

A Prototype Implementation of Recommendation Engine Using Big Data Analytics

GRADUATE PROJECT REPORT

Submitted to the Faculty of
The School of Engineering & Computing Sciences
Texas A&M University-Corpus Christi
Corpus Christi, TX

In Partial Fulfillment of the Requirements for the Degree of
Master of Science in Computer Science

by

Satya Narayana Murthy Kota
Spring 2015

Committee Members

Dr. Dulal C Kar
Committee Chairperson

Dr. Longzhuang Li
Committee Member

ABSTRACT

While Big Data is revolutionizing the IT world by solving large dataset problems with performance and scalability, software firms in many sectors are not ready to shift to a wholly Big Data project. The reason being the line of business involves atomic transactions that are well supported by relational databases. Hence, to leverage the benefits of both technologies, solutions that could drive the business are solved using big data analytics.

Accordingly, an architecture is proposed in this paper which could handle Big Data content. This platform architecture discovers useful content easily with different data mining algorithms such as prediction algorithm and classification algorithms. Concluding these algorithms with reports will form a business intelligence suite that will improve the business value.

In this project, the use case of online book store is developed. The users' behavioral data is collected in the format of a text file that is mined and sorted with projects in the Big Data environment to provide recommendations to the user. These recommendations help the user find the right product and sales for the e-commerce platform.

Keywords: Big data, data mining algorithms, prediction, classification.

TABLE OF CONTENTS

Abstract.....	ii
Table of Contents.....	iii
List of Figures.....	v
1. Background and Rationale.....	1
1.1 Introduction to Recommendation Engine using Big Data Analytics.....	2
1.2 Existing Applications.....	3
1.3 Proposed Solution	3
1.4 Technologies.....	5
1.4.1 Java.....	5
Java Server Pages	5
Servlet	6
JDBC	6
1.4.2 MySQL server.....	6
1.4.3 Hadoop Ecosystem.....	6
2. Narrative.....	11
2.1. Problem Statement.....	11
2.2. Motivation.....	11

2.3. Product Modules Description.....	11
2.4. Product Scope.....	10
3. System Design and Architecture.....	15
3.1. Proposed System Architecture.....	15
3.2 Proposed System Design.....	17
3.3 System Requirements.....	19
3.3.1 Functional Requirements.....	19
3.3.2 Software Requirements.....	20
3.3.3 Hardware Requirements.....	20
4. System Implementation	21
4.1 Environment.....	21
4.1.1 Eclipse.....	21
4.1.2 Apache tomcat.....	22
4.1.3 Code organization.....	22
4.2 User Interface.....	22
4.2.1 Admin Web User Interface.....	23
4.2.2 Command User Interface for Admin.....	24

4.2.3 User Interface for User Module.....	26
4.3 Data Mining Algorithm for recommendations to Search Results.....	28
4.3.1 Log-Likelihood Algorithm.....	30
5. Testing and Evaluation.....	34
5.1 Test Cases.....	34
5.2 Evaluation of Recommendations.....	36
5.3 Usability Testing.....	37
6. Conclusion and Future Work.....	31
7. Bibliography and References.....	32

LIST OF FIGURES

Fig 1.1: 4 V's of Big Data.....	2
Fig 1.2: Big Data Processing Framework.....	4
Fig 1.3: Hadoop Ecosystem.....	9
Fig 3.1: System Architecture.....	15
Fig 3.2: System design Flow chart for Admin.....	18
Fig 3.3 System Design Flowchart for User.....	19
Fig 4.1 Admin Home	23
Fig 4.2 Admin adds books into RDBMS	23
Fig 4.3 Admin runs queries to perform processing.....	24
Fig 4.4 Code for Admin Script.....	25
Fig 4.5 Tables exported from to MySQL database	26
Fig 4.6 Home Page of the web app.....	27
Fig 4.7 User Login page.....	27
Fig 4.8 User Home Page.....	28
Fig 4.9 Top rated Unseen books	28
Fig 4.10 User Ratings	29

Fig 4.11 Ratings Stored in a Text File	29
Fig 4.12 Search Functionality.....	30
Fig 4.13 Search Result from Database Along with Recommendations.....	31
Fig 4.14 Code Block for Fetching Recommendations	32
Fig 5.1 Test Case 1.....	34
Fig 5.2 Test Case 2.....	35
Fig 5.3 Test Case 3.....	36
Fig 5.4 Evaluator Code.....	37

1. Background And Rationale

Big data can solve fundamental challenges that the new era of data world is presenting in an efficient manner. The challenges include the four fundamental V's [1]:

- **Volume:** With all the data in the world growing two times in every two years, there is a need for an efficient and scalable platform to serve the large datasets that are going to be produced in future as well.
- **Variety:** The database system should be able to store and process of a variety of structured, unstructured or semi-structured data in any format (Examples: XML, JSON, CSV, JPEG, PDF).
- **Veracity:** Gone are the days, where few business executives take market decisions. The truthfulness of the decisions made by analytics depends upon the veracity of the data, which ultimately drives business.
- **Velocity:** The speed at which data is growing is tremendous and there lies a need for organizations to predict future's data.

As shown in Fig 1.1 below, IBM's study showcases the definition and facts of Big Data. Big Data is characterized by Heterogeneous, Autonomous, Complex and Evolving associations in HACE theorem [2]. Data mining analysis on Big Data should be a data-driven model involving user interest modeling, aggregation of various data sources based on demand, applying mining and analysis [2].

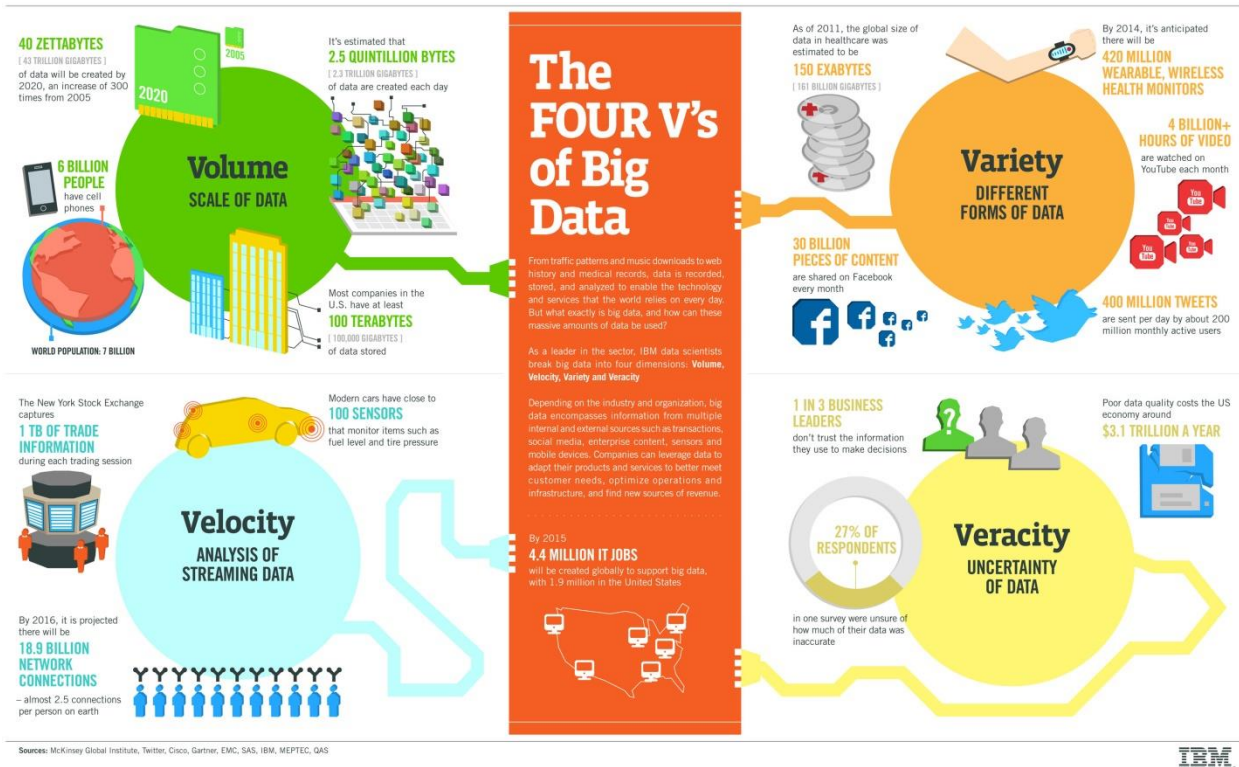


Fig 1.1 4 V's of Big Data

1.1 Introduction to Recommendation Engine using Big Data Analytics

Big Data analytics has revolutionized the analytics world by producing results that a traditional Business Intelligence platform could not have achieved by leveraging on the cheap commodity hardware and distributed computing. On the other hand, Recommendation Engines have produced much business in e-commerce platforms. Therefore, there is a need for a reliable [2] recommendation engine that can understand the behavior of the item in the market. It is potentially a Big Data problem since it involves mining large and variety of datasets. Hence, using Big Data Analytics for building a Recommendation Engine helps to induce business from the customers in efficient and economical way.

The project deals with the use case of an online bookstore, where the user is displayed recommendations with the product searched. These recommendations are made using Big Data Analytics platform namely Hadoop. Mining of data is done through the ratings provided by the user.

1.2 Existing Applications

Traditional Business Intelligence platforms can work with Relational data and can produce better results. However, in a platform to work with text files and cross-platform files, existing relational warehouses encounter problems in processing data. Applying existing data mining algorithms and techniques to real-world problems has been recently running into many challenges due to the inadequate scalability [2]. Not only the scale of data generated today is unprecedented, the produced data is often continuously made in the form of streams that require being processed and mined in (nearly) real time [2]. Delayed discovery of even highly valuable knowledge invalidates the usefulness of the discovered knowledge.

1.3 Proposed Solution

Big Data technologies can process a vast variety of behavioral data which involves data from multiple sources. Fig 1.2 displays a view of Big Data processing framework [2], which involves:

- **Big Data Mining Platform**

In order to implement data processing in Big Data architecture, there should be a mining platform that supports different mining algorithms to be distributed (i.e. ability to run data mining algorithms in parallel). The algorithms operate as Map Reduce jobs across systems. However, this project is run on the standalone system, which could be replicated across a cluster of machines.

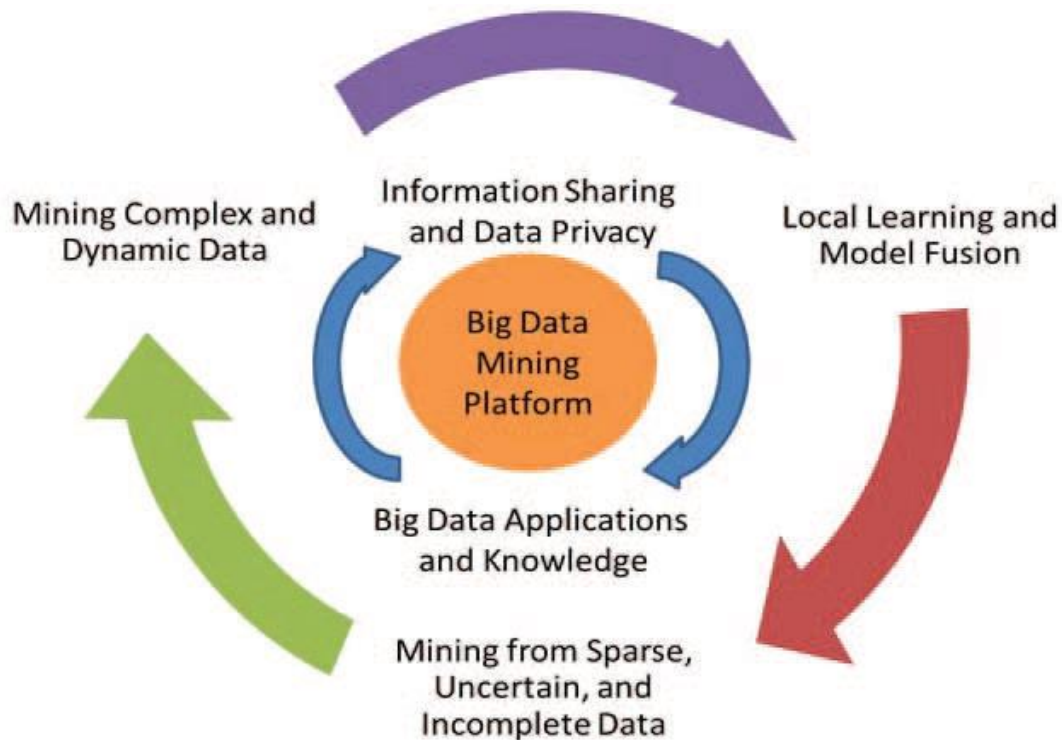


Fig 1.2 Big Data Processing Framework [2]

- Big Data Semantics and Application Knowledge

Possessing the domain knowledge helps to understand the type of data coming in with the help of semantics [3]. The projects or utilities in Big Data platform for example in Apache Hadoop Ecosystem helps to solve problems or provide solutions of business use-case.

- Big Data Mining Algorithms

The algorithms for data mining are developed by a group of enthusiastic mathematicians that are grouped together into an API. There are some open source data mining libraries that could be used in Big Data Platform to run them across the distributed systems in the big data cluster to make an analysis.

Applying the above framework, the potential problem of running data mining tasks on large and a variety of datasets can be achieved.

1.4 Technologies used

Open Source platforms have been chosen to implement this project. They are Java, MySQL Database Server, and Apache Hadoop ecosystem.

1.4.1 Java

Java is an open source object-oriented programming language originally owned by Sun Micro Systems. However, Oracle procured Sun Micro Systems later on [4]. Java is everywhere right from microwave oven to Enterprise applications. The reason being its acceptance is its substantial compliance with Object oriented paradigm. It has grown into multiple frameworks to manage the projects in industry. The API of Java is available in the form of Java Development Kit for developers as SE (Standard Edition) for standalone Java applications. It is available as Java EE (Java Enterprise Edition) for running enterprise web applications. With the source code portability, unlike Microsoft.Net, Java has gained immense importance in enterprise applications. There are versions of Java Development Kit from various vendors like IBM JDK, OpenJDK, and Oracle. In this project, Oracle JDK version 1.6 is used.

JSP (Java Server Pages):

Java Server Page is meant to render the client side dynamic web application by executing on the server. The fundamental difference between HTML and JSP is JSP's ability to maintain state. Internally, a JSP will be converted into a Servlet, which is executed on the server. Java code can also written JSP's but it is not mandated as per Industry Standards. The JSP Standard Tag

Library provides a framework for tags that offer core functionality for web applications. It is implemented using the taglib directive `<%@ taglib ... %>`. JSP tags help work on the Java collection objects with the help of ‘scriptlet’ tags.

In addition to the taglib directives and ability to maintain a session, JSP offers implicit objects such as request, out, response, Page Context, Exception. These implicit objects help in getting data from the session. The data in the session help the pages to be active while moving from one page to another.

Servlet:

The servlet is Java class that can procure HTTP connection and run on the server side. The servlet is the where most of the action takes place in a web application. Since Servlet’s execution time on the server is less, as, per coding standards, it is always advised to write Java code in Servlet.

JDBC:

Java Database Connectivity is the utility provided by Java API to connect to database management systems such as Oracle, MySQL, and DB2. The database connection is achieved by using drivers that talk to the database.

1.4.2 MySQL server

MySQL Server is an open source relational database management system. It functions like any other relational database and executes SQL queries. In transactional data, MySQL server is an excellent fit for any other relational database system. MySQL Workbench is one more solution offered by MySQL community on top of the MySQL server. It helps the administrators to visualize

the schema using graphical user interface to design, develop, administer and even migrate the database [6].

1.4.3 Hadoop Ecosystem

Apache Hadoop is an ecosystem developed by Apache Software Foundation to solve Big Data challenges. Apache Hadoop evolved from Map-Reduce paradigm taking inspiration from Google, which officially announced their Map-Reduce work and its database Big Table in a white paper [7]. Apache Hadoop is not just a framework. In fact, it is an ecosystem of projects that are aimed at solving a single corresponding challenge of Big Data. Apache Hadoop leverages on cheap commodity hardware and distributed computing rather than using server grade machines.

The core components that come along with Apache Hadoop distribution 1.0 are HDFS (Hadoop Distributed File System) and Map Reduce. Hadoop gains its importance with the resilient architecture of the file system and ability to incorporate parallel processing using the Map Reduce. The Map-Reduce programs could be developed using Java/Python, later back in 2009 using a utility provided by Hadoop called Hadoop Streaming. It was a very naïve platform for data mining then since developers have to write the code for solving each type of business problem, when there were traditional Business Intelligence platforms are offering solutions. Business were skeptical about shifting to a whole new platform, whose results are not fit since they have to invest in manpower of developers, who had sufficient knowledge of the platform.

Consequently, Apache Hadoop Community released a stable version of 1.2.1 in August 2013 [9]. At this point, few IT firms started slowly migrating their data from relational warehouses to Hadoop Distributed File System for data analysis. There are two reasons for this. Primarily, Apache Hadoop Ecosystem's stack has then grown to mature platform to solve Big Data platforms

with less effort. The beauty of the projects in the stack is that they understand how to work on different file systems present in HDFS. Secondly, vendors for Hadoop have also grown to level, where in they can provide industry-cut solutions to IT firms. The solutions offered by the vendors are providing clusters that could be rented to provide analysis to the clients. They also provide support for developing, deploying and administering applications in their clusters. Cloudera and Hortonworks are the major vendors in the market for Hadoop Distributions [10].

However, in the later version, next generation Map Reduce (MR2 or Yet Another Resource Negotiator) redesigned Map Reduce architecture to provide enhanced availability and reliability [11]. With the advent of this new design and new projects in Hadoop stack, a new-age Database developer needs to code his project with programming languages such java, python or scala. With the project's developer strength in programming languages and programmers' ease of working with scripting languages or querying platforms, a Hadoop Architect of the use-case is designing the work involving those projects in Hadoop Ecosystem which includes Map Reduce programming or tools that in turn generate map-reduce jobs. Fig 1.3 describes the Hadoop stack implementation in Horton Works platform.

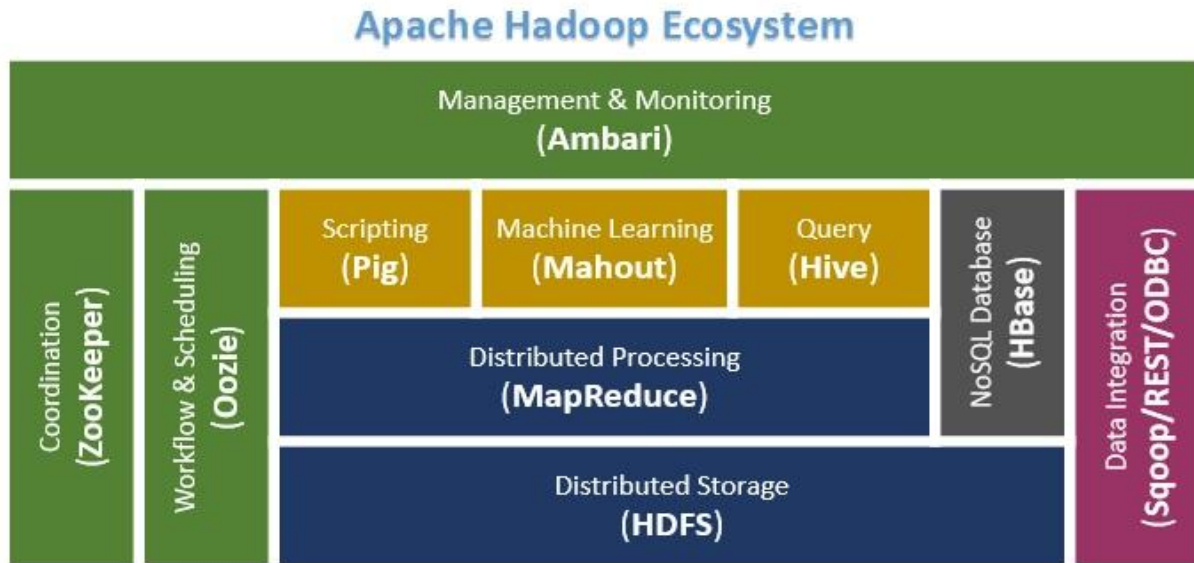


Fig 1.3 Hadoop Ecosystem [12]

The following summarizes the functions of stack in Hadoop Ecosystem:

- SQOOP is a data integration tool for fetching data from the relational database into Hadoop Distributed File System. It connects to a database using the JDBC connection and basing on the requirement, imports or exports data into Hadoop Distributed File System.
- HBase is NoSQL database aimed for storing wide columnar data.
- Hive is an extensive data processing and querying command line utility. It contains metastore, data about data which is stored in Hadoop Distributed File System. The beauty of Hive project is its ability to execute SQL queries along with few modifications in Hive query language. All the queries that are executed by Hive will be converted into map-reduce jobs that are run across the data nodes in Hadoop cluster. Hive performs better with large datasets and suffers from latency of phases with small datasets.
- Pig is scripting utility that facilitates programmer by converting those scripts into Map Reduce jobs. Database developers who have the ease in writing scripts prefer pig latin instead of writing map reducing code using java.

- Apache Mahout is machine learning library used to perform data mining tasks in Hadoop distributed environment. Apart from the distributed mode, the mining platform is also available in the standalone mode. It has algorithms that could be run on different engines [13]. It offers an API or library files which can be used on run upon the requirement such as User-based collaborative filtering [13], Item-based collaborative filtering and clustering algorithms for pertaining to particular use-case. Developers need to either build an apache maven project or download the libraries to add to the build path. They need to implement the algorithm that solves the use-case problem by designing it to their application.
- There are monitoring, scheduling and coordination tools for efficient utilization of the cluster such as Ambari, Hue or Oozie, and Zookeeper respectively.

The ecosystem will only get better with time by including projects that ease the solutions to the problems posed by big data.

2. Narrative

2.1 Problem Statement

The use case of the project is an online bookstore running with a relational database. The project aims to implement an architecture to do analysis on products in Big Data domain while transactions still happen in RDBMS. To start with describing the use-case, a user can logon to the main web page, where he/she is shown the top rated books of the online store to make him/her interested in using the online web application of the store. The user of the system can register to the system and login with the authentication procedure. Once the user logs in, he will be displayed a hyperlink to see his unseen books of store. The user can search the products he/she is interested in, where he/she is presented the recommended books basing on the mining analysis done on the books. The mining analysis is done using Item-based collaborative filtering to find the similar book that stores back the recommended books back in Relational Database in turn.

2.2 Motivation

In the market, there are recommendation engine products from various vendors. There are solutions from traditional database systems also. They can interact with different tables of a same database or similar kind of database (RDMS) to give recommendations. There are two problems with the Business Intelligence or data mining solutions offered by the traditional systems. Firstly, they cannot handle data from a variety of sources whose schema is unknown. Secondly, these systems do not possess sophisticated query processing to handle recommendations. Therefore, if we build a recommendation engine using Big Data Analytics platform, we can improve recommendations to drive business. The analysis of data gives the best results when data is collected from various sources.

2.3 Product Modules Description

- **User module**

The user can logon to the home page of the system. User is shown the top rated books as recommendation for him/her. While a user clicks on any book or the link saying ‘you may like these products’, user is redirected to page where he can login to the system. If the user does not have an account, he/she can register into the system by creating an account. Once user logs into the system using the username and password, the user is given a link to see unseen books of the user generated by the system. The user can enter the book name in the search bar, in which he/she is interested in seeing. The system then displays the relevant books basing on recommendations generated by the data mining algorithm implemented. The user may select a particular book and post comments for the book or give ratings. These ratings are the parameters for recommending books to the user.

- **Admin module**

It is an interface for the administrator to upload the new books. The administrator can add the books with different fields. Those fields are book id, book name, year, image, publication year and genre. There is an authentication system to the admin before he/she logs into their account.

- **Creation of Hadoop framework module**

As mentioned earlier, Hive contains a metastore where we create tables to work with Hive Queries. Metastore is the description of data present in Hadoop Distributed File System. The tables required for query processing are Book, Book Ratings, and Book genres. The data can be stored in Hive Tables using a simple select statement of data present in HDFS.

- **Rating Module**

Firstly, the Book users table is created to store every book for which user has provided rating information. Map-Reduce jobs are then invoked for analyzing the rating information provided by the users. After analyze we define the top rating books. Top rated books information is stored in RDBMS back from HDFS with the help of SQOOP utility. Those top rated books displayed for admin.

- **Recommendation module**

Initially the unseen and high rated books are displayed as Recommended books for users. Later on, when recommendations are made with comments/rating provided by the user, those books are exhibited to the user.

2.4 Product Scope

The project's primary idea is to design a system that works with both traditional RDBMS and Big Data systems. The project does not focus on showing the statistical data that big data systems are faster than RDBMS systems. It mainly focuses on working with two different systems by leveraging their benefits. RDMS systems are good at working with transactional data. Hence, IT business working in industries like insurance, banking, and financial services, e-commerce and communication are skeptical in migrating to a distributed platform. The reason for this is those industries work on transactional data where RDBMS is an excellent fit.

However, to improve the business they need to consider user related data from various sources for analysis. These different sources will provide data in a variety of formats. So, the analysis in project deals with providing recommendations to the user in an e-commerce platform.

The Big Data analysis implemented as a prototype that work with large sets of data and vast number of nodes in the cluster. It is an implementation of the architecture of a web application which works with both traditional RDMS system and analysis in Big Data platform.

3. System Design And Architecture

Recommendation Engine provides the recommended products basing on the search criteria given by the user after logging into his account.

3.1 Proposed System Architecture:

As discussed, the proposed system works with the transactions in Relational database system. All products and recommendations shown to the user as part of the user interface are fetched from the Relational Database System.

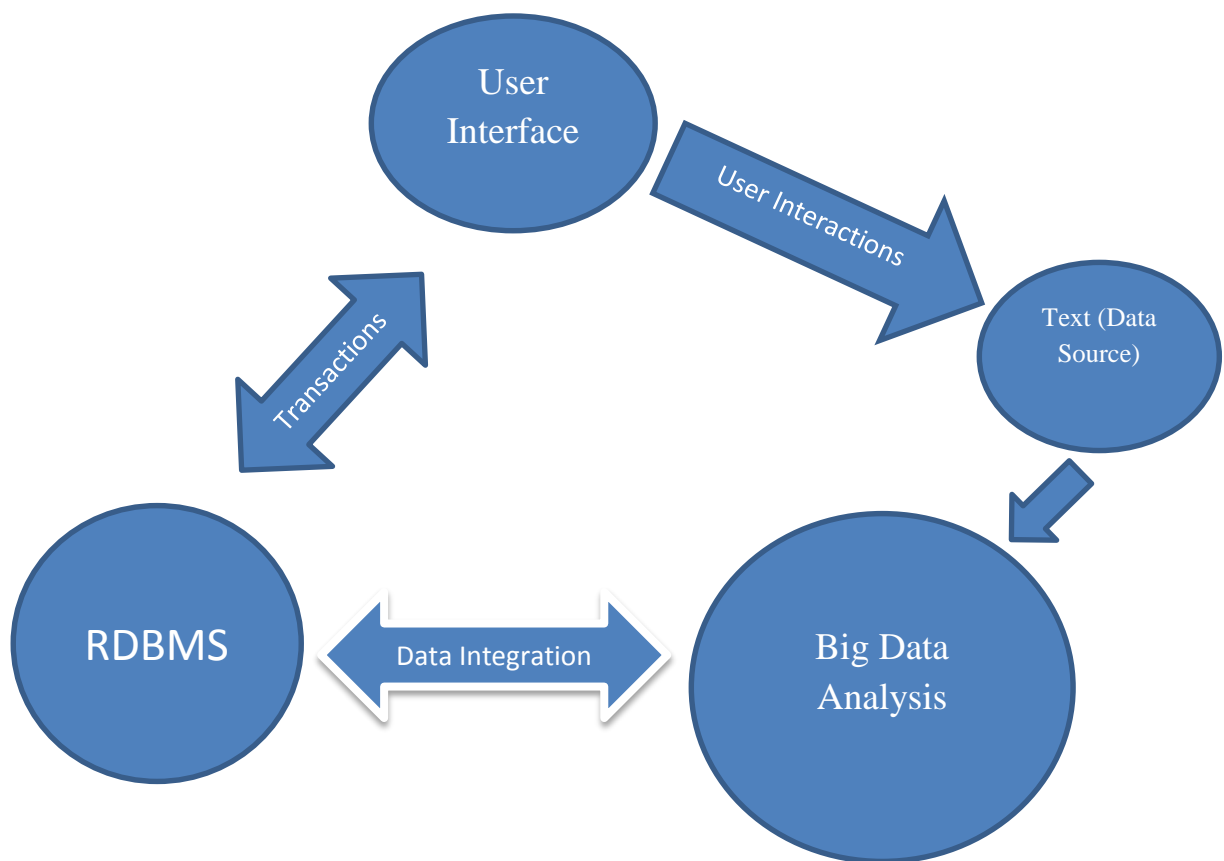


Fig 3.1 System Architecture

The following are the abilities that a Big Data processing framework needs to possess to solve this use case:

- Distributed processing engine

There is a requirement for a software system that permits the processing of large datasets across a distributed system environment. It should be able to incorporate new frameworks that aim to solve the problems to achieve the solution. As discussed in the section 1.4.3, Apache Hadoop ecosystem has addressed this. So, working with Apache Hadoop or any other distribution will enable libraries/ projects in the ecosystem (mentioned below) will be able to solve this problem. In this project, the community edition of Hadoop, i.e., Apache Hadoop of the stable version 1.2.1 is used. It could be used to scale up to 1000's of machines in the cluster without compromising on availability and reliability. However, the project is a prototype to work with Big Data and is limited to standalone installation.

- Distributed File System

To be able to work the problem in parallel to achieve the goal, there is a requirement for a distributed file system, which could be useful to distribute the content across nodes or systems in blocks. These blocks or chunks of data could be worked by single nodes to achieve the task in parallel. Hadoop Distributed File System, a core component that comes with the distribution of Hadoop. It is the file system where the data is stored and worked.

- Sophisticated query processing

Query processing is required to process the data stored in formats such text and data. Also, ability to make queries by understanding the metadata information is the key to functioning of such processing tools. In order to migrate the data back into the relational database system, it must

have a mechanism to store the output data into tables using the metadata information. Apache Hive is a query processing utility in Hadoop Ecosystem to address this requirement as discussed in Section 1.4.3. These queries could be run in the script as a batch process during non-business hours if the application servers are busy serving applications during high traffic business time.

- Data Integration mechanism

Data integration is required to retrieve data from Relational Database system and Big Data environment for analysis. It is also used to put results back into the RDBMS system in the form of tables, which is the core functionality of the RDBMS. SGOOP is the utility to connect the database remotely (since both RDBMS and HDFS are in different file systems) using Java Database Connectivity and fetch the data into Hadoop Distributed File System in the form of tables. It is also used to export the relational data into RDBMS such as MySQL.

- Machine Learning Library

A Machine library needs to be available in the environment to process the data and extract the knowledge. The requirement of new machine library is essential because, algorithms in the library should be able to run on and take advantage of distributed or parallel environment supporting the platform. A collaborative algorithm is needed to be implement for achieving user based item recommendations. Apache Mahout is one such platform to provide distributed algorithms aimed to work with Hadoop Engine 1.4.3 (v).

3.2 Proposed System Design

The figure below (Fig 3.2) describes the design of the admin interactions in the project. The administrator is responsible uploading products. He generates top rated books and unseen books for the user using batch processing. Those produced top rated books and unseen books are

stored in Relational Database and may be seen by the viewed by logging into the web page and web application. User searches the products he is interested. He can browse the displayed search results and provide ratings that become the basis for recommendations.

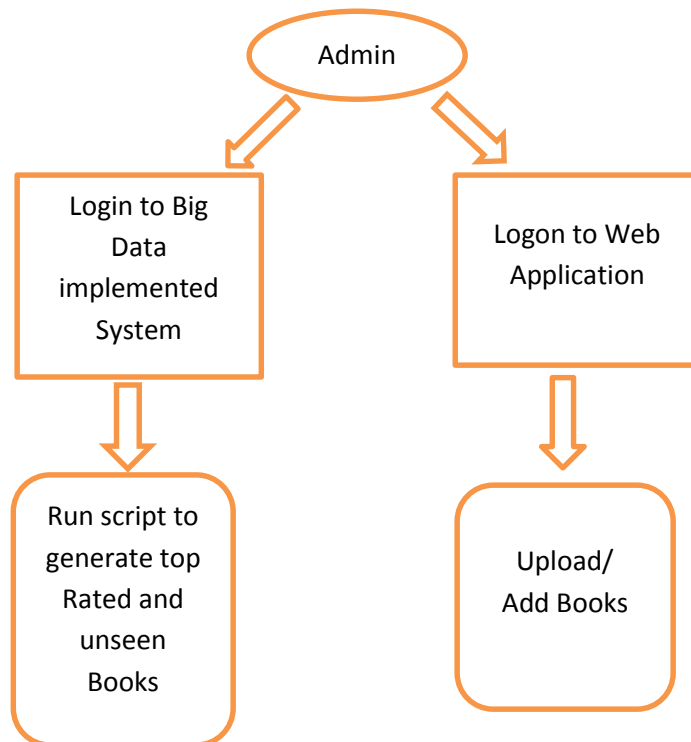


Fig 3.2 System Design Flowchart for Admin

Fig 3.3 portrays the system design flowchart for the user. The user may logon to the homepage of the system, where the user is shown the top rated books generated by the query processed by the admin. Upon selecting a product, the user will be redirected to login page. Upon successful login, the user may then look for unseen books through the link on the home page. He may search for a particular item and view results of recommendations along with the searched products on top.

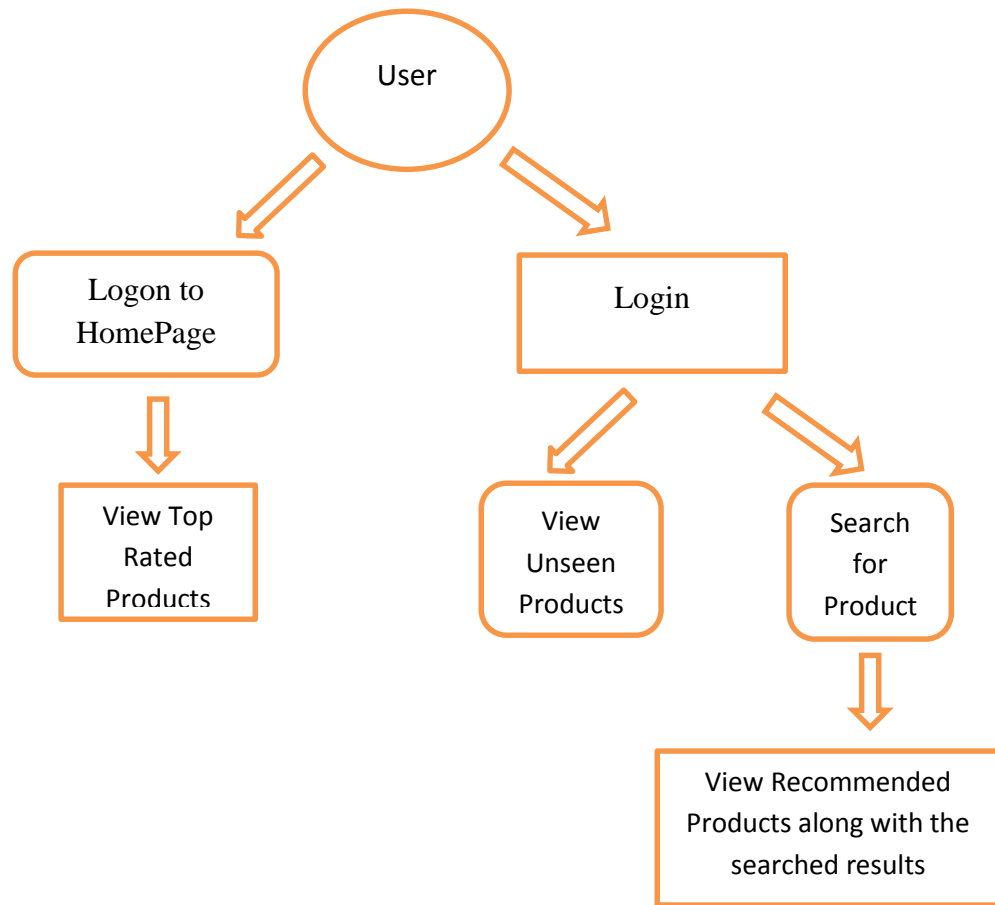


Fig 3.3 System Design Flowchart for User

3.3 System Requirements

3.3.1 Functional Requirements

- The user needs to logon to the home page of the application.
- The user needs a registered account in the web application.
- Admin runs a script to generate top rated and unseen books.
- Admin needs to login to upload books.
- User needs to hit a hyperlink to view unseen books
- Users need to search for a book to see recommendations.
- Users click on a particular product to give ratings and comments.

3.3.2 Software Requirements

- Programming Language : Core Java, JSP, Servlets, JDBC
- Version : JDK 1.6
- Integrated Development Environment : Eclipse
- Database Server : MySQL server 5.1
- Application Container : Apache Tomcat 6
- Big Data platform : Apache Hadoop 1.2.1
- Query processing utility : Apache Hive 0.10.0
- Data mining library : Apache Mahout 0.9
- Operating System : Ubuntu 12.04

3.3.2 Hardware Requirements

- Processor : Intel Pentium Dual Core
- Clock Speed : 2.5 GHz
- Ram capacity : 2 GB
- Hard Drive Capacity : 250 GB

4. System Implementation

To implement the proposed architecture, this project requires software as mentioned in section 3.3.2. To install Apache Hadoop, the system requires a Linux-based operating system. So, Ubuntu 12.04 operating system is chosen to install Apache Hadoop and its related projects on the Linux platform.

4.1 Environment

The user interface is developed using Eclipse IDE since it provides a folder structure and ease of programming. Gedit is used for writing scripts in Ubuntu environment. It is also used to view the written SQL queries for Hive. Apache Mahout Libraries are set to the build path of the environment so that its API could be used for performing data mining tasks.

4.1.1 Eclipse

Eclipse is an Integrated Development Environment, a best-known platform for developing Java applications. The core advantages of IDE is that it automates the folder structure depending on the type of project and provides IntelliSense. IntelliSense is the ability to provide suggestions that increase the speed of coding. It also has the inbuilt drivers to connect to some of the application servers. This project requires a Java dynamic web application to be developed that talks to MySQL database. This IDE provides a build environment allowing Libraries to add to the project's build path.

4.1.2 Apache Tomcat

“Apache Tomcat is an open source web server and a web container developed by Apache Software Foundation” [15]. It is used by many enthusiasts and students in the IT industry since it is freely available and easy to operate.

4.1.3 Code organization

As it is a Java web Application, this project follows the industry standards for implementing a Java web project. Java Design Patterns are used for flow of the project. It builds the code standardized, loosely coupled and readable. However, a higher level of Java frameworks such as Spring/Struts is not implemented. The packages are divided in such a way that the naming convention says it all. For example, actions package deals with servlets and Java classes and daoImpl deals with the SQL queries done on the RDBMS (MySQL).

4.2 User Interface

The user interface of the project is developed using Java Server Pages. The needed styling for the pages is done using Cascading Style Sheet. Validations required on the client side are implemented using Java Scripts. This project uses one of the javascript files from apycom.com. Hence, a hyperlink to their website may be shown in the header. Sessions are maintained till logout of the system, to make sure the authenticated user stays connected with the system. There might be a chance of glitches in one or two JSP files, which could be fixed easily.

4.2.1 Admin Web User Interface:

Admin can login to the system using authentication system and is redirected to his home page. Once administrator logs on the web interface module, he will be shown the admin home page as shown in Fig 4.1

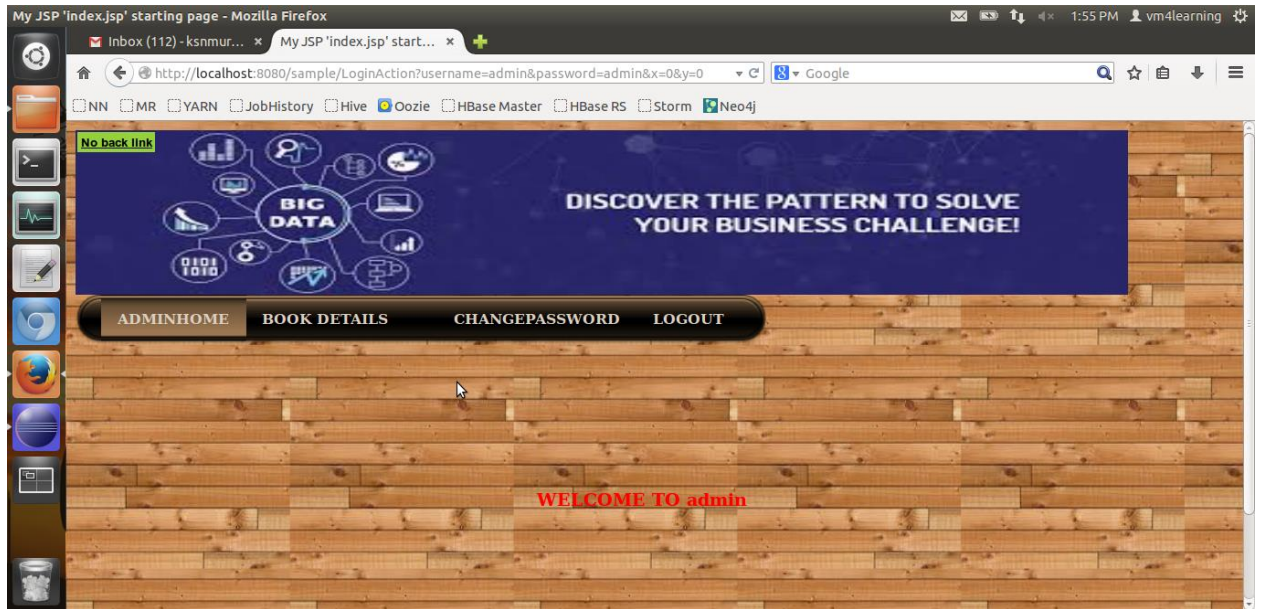


Fig 4.1 Admin Home

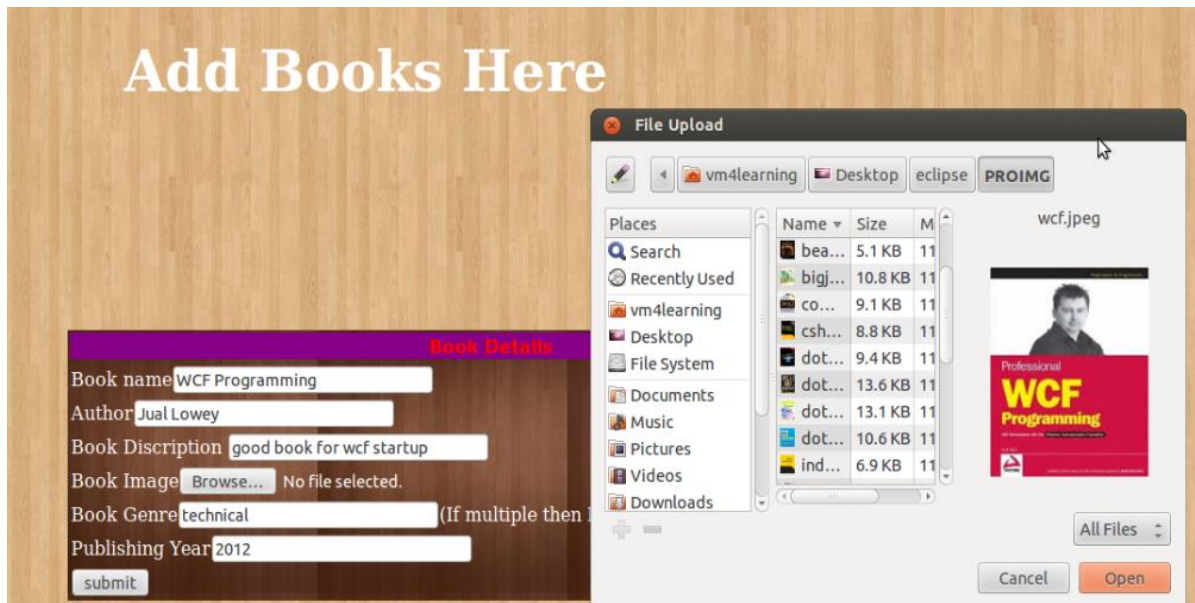


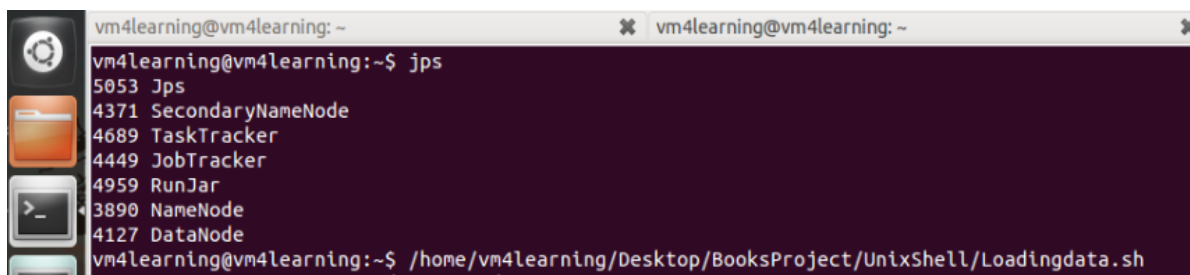
Fig 4.2 Admin adds books into RDBMS

Once the user logs into the system, he can add books by providing the information required to store the data in MySQL server as shown in Fig 4.2.

4.2.2 Command User Interface for Admin

The administrator of the project is responsible installing and setting up the Hadoop ecosystem in the project. Once the system reboots or starts, Admin runs the commands to start Hadoop Engine and its components. He should start the name node, data node and secondary name node that are managing and work horses to run Hadoop Ecosystem.

Once, the Hadoop components are up and running, which can be viewed by running the command 'jps' on the terminal, administrator runs the script to load Top rated books and unseen top rated books into MySQL server as shown in figures Fig 4.3 and Fig 4.4.

A terminal window titled 'vm4learning@vm4learning: ~' with a dark purple background. The user has entered the command 'jps'. The output lists several Hadoop processes with their PIDs: 5053 Jps, 4371 SecondaryNameNode, 4689 TaskTracker, 4449 JobTracker, 4959 RunJar, 3890 NameNode, and 4127 DataNode. Below this, the user has entered the command '/home/vm4learning/Desktop/BooksProject/UnixShell/Loadingdata.sh'.

```
vm4learning@vm4learning:~$ jps
5053 Jps
4371 SecondaryNameNode
4689 TaskTracker
4449 JobTracker
4959 RunJar
3890 NameNode
4127 DataNode
vm4learning@vm4learning:~$ /home/vm4learning/Desktop/BooksProject/UnixShell/Loadingdata.sh
```

Fig 4.3 Admin runs queries to perform processing

Data Flow mechanism built in Shell Script:

- Hadoop distributed file system fetches the rating data stored in the application server using `hdfs -put` command.
- Once the data is fetched into HDFS, Hive query processing tool can work on the data stored in HDFS to create tables in Hive Metastore. Hive Metastore contains the data about the data fetched into HDFS. It does not store the data in tabular form. The reason for storing metadata in the form of table description is the requirement to attain the ability to process Hive Queries (Similar to SQL). So, creating the tables (metadata) is the next step in processing the data. External tables, Managed tables are the two types of creating tables in Hive. External tables when dropped, data in the HDFS will not be lost and in case of Managed tables, data would be lost.

The tables created are Book, Book ratings and Book users that help in analyzing to retrieve top rated and unseen books.

- The script then executes the hive query to create top rated books tables and generates top rated books based on average rating in the book. The query also takes the minimum average rating and to maximum number of items that could be the output as arguments. This query invokes the map-reduce jobs and generates the top rated books by dividing the work.

```
#!/bin/sh

cd $HADOOP_HOME
bin/hadoop fs -rmr /user/vm4learning/*
bin/hadoop fs -mkdir /user/vm4learning/Books/
bin/hadoop fs -mkdir /user/vm4learning/Books/Books/
bin/hadoop fs -mkdir /user/vm4learning/Books/Ratings/

bin/hadoop fs -put /home/vm4learning/Desktop/Books/Books.dat /user/vm4learning/Books/Books/
bin/hadoop fs -put /home/vm4learning/Desktop/Books/Ratings.dat /user/vm4learning/Books/Ratings/

cd $HIVE_HOME
bin/hive -f /home/vm4learning/Installations/hive-0.11.0-bin/Project/BooksRec/CreatingBasetables.sql

cd $HIVE_HOME

bin/hive -f /home/vm4learning/Installations/hive-0.11.0-bin/Project/BooksRec/CreatingTopRated.sql
bin/hive -f /home/vm4learning/Installations/hive-0.11.0-bin/Project/BooksRec/TopRated.sql -hiveconf sample_min=2 -hiveconf topN=10

cd $SQOOP_HOME

bin/sqoop export --connect jdbc:mysql://localhost/bdata --table TopRatedBooks --export-dir /user/hive/warehouse/topratedbooks/000000_0 -m 1

cd $HIVE_HOME

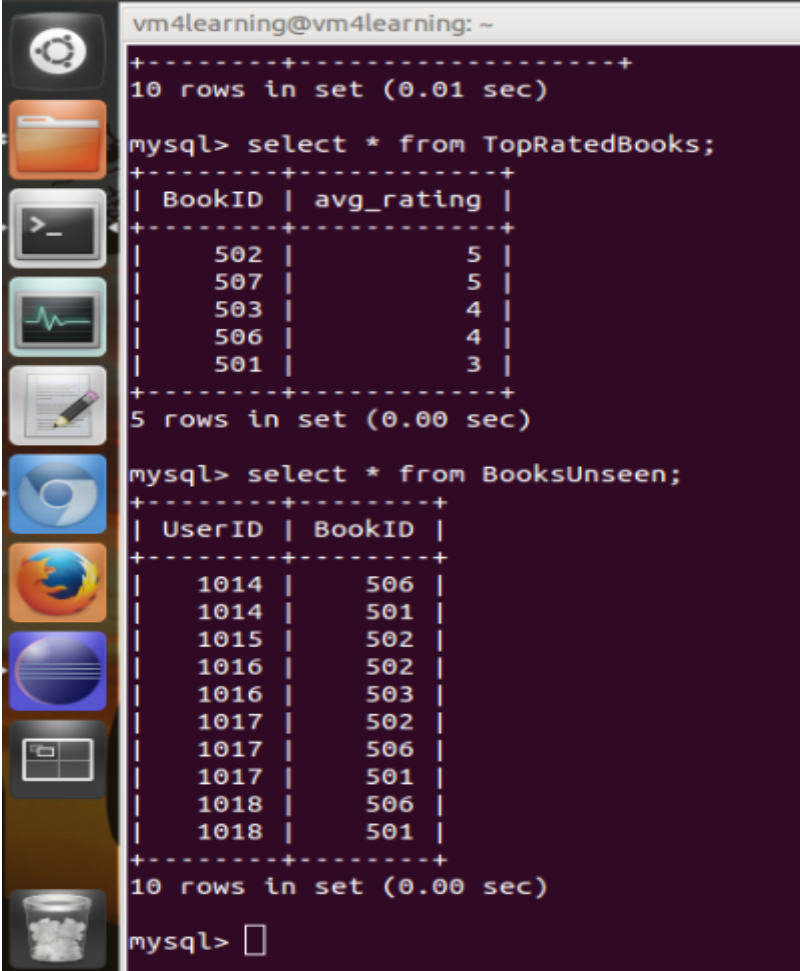
bin/hive -f /home/vm4learning/Installations/hive-0.11.0-bin/Project/BooksRec/CreateUnseenBooks.sql
bin/hive -f /home/vm4learning/Installations/hive-0.11.0-bin/Project/BooksRec/TopUnseenBooks.sql

cd $SQOOP_HOME
bin/sqoop export --connect jdbc:mysql://localhost/bdata --table BooksUnseen --export-dir /user/hive/warehouse/booksunseen/000000_0 -m 1
```

Fig 4.4 Code for Admin Script

- Once the functionality is achieved, the table is exported to the MySQL server using data integration utility, SQOOP. This utility connects to the MySQL database remotely using the JDBC connection to achieve the migration.
- Similarly, the tables required for generating top rated unseen books of the user are created. For achieving this, the unseen books (books for which rating is not given) are generated. These unseen books are joined with the top rated books to generate the top unseen books. This table contains user id and book id of users and books respectively.

- Finally, this table is also exported to the MySQL server using the SQOOP utility, which could be observed in Fig 4.5.



```

vm4learning@vm4learning: ~
+-----+-----+
10 rows in set (0.01 sec)

mysql> select * from TopRatedBooks;
+-----+-----+
| BookID | avg_rating |
+-----+-----+
| 502    | 5          |
| 507    | 5          |
| 503    | 4          |
| 506    | 4          |
| 501    | 3          |
+-----+-----+
5 rows in set (0.00 sec)

mysql> select * from BooksUnseen;
+-----+-----+
| UserID | BookID |
+-----+-----+
| 1014   | 506    |
| 1014   | 501    |
| 1015   | 502    |
| 1016   | 502    |
| 1016   | 503    |
| 1017   | 502    |
| 1017   | 506    |
| 1017   | 501    |
| 1018   | 506    |
| 1018   | 501    |
+-----+-----+
10 rows in set (0.00 sec)

mysql>

```

Fig 4.5 Tables exported to MySQL database

4.2.3 User Interface for User Module

The user may logon the homepage of the web application, where he is shown the top rated books from the Top rated books table from the database as shown in Fig 4.6. The user is redirected to login page upon clicking a particular any item or the hyperlink displayed on the homepage. The login page displayed is shown in Fig 4.7.

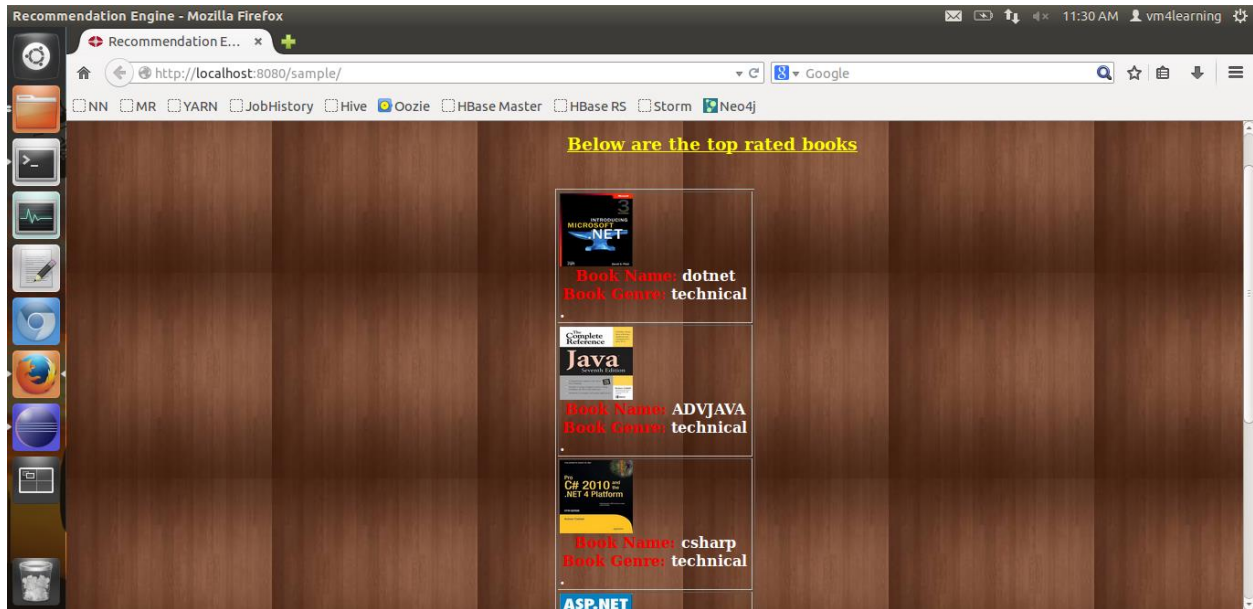


Fig 4.6 Home Page of the web app



Fig 4.7 User Login page

Once users login to the system, they are shown the homepage of the application. There is a link to access the top rated unseen books in the store as shown in Fig 4.7. Once he/she clicks the unseen books' link, he/she is shown the top rated unseen books of the system as shown in Fig 4.8.



Fig 4.8 User Home Page

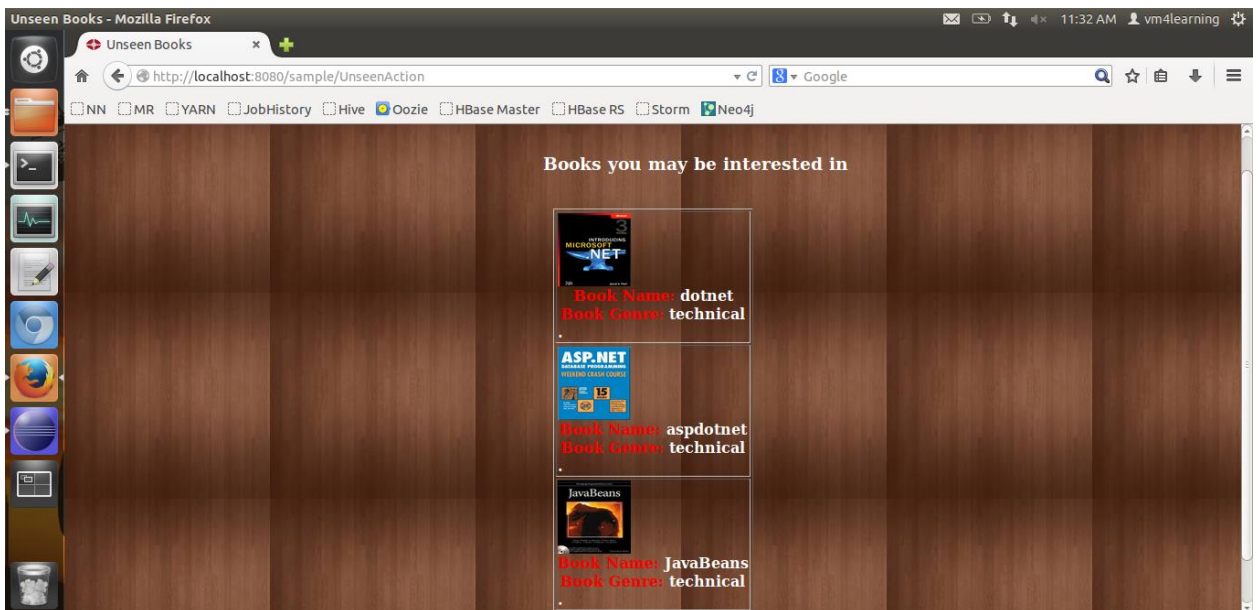


Fig 4.9 Top rated unseen books

After users login to their account, they can search for a book using book name or author name, as shown in Fig 4.12, the results come up as shown in Fig 4.13. The user may click on any of the products displayed, which will ask for ratings and comments as shown in Fig 4.10.

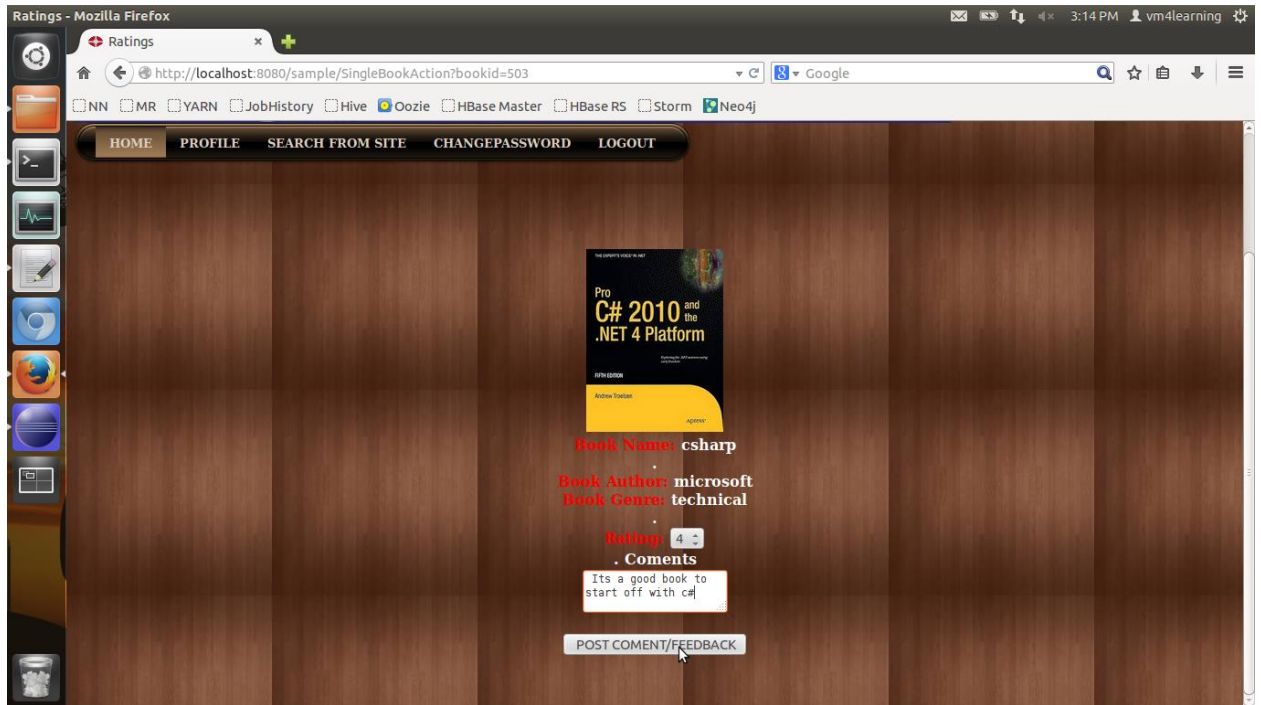


Fig 4.10 User Ratings

```

1016,507,5,
1016,506,4,
1016,501,1,
1014,502,5,
1014,503,2,
1015,510,5,
1015,503,4,
1015,506,4,
1015,507,5,
1015,501,5,

```

Fig 4.11 Ratings Stored in a Text File

When the user gives ratings, User Id, Book Id and rating are stored in a text file. The text file is stored as shown in Fig 4.11. This text file is fetched into the Hadoop Distributed File System and worked upon by invoking shell scripts, which in turn runs Hive queries to work on them.

4.3 Data Mining Algorithm for Recommendations to Search Results

User can search for a book he/she is interested in using a search box as shown in Fig 4.12. When the user searches for a particular book using search criteria as book name or book author, the product is searched in MySQL database and fetches the results. While fetching the results of book such as book id, book name, author, publication year, genre and image into a Value Object called RegisterTo, it also uses the book id of the product to produce the item based recommendations.

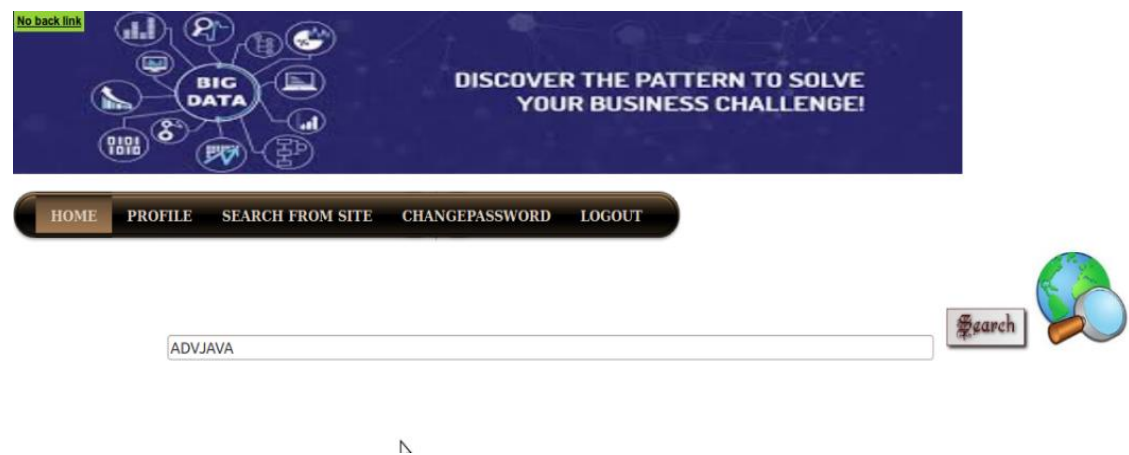


Fig 4.12 Search Functionality

To implement the recommendations, the rating file generated by the users is given input to the algorithm. The algorithm takes the data file in text format, and the book id of book searched for to provide item based recommendations

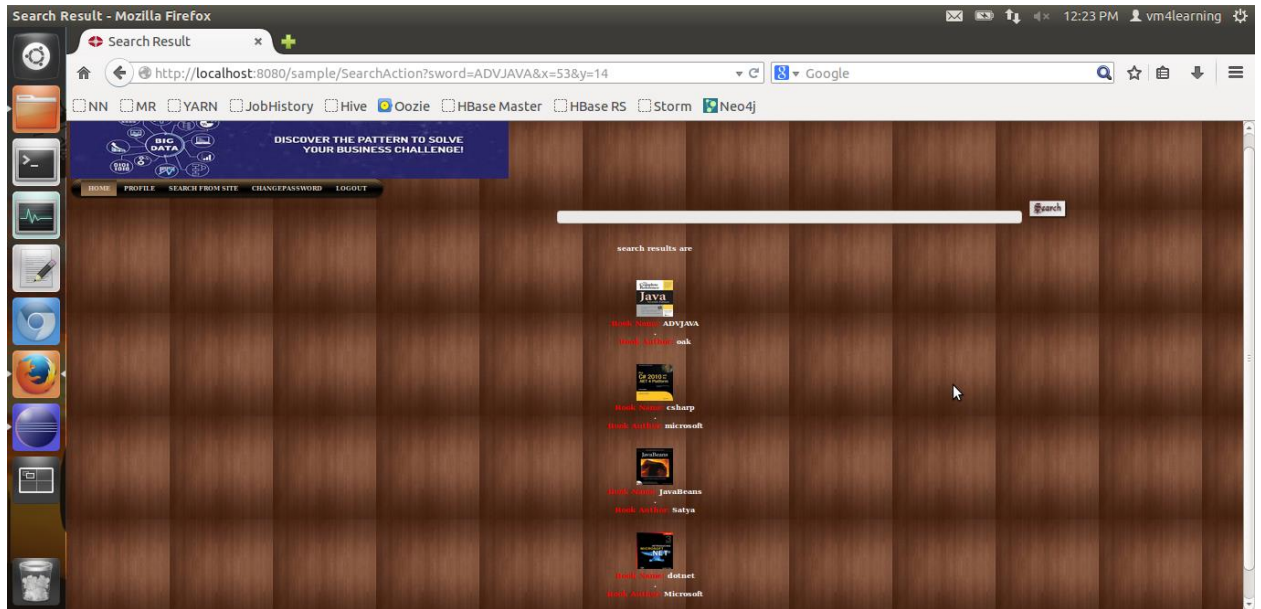


Fig 4.13 Search Result from Database Along with the Recommendations

4.3.1 Log-Likelihood Similarity Algorithm

The Log likelihood Similarity algorithm predicts the probability of a user liking the item based on the user's similarity of liking an algorithm. If x is a user, who rates 200 products in the system, and y is a user who rates the 300 products in the system, and the overlap is 150 products. There is a high probability that user x may like 100 other products and user y might like the 50 other products. On the other hand, if there is an overlap of only 50, then they may not be the similar users [16].

The algorithm is designed basing on the log-likelihood ratio on the user similarity by the logarithm of probability difference. This algorithm performs better when we need less number of recommendations. On the other hand, there are some algorithms such as Pearson Correlation algorithm and Euclidian Distance, which could solve the problem for recommendations using

different approach. This algorithm is implemented in Apache Mahout Library by the community in Apache Mahout 0.7 version. The library version that the project currently uses is 0.9.

The algorithm takes the user rating text file as an input using a class Data Model, the percentage of minimum similarity, number of recommendations required and the product id (book id). It gives the output of product ids (book ids) which is stored in an array and this array is run in a while loop to fetch all the books information by querying the MySQL server and display in the user interface. The implementation of the code is shown in Fig 4.14.

```
public static int[] itemRecommend(int bid){
    int[] itemRec=new int[3];
    int i=0;
    try {
        DataModel model = new FileDataModel(new File("/home/vm4learning/Desktop/Books/Ratings.dat"));
        ItemSimilarity similarity = new LogLikelihoodSimilarity(model);
        GenericItemBasedRecommender recommender = new GenericItemBasedRecommender(model,similarity);
        long itemId = bid;
        List<RecommendedItem>recommendations = recommender.mostSimilarItems(itemId, 3);
        for(RecommendedItem recommendation : recommendations) {
            System.out.println(itemId + "," + recommendation.getItemID() + "," + recommendation.getValue());
            itemRec[i]=(int)recommendation.getItemID();
            i++;
        }
    } catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
    catch (TasteException e1) {
        System.out.println("There was a Taste Exception\n");
        e1.printStackTrace();
    }
    return itemRec;
}
```

Fig 4.14 Code Block for Fetching Recommendations

The method '*itemRecommend*' takes the book id as the recommendation to return an array of similar book ids. The FileDataModel, LogLikelihoodSimilarity, GenericItemBasedRecommender are the classes available in Apache Mahout API. These classes are built on implementing the Java Interfaces

DataModel, ItemSimilarity and ItemBasedRecommender. The Java class of this method imports the above classes for initializing the objects. DataModel Java Class has the method to work with text files, CSV data files with delimiters. It takes the data file as input and understands the format. The object of the DataModel is given to the LogLikelihood similarity algorithm so that it appropriately could call the methods to work with file data types. The object of Similarity Algorithm Class and DataModel is given to GenericItemBasedRecommender to produce recommendations List. The List is created by calling *mostSimilarItems* method from Recommender object while limiting its output to a certain number. The output is the item ids of products (book ids in our use-case). These book ids are returned as an array to the database, so that the books information can be fetched and pushed to the front end.

5. Testing And Evaluation

Software testing is the process of evaluating the product by testing with different inputs the system takes and product output results.

In our use case, we are mostly involved in getting recommended products to the user based on the search. Here are three test cases with three different acceptable test cases, where the system could find the product in the database and the results.

5.1 Test Cases

Firstly, as shown in Fig 5.1, the user searches for the book 'JavaBeans'. The user is displayed with the JavaBeans book along with three item recommendations basing on user similarity. The user has already rated the book 'JavaBeans', basing on the other users who liked that book, the user is shown the recommendations.

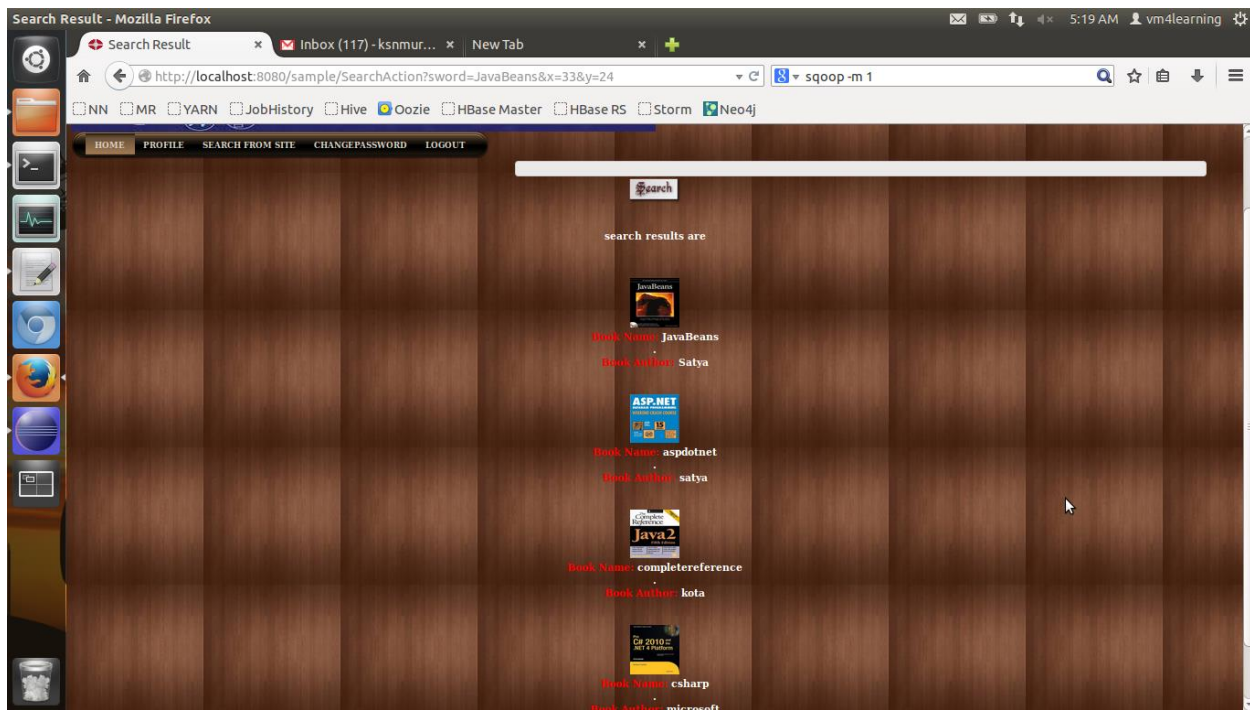


Fig 5.1 Test Case 1

In the second test case as shown in Fig 5.2, the user did not rate the book 'learner', when the user searches for the book 'learner', the user similarity of book who rated the book is checked, the books liked by the similar user are recommended.

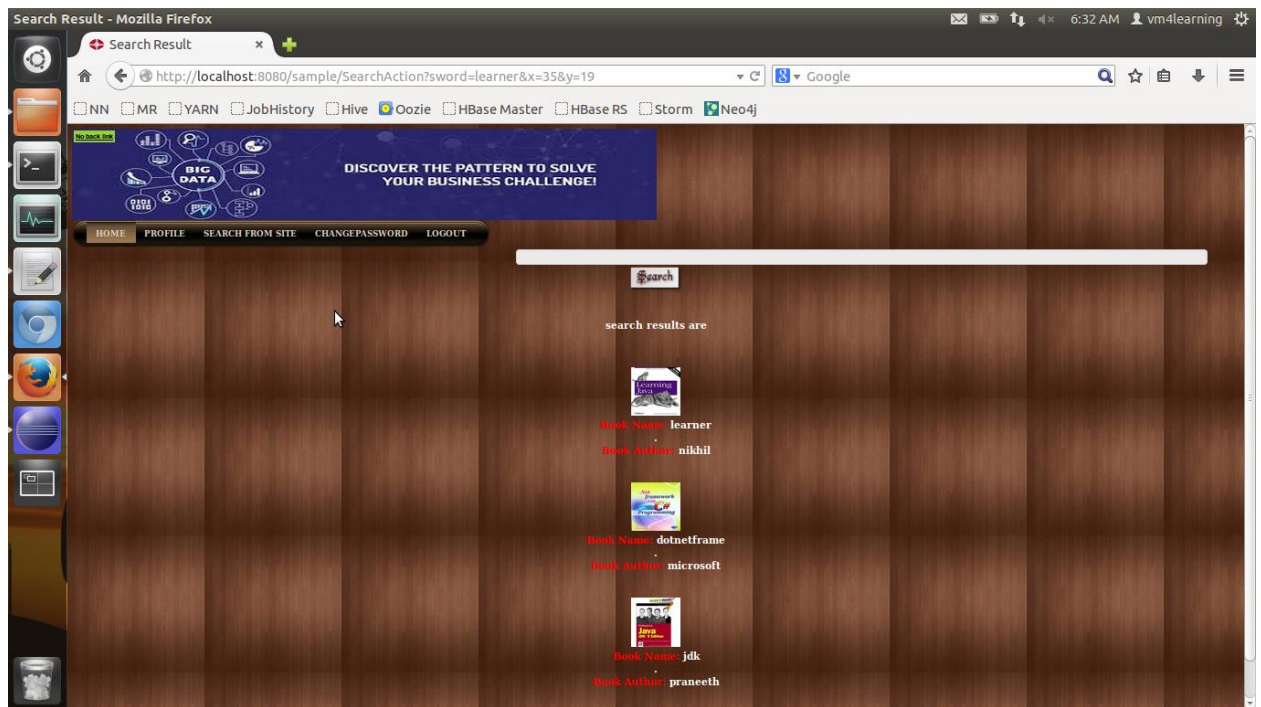


Fig 5.2 Test Case 2

In Fig 5.3, the user searches for a book 'WCF Programming', that is not rated by any user. The user will be displayed with only that book since he does not have user-based recommendations. To improve this, the top rated unseen books can be shown as recommendations to the user.

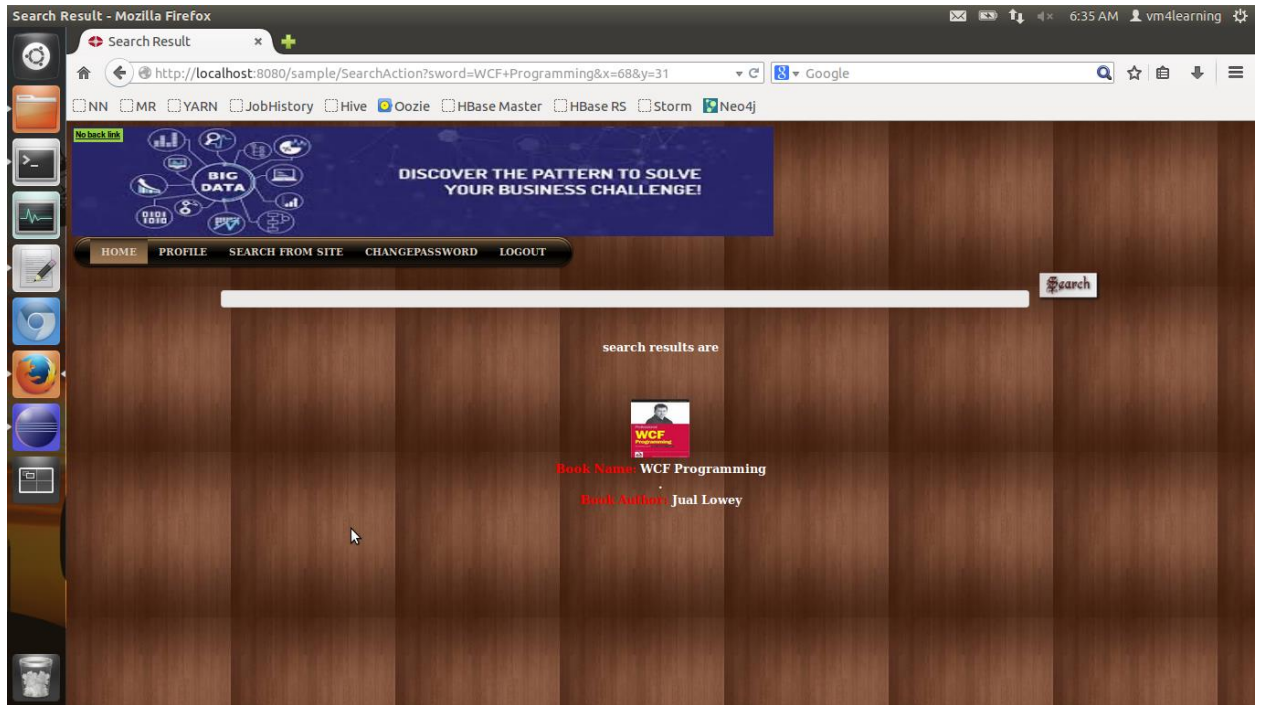


Fig 5.3 Test Case 3

5.2 Evaluation of Recommendations

There is no accurate way to evaluate a Recommendation System [17]. To decide the quality of a Recommendation System, an A/B test is conducted in a live environment or a live dataset. The results would get better with increased number in similar users and a number of items a user rates.

However, in a developer environment, the quality of the algorithm could be felt by statistical evaluation. In addition to this, a hold-out test could be performed on the partitioned dataset by taking a training set and test set and compare with the original results [17].

The Java class *AverageAbsoluteDifferenceRecommenderEvaluator* is used to find the real interaction similarity. This test must be run several times, since splitting the training set, and test set is done on a random basis. The code snippet in Fig 5.4 is run several times in the development environment and the results are noted and averaged.

```
DataModel model = new FileDataModel(new File("/path/to/dataset.csv"));
RecommenderEvaluator evaluator = new AverageAbsoluteDifferenceRecommenderEvaluator();
RecommenderBuilder builder = new MyRecommenderBuilder();
double result = evaluator.evaluate(builder, null, model, 0.9, 1.0);
System.out.println(result);
```

Fig 5.4 Evaluator Code

The results are in double format, and 1.0 is the result if the test set and training set are same. Occasionally, the results might be shown as 'NaN' (Not a Number) due to either small test cases or the compiler's inability to show the result. In such case, the code is re-run to get the result in order to evaluate the similarity.

The following is the test results for the five times (0.91932843, 0.746512, 0.8578123, 0.7586234, 0.84678712). The average test result is 0.82581877. The test case results could be improved by the employing more test data from the live users and ratings. Results improve in production environment as rating increase.

5.3 Usability Testing

Usability testing is an evaluation method to understand the interaction design [18]. Feedback from the users is taken to understand their likeliness. Then, feedback is taken into consideration and time for implementing the requirements is evaluated. This could help the enterprise to enhance the application in future releases or the current release basing on time.

The project takes a survey to understand the usability of the application especially relating to the recommendations. During the survey, the users searched for their book they are interested. They shared their preference of the recommendations provided by the system with a percentage. The average likelihood of the users is 85%. The preference of five users as follows: [75, 80, 85, 90, 95]. User 1 stated that the recommendations could be also extended to similar items, so that

they can decide on what products to choose. User 2 is happy with the recommendations and also suggested that system would be better if more number of recommendations could be provided. User 3 likes the recommendations provided but likes to see an enhanced User Interface. User 4 and User 5 are happy with the recommendations and shared their preference. All the users found one or more recommendations interesting and browsed the product shown in recommendations. The feedback provided by some users could be extended in the future work.

6. Conclusion and Future Work

The project proposes an idea of recommending books to the user using Big Data framework. The information on books' and users' is present in the traditional RDBMS. While all the transactions happen through RDBMS, behavioral data of the user such as giving ratings to a particular product could be utilized to provide recommendations. By leveraging the batch processing power of Big Data environments such as Hadoop and advantages of RDBMS over transactional data (since Relational databases are good at maintaining ACID properties of data), this project aims to build a reliable platform for recommendations or analysis. These mining analyses will help the firms to get more business. This use case is the successful implementation of a prototype that could work with Big Data.

On the other hand, the use case of an online bookstore could be extended to a real-time shopping cart application by using the add to cart and billing modules. Secondly, the scripts run by the administrator could be automated by running a shell script. The application could be hosted in a cloud-based cluster environment such as Amazon Elastic Map Reduce, Cloudera, and Horton Works to take advantage of distributed computing by working with Big Data.

6. Bibliography and References

- [1] <http://www.ibmbigdatahub.com/infographic/four-vs-big-data> -visited on 02/25/2015.
- [2] Data Mining with Big Data, IEEE Transactions on knowledge and data/ Engineering, vol. 26, no. 1, January 2014
- [3] Semantics-Empowered Big Data Processing with Applications, Krishnaprasad Thirunarayan, Amit P. Sheth, 11-2013
- [4] <https://www.oracle.com/java/index.html> - ‘visited on 02/12/2015’
- [5] <http://www.oracle.com/technetwork/java/index-jsp-135995.html> - ‘visited on 02/25/2015’
- [6] <https://www.mysql.com/products/workbench/> - ‘visited on 02/27/2015’
- [7] MapReduce: Simplified Data Processing on Large Clusters, Jeffrey Dean and Sanjay Ghemawat , Google Inc, 2004
- [8] <https://hadoop.apache.org/> -‘visited on 01/25/2015’
- [9] <http://hadoop.apache.org/releases.html> - ‘visited 02/02/2015’
- [10] <https://gigaom.com/2013/03/07/5-reasons-why-the-future-of-hadoop-is-real-time-relatively-speaking/> -‘visited on 02/13/2015’
- [11] <http://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html> - ‘visited on 02/03/2015’
- [12] MySQL tips: <http://www.mssqltips.com/sqlservertip/3262/big-data-basics--part-6--related-apache-projects-in-hadoop-ecosystem/> - ‘visited on 03/2/2015’
- [13] <https://mahout.apache.org/users/basics/algorithms.html> - ‘visited on 04/01/2015’
- [14] http://en.wikipedia.org/wiki/Eclipse_%28software%29 - ‘visited on 04/05/2015’
- [15] http://en.wikipedia.org/wiki/Apache_Tomcat -‘ visited on 04/04/2015’

- [17] http://mail-archives.apache.org/mod_mbox/mahout-user/201105.mbox/%3CBANLkTi=TeA56NLOjmquveNxwB_C5HQAFEQ@mail.gmail.com%3E - ‘visited 04/10/2015’
- [17] <http://grouplens.org/site-content/uploads/evaluating-TOIS-20041.pdf> - ‘visited on 05/01/2015’
- [18] http://en.wikipedia.org/wiki/Usability_testing - ‘visited on 05/02/2015’