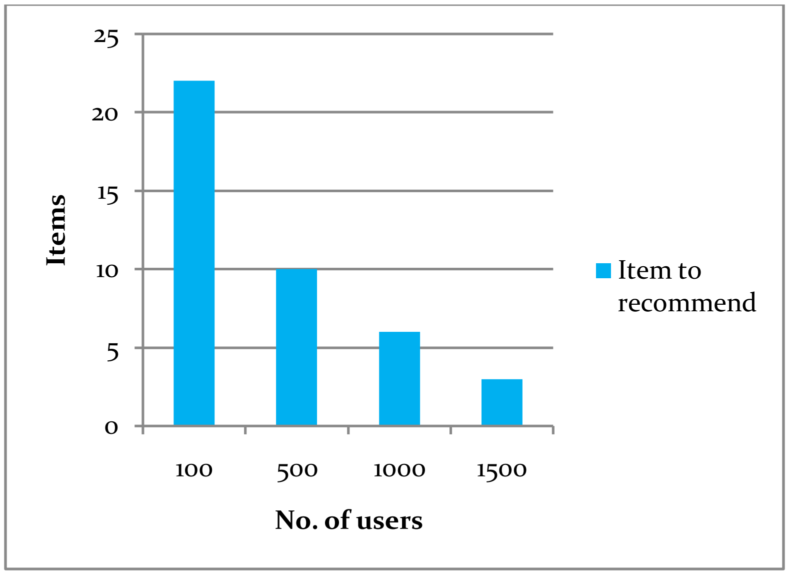


Figure 6.2: Accuracy of recommendation

Next we have made four sets of data by taking different numbers of users as 100 users, 500 users, 100 users and 1500 users. As figure 6.2 shows the accuracy of recommendation has increased with respect to the increase in number of users who like an item because with increase in number of users the number of resulted recommendations are decreasing which shows less recommendations but with relevant result.

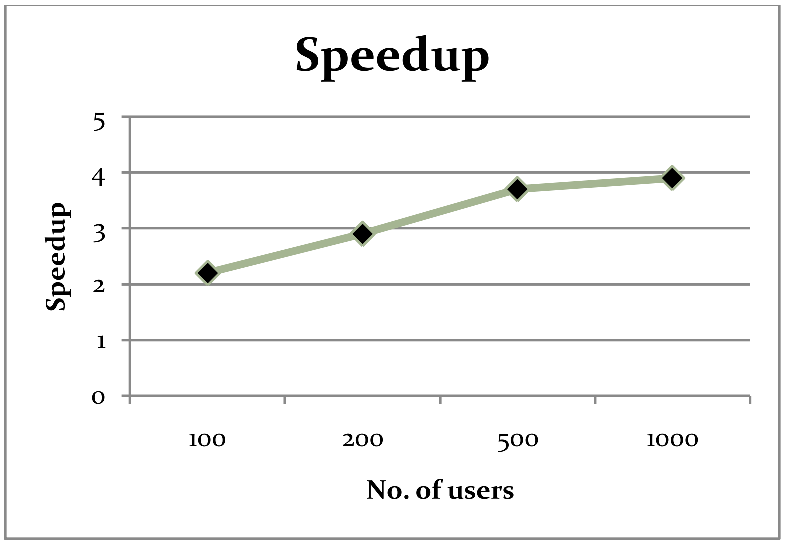


Figure 6.3: Speedup with respect to increase in dataset.

Next we fixed the number of node to four and made 4 sets of users data. As shown in figure 5. 7, we have 100 users, 200 users, 500 users and 1000 users. We can see that as number of user increased in dataset the speed has also increased which shows that more the dataset more the speedup. We can see that to achieve the best performance by Hadoop we need to provide more data.

Chapter 7

CONCLUSION AND FUTURE WORK

7.1 Conclusion

In this thesis, a combined approach of user-user CF and item-item CF has been presented to generate recommendations on Hadoop cluster using Apache Mahout, a library for machine learning algorithms. By using combined approach, accuracy of recommendation has improved. This approach has scaled well with the hadoop platform. Time needed to solve the problem has reduced. Mahout is able to handle big data but it still lack some algorithms. The recommendation for single user need to be improved for better results.

7.2 Future work

Future of research in the area of Big Data and Recommendation systems has vast scope. New computing platforms like Apache Spark are getting prominent in the field of Big Data analysis. Recommendation algorithms can perform better on such platforms for faster performance. Collaborative filtering can be implemented using Apache Spark and its performance can be compared with performance of collaborative filtering on Hadoop.

REFERENCES

[1] Apache Hadoop, *https://hadoop.apache.org/.*

[2] Apache Mahout, *http://mahout.apache.org/.*

[3] Movielens Dataset, *http://grouplens.org/datasets/movielens/.*

[4] D. Hee Park et al., "A literature review and classification of recommender systems research," Elsevier *Expert Systems with Applications,* vol. 39, 10059–10072, 2012. Doi:10.1016/j.comcom.2013.06.009

[5] X. Yang et al., "A survey of collaborative filtering based social recommender systems", Elsevier *Computer Communications,* vol. 41, 1–10, 2014. Doi:10.1016/j.comcom.2013.06.009.

[6] G. Linden et al., "Amazon.com recommendations: Item-to-item collaborative filtering," *IEEE Internet Computing*, vol. 7, no. 1,pp. 76–80, 2003. Doi: 10.1109/MIC.2003.1167344.

[7] Ruzhi Xu et al., "Distributed collaborative filtering with singular ratings for large scale recommendation", Elsevier *The Journal of Systems and Software* , vol. 95 , 231–241, 2014. Doi:10.1016/j.jss.2014.04.045.

[8] J. Bobadilla et al. "Recommender systems survey", Elsevier, *Knowledge-Based Systems,* vol. 46, 109–132, july 2013. doi:10.1016/j.knosys.2013.03.012.

[9] Saikat Bagchi, "Performance and Quality Assessment of Similarity Measures in Collaborative Filtering Using Mahout", *2nd International Symposium on Big Data and Cloud Computing (ISBCC'15)*, Procedia Computer Science vol. 50, 229 – 234, 2015. Doi:10.1016/j.procs.2015.04.055.

[10] Xiwei Wang et al., ”A Case Study of Recommendation Algorithms”, IEEE *2011 International Conference on Computational and Information Sciences* (ICCIS). Doi: 10.1109/ICCIS.2011.20.

[11] Juha Leino and Kari-Jouko Räihä, “Case Amazon: Ratings and Review as Part of Recommendations”, ACM Press, *RecSys'07*, October 19–20, 2007. Doi: 10.1145/1297231.1297255.

[12] Carlos E. Seminario and David C. Wilson, “Case Study of Mahout as a Recommender Platform” ACM *RecSys 2012.*

[13] Xing Xie, “Potential Friend Recommendations in Online Social Network” *IEEE/ACM International Conference on Green Computing and Communications,* 2010. Doi: 10.1109/GreenCom-CPSCom.2010.28.

[14] G. Adomavicius and A. Tuzhilin. “Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions”, *IEEE Transactions on Knowledge and Data Engineering,* vol. 17, no. 6, June 2005. Doi: 10.1109/TKDE.2005.99.

[15] Ricci F. Et al., “Introduction to Recommender Systems Handbook” (http://www.inf.unibz.it /~ricci/papers/intro-rec-syshandbook.pdf), *Recommender Systems Handbook*, Springer, 2011.

[16] Owen S. Et al., “Mahout In Action”, *Manning Publications* Co. ISBN 978-1-9351-8268-9, 2012.

[17] Wu Yueping and Zheng Jianguo, “A research of recommendation algorithm based on cloud model”, *International Conference on Intelligent Computing and Intelligent Systems (ICIS),* IEEE 2010. Doi: 10.1109/ICICISYS.2010.5658595.

[18] Yanhong Guo et al., “An improved collaborative filtering algorithm based on trust in e-commerce recommendation system”, *International Conference on Management and Service Science (MASS)* IEEE Aug 2010. Doi: 10.1109/ICMSS.2010.5576688.

[19] B.M. Sarwar et al., “Item-item Collaborative Filtering Recommendation Algorithms,” ACM Press, *10th Int’l World Wide Web Conference*, , pp. 285-295, 2001. Doi: 10.1145/371920.372071.

[20] B.M. Sarwarm et al., “Analysis of Recommendation Algorithms for E-Commerce,” *ACM Conf. Electronic Commerce*, pp.158-167, 2000. Doi: 10.1145/352871.352887.

[21] Jeffrey Dean and Sanjay Ghemawat, “Map-Reduce: Simplified data processing on large clusters”, appeared in *OSDI* 2004.

[22] “The Emerging Big Returns on Big Data” *TCS Big Data Global Trend Study 2013*.