

**A
PROJECT REPORT
ON
SMART PRODUCT RECOMMENDATION**

Submitted in partial fulfillment of award of the degree of

**BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE & ENGINEERING**



SESSION 2014-15

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SRM UNIVERSITY, NCR CAMPUS,
MODINAGAR**

SUPERVISED BY:

Ms. Arti Tiwari
Assistant Professor (CSE)

SUBMITTED BY:

Jatin Jha (1031130136)

BONAFIDE CERTIFICATE

This is to certify that project Report entitled “**Smart Product Recommendation**”, which is submitted by **Jatin Jha (1031130136)** in the partial fulfillment of the requirement for the award of degree B. Tech. in Department of Computer Science & Engineering of **SRM University, NCR Campus, Modinagar** is a record of the candidate’s own work carried out by them under my own supervision. The matter embodied in this thesis is original and has not been submitted for the award of any other degree.

.....
Dr. R. P. Mahapatra
HOD (CSE)
SRM University
NCR Campus
Modinagar, Ghaziabad.

.....
Ms. Arti Tiwari
Assistant Professor
SRM University
NCR Campus
Modinagar, Ghaziabad

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

I owe special debt of gratitude to **Ms. Arti Tiwari, Assistant Professor and Project Supervisor of SRM University, NCR Campus, Modinagar**, for his insight and knowledge on the subject, who imbued us with the feeling to work assiduously.

I feel compelled to articulate our thankfulness to **Ms. Arnika and Mr. Manish Bhardwaj, Project Coordinators, Computer Science and Engineering Department, SRM University NCR Campus, Modinagar**, for their enlightening dexterity which was a perennial source of inspiration.

Also, I extend my sincere regards and thanks to **Dr. R. P. Mahapatra, Associate Dean & Head of the Department, Computer Science and Engineering, SRM University, NCR Campus, Modinagar**, for his thoughtful and clairvoyant suggestion.

I would like to express my hearty gratitude to **Dr. (Prof.) Manoj Kumar Pandey, Director, SRM University NCR Campus, Modinagar**, under whose auspices we were able to work on our project work.

I am also indebted to all teaching and non-teaching staff members of our college for helping us directly or indirectly by all means throughout the course of our study and project work.

Finally, I take this opportunity to thank our friends for their moral support and help.

DECLARATION

I **Jatin Jha (1031130136)** hereby *declare that the work which is being presented in the project report “**Smart Product Recommendation**” is the record of authentic work carried out by us during the period from January '15 to May '15 and submitted by us in partial fulfillment for the award of the degree “**Bachelor of Technology in Computer Science and Engineering**” to **SRM University, NCR Campus, Ghaziabad (U.P.)**. This work has not been submitted to any other University or Institute for the award of any Degree/Diploma.*

Jatin Jha (1031130136)

ABSTRACT

Recommender systems apply knowledge discovery techniques to the problem of making personalized recommendations for information, products or services during a live interaction. These systems, especially the k-nearest neighbor collaborative Filtering based ones, are achieving widespread success on the Web. The tremendous growth in the amount of available information and the number of visitors to Web sites in recent years poses some key challenges for recommender systems. These are: producing high quality recommendations, performing many recommendations per second for millions of users and items and achieving high coverage in the face of data sparsity. In traditional collaborative Filtering systems the amount of work increases with the number of participants in the system. New recommender system technologies are needed that can quickly produce high quality recommendations, even for very large-scale problems. To address these issues we have explored item-based collaborative Filtering techniques. Item-based techniques first analyze the user-item matrix to identify relationships between different items, and then use these relationships to indirectly computer recommendations for users.

In this project, we analyze the problems of the current recommendation system used i.e. Collaborative Filtering and aims to resolve these problems. The ultimate work of this paper is to showcase a recommendation system with a little hint of common sense. This would not only give better results but also avoid absurd results which we get in previous recommendation systems. Our experiments suggest that item-based algorithms provide dramatically better performance than user-based algorithms.

Keywords: Electronic Commerce, Recommender System, Collaborative Filtering, Similarity Coefficient.

TABLE OF CONTENTS

Chapter No.	Page No.
Certificate.....	ii
Acknowledgement.....	iii
Declaration.....	iv
Abstract.....	v
List of Equations.....	viii
List of Figures.....	ix
Chapter 1: Introduction	
1.1 About Recommendation System.....	1
1.2 Types of Recommendation System.....	3
1.3 Introduction to Collaborative Filtering.....	6
1.4 Introduction to Similarity.....	9
1.5 Examples of Recommender System.....	11
Chapter 2 : Literature Survey	
2.1 Base Paper 1.....	13
2.2 Base Paper 2.....	15
Chapter 3 : Hardware and Software Requirements	
3.1 Hardware Requirements.....	17
3.2 Software Requirements.....	17
Chapter 4 : Proposed Work	
4.1 Existing Problem.....	18
4.2 Existing Technique.....	19

4.3 Proposed Solution.....	21
4.4 Proposed Algorithm.....	23
Chapter 5: Methodology	27
Chapter 6: Implementation	
6.1 Algorithm Used.....	29
6.2 Context Diagram	
6.2.1 UML Diagram.....	31
6.2.2 State Diagram.....	32
6.2.3 Use Case Diagram.....	33
6.2.4 Data Flow Diagram.....	34
Chapter 7: Result	
7.1 Home Page.....	36
7.2 Sign Up Page.....	37
7.3 Add Product.....	38
7.4 User 1.....	39
7.5 User 2.....	42
7.6 User 3.....	45
7.7 Search Page.....	48
7.8 Change Password Page.....	49
Chapter 8: Conclusion and Future Works	
8.1 Conclusion.....	50
8.2 Future Work.....	50
References.....	51

LIST OF EQUATIONS

	Page No
Eq 1- Pearson Correlation Similarity Coefficient	9&20
Eq 2- Constrained Pearson Correlation Coefficient	10
Eq 3- Cosine Similarity Coefficient	10
Eq 4- Weighted Average of Deviations from the Neighbor's Means	20
Eq 5- Pseudo User Rating Value	24
Eq 6- Cumulative Value of Similarity Coefficient	26

LIST OF FIGURES

Fig 4.1- Isolation of Correlated Items and Similarity Computation	21
Fig 5.1- Methodology Diagram of the project	28
Fig 6.1- UML Diagram	31
Fig 6.2- State Diagram	32
Fig 6.3- Use Case Diagram	33
Fig 6.4- Level 0 DFD	34
Fig 6.5- Level 1 DFD	35
Fig 6.6- Level 2 DFD	35
Fig 7.1-7.14 Snapshots of Implementation	36-49

CHAPTER - 1

INTRODUCTION

1.1 About Recommendation System

Every day, we are inundated with choices and options. What to wear? What movie to rent? What stock to buy? What blog post to read? The sizes of these decision domains are frequently massive: Netflix has over 17,000 movies in its selection, and Amazon.com has over 410,000 titles in its Kindle store alone. Supporting discovery in information spaces of this magnitude is a significant challenge. Even simple decisions — what movie should I see this weekend? — can be difficult without prior direct knowledge of the candidates [3].

Historically, people have relied on recommendations and mentions from their peers or the advice of experts to support decisions and discover new material. They discuss the week's blockbuster over the water cooler, they read reviews in the newspaper's entertainment section, or they ask a librarian to suggest a book. They may trust their local theater manager or news stand to narrow down their choices, or turn on the TV and watch whatever happens to be playing.

These methods of recommending new things have their limits, particularly for information discovery. There may be an independent film or book that a person would enjoy, but no one in their circle of acquaintances has heard of it yet. There may be a new indie band in another city whose music will likely never cross the local critic's radar. Computer-based systems provide the opportunity to expand the set of people from whom users can obtain recommendations. They also enable us to mine users'

history and stated preferences for patterns that neither they nor their acquaintances identify, potentially providing a more finely-tuned selection experience [3].

There is also a growing interest in problems surrounding recommendation. Algorithms for understanding and predicting user preferences do not exist in a vacuum — they are merely one piece of a broader user experience. A recommender system must interact with the user, both to learn the user’s preferences and provide recommendations; these concerns pose challenges for user interface and interaction design.

Systems must have accurate data from which to compute their recommendations and preferences, leading to work on how to collect reliable data and reduce the noise in user preference data sets. Users also have many different goals and needs when they approach systems, from basic needs for information to more complex desires for privacy with regards to their preferences [1].

However, there remain important questions in overcoming two fundamental challenges for recommendation systems. The first challenge is to improve the scalability of the recommender systems. They are able to search tens of thousands of potential neighbors in real-time, but the demands of modern systems are to search tens of millions of potential neighbors. Further, they have performance problems with individual users for whom the site has large amounts of information. For instance, if a site is using browsing patterns as indicators of content preference, it may have thousands of data points for its most frequent visitors. These “long user rows” slow down the number of neighbors that can be searched per second, further reducing scalability [2].

The second challenge is to improve the quality of recommendations for the users. Users need recommendations they can trust to help them find items they will like. Users might refuse to use recommender systems that are not consistently accurate for them [2].

In some ways, these two challenges are in conflict, since the less time a system spends searching for neighbors, the more scalable it will be, and the worse its quality. For this reason, it is important to treat the two challenges simultaneously so the solutions discovered are both useful and practical.

1.2 Types of Recommendation Systems:

The myriad approaches to Recommender Systems can be broadly categorized as

- Collaborative Filtering (CF): In CF systems a user is recommended items based on the past ratings of all users collectively.
- Content-based recommending: These approaches recommend items that are similar in content to items the user has liked in the past, or matched to attributes of the user.
- Hybrid approaches: These methods combine both collaborative and content based approaches.

1.2.1 Collaborative Filtering:

Collaborative filtering (CF) is a popular recommendation algorithm that bases its predictions and recommendations on the ratings or behavior of other users in the system. The fundamental assumption behind this method is that other users' opinions can be selected and aggregated in such a way as to provide a reasonable prediction of the active user's preference. Intuitively, they assume that, if users agree about the

quality or relevance of some items, then they will likely agree about other items — if a group of users likes the same things as Mary, then Mary is likely to like the things they like which she hasn't yet seen.

The majority of collaborative filtering algorithms in service today, including all algorithms detailed in this section, operates by first generating predictions of the user's preference and then produces their recommendations by ranking candidate items by predicted preferences. Often this prediction is in the same scale as the ratings provided by users, but occasionally the prediction is on a different scale and is meaningful only for candidate ranking. This strategy is analogous to the common information retrieval method of producing relevance scores for each document in a corpus with respect to a particular query and presenting the top-scored items. Indeed, the recommend task can be viewed as an information retrieval problem in which the domain of items (the corpus) is queried with the user's preference profile [3].

1.2.2 Content Based Recommendation:

Pure Collaborative Filtering recommenders only utilize the user ratings matrix, either directly, or to induce a collaborative model. These approaches treat all users and items as atomic units, where predictions are made without regard to the specifics of individual users or items. However, one can make a better personalized recommendation by knowing more about a user, such as demographic information, or about an item, such as the director and genre of a movie. For instance, given movie genre information, and knowing that a user liked "Star Wars" and "Blade Runner", one may infer a predilection for Science Fiction and could hence recommend "Twelve Monkeys".

1.2.3 Knowledge Based Recommendation:

This technique uses the knowledge about products and users' needs for making recommendations. Recommendation is made by matching the similarity between user's preference and product description. It does not suffer from ramp up problem since it does not depend on the ratings given by user. This system needs a database and needs to be updated for making useful recommendations [3].

1.2.4 Outside The Box Recommendation:

The problem of overspecialization is overcome using OTB recommendations, and helps to make fresh discoveries. This technique uses a concept called item region. Region (i.e. the "box") is defined as the group of similar items. Regions are created based on similarity distances between items. Stickiness is user's familiarity to a region. Based on the stickiness the system finds items that are not familiar to the user and recommends those items. This technique increases the novelty [3].

1.2.5 Graph Based Recommendation:

Most of the recommender system uses two dimensional information like user and item. Homogeneous and heterogeneous graphs can be used which provides the capability to deal with multidimensional information like user's intentions. A graph-theoretic approach based on maximum flow or maximum bipartite computations represent user and item as vertices and the flow is calculated. This technique improves aggregate diversity of recommendations. One of the major drawback with the graph-based approach is when the input data becomes large it becomes tedious to rebuild graph [4].

1.2.6 Hybrid Recommendation:

Hybrid recommendation uses a combination of recommendation technique. Content based recommendation and collaborative filtering recommendation is the commonly used combination. Both the system suffers from ramp up problem. The disadvantages in both the techniques can be overcome by combining them in parallel or cascade. Both the rating and the profile data can be used for finding recommendations. This technique improves the recommendation accuracy but diversity is not considered [3].

1.3 Introduction to Collaborative Filtering

Collaborative filtering (CF) techniques have become well-known through their use in on-line stores such as Amazon to recommend items to users based on buying patterns. The underlying assumption is that users who bought the same items in the past are likely to do so in the future. CF systems can either be user based or item-based. In user-based CF, the system attempts to match users based on system-maintained profiles that may be a combination of explicitly supplied preferences and patterns derived from their actions. In contrast, item-based CF performs the filtering by matching items based on either similarity of features or simple association. For example, Amazon maintains a correlation matrix for items that indicates which items are related by the fact that a user purchased both of them.

Although there are many CF variants, what is common to current techniques is that they base their filtering on some form of stored profiles of users and/or items. These profiles may be based on information provided explicitly by the users or the producers of the items, but, since this is typically a tedious task, it is generally preferable to find some means of automatically generating profiles based on user actions. The profiles will typically be stored on a server and, on receipt of a request, a CF algorithm will be

applied to perform the filtering and return the items most likely to be of interest to the user.

We were interested in providing CF in a mobile tourist information system designed to support visitors to an arts festival. Our goal was that tourists should be able to carry their own personal copy of the festival information system with them on a portable computer, but somehow be recommended events, venues, bars and restaurants based on ratings and reviews from other tourists. The system differs from traditional systems with CF in two ways. First, the system is designed so that it can deliver context-aware information to tourists without the need to have a network connection and therefore we did not want to have to introduce a server into the architecture. Second, unlike on-line stores, user and item profiles cannot evolve gradually over time. Visitors to the festival will want a filtering of relevant information and recommendations as soon as they arrive at the festival for what might be a visit of only one or two days.

We therefore decided to introduce a new variant of user-based CF that matches users based on social contexts rather than user interactions with the system. The idea is that users who go to the same place at the same time tend to have similar tastes. So if one festival visitor likes a particular style of trendy bar, then they may share other interests with the people who also like that bar such as theatrical events or contemporary restaurants. We therefore filter ratings and reviews based on spatial-temporal proximity of users as a basis for the formation of social networks. This works by users carrying their ratings and reviews with them and they are shared with other users in an opportunistic way based on ad-hoc network connections.

1.3.1 Types of Collaborative Filtering:-

1.3.1.1 Model-based Collaborative Filtering

Model-based techniques provide recommendations by estimating parameters of statistical models for user ratings. For example, describe an early approach to map CF to a classification problem, and build a classifier for each active user representing items as feature vectors over users and available ratings as labels, possibly in conjunction with dimensionality reduction techniques to overcome data sparsity issues.

1.3.1.2 User–User Collaborative Filtering

User–user collaborative filtering, also known as k-NN collaborative filtering, was the first of the automated CF methods. It was first introduced in the Group Lens Usenet article recommender. The Ringo music recommender and the Bell Core video recommender also used user-user CF or variants thereof. User–user CF is a straightforward algorithmic interpretation of the core premise of collaborative filtering: find other users whose past rating behavior are similar to that of the current user and use their ratings on other items to predict what the current user will like.

1.3.1.3 Item-Based Top-N Recommendation

In this section, we study a class of model-based top-N recommendation algorithms that use item-to-item similarities to compute the relations between the different items. The primary motivation behind these algorithms is the fact that a customer is more likely to purchase items that are similar to the items that he/she has already purchased in the past; thus, by analyzing historical purchasing information (as represented in the user–item matrix) we can automatically identify these sets of similar items and use

them to form the top-N recommendations. These algorithms are similar in spirit to previously developed item-based schemes but differ in a number of key aspects related to how the similarity between the different items is computed and how these similarities are combined to derive the final recommendations [3].

1.4 Introduction to Similarity

1.4.1 Pearson correlation.

This method computes the statistical correlation (Pearson's r) between two user's common ratings to determine their similarity. Group Lens and Bell Core both used this method. The correlation is computed by the following:

$$s(u, v) = \frac{\sum_{i \in I_u \cap I_v} (r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I_u \cap I_v} (r_{u,i} - \bar{r}_u)^2 \times \sum_{i \in I_u \cap I_v} (r_{v,i} - \bar{r}_v)^2}} \quad (1)$$

Pearson correlation suffers from computing high similarity between users with few ratings in common. This can be alleviated by setting a threshold on the number of co-rated items necessary for full agreement (correlation of 1) and scaling the similarity when the number of co-rated items falls below this threshold. Experiments have shown a threshold value of 50 to be useful in improving prediction accuracy, and the threshold can be applied by multiplying the similarity function by $\min \{|I_u \cap I_v|/50, 1\}$.

1.4.2 Constrained Pearson correlation.

Ringo solicited ratings from its users on a 7-point scale, providing a rating guide that fixed 4 as a neutral (neither like nor dislike) value r_z . With an absolute reference, it is possible to correlate absolute like/dislike rather than relative deviation (as the

standard Pearson r does). This led Shardanand and Maes to propose the constrained Pearson correlation:

$$s(u, v) = \frac{\sum_{i \in I_u \cap I_v} (r_{u,i} - r_z)(r_{v,i} - r_z)}{\sqrt{\sum_{I \in I_u \cap I_v} (r_{u,i} - r_z)^2 \times \sum_{I \in I_u \cap I_v} (r_{v,i} - r_z)^2}} \quad (2)$$

1.4.3 Cosine similarity.

This model is somewhat different than the previously described approaches, as it is a vector-space approach based on linear algebra rather than a statistical approach. Users are represented as $|i|$ -dimensional vectors and similarity is measured by the cosine distance between two rating vectors. This can be computed efficiently by taking their dot product and dividing it by the product of their L2 (Euclidean) norms:

$$s(u, v) = \frac{r_u \cdot r_v}{\|r_u\|_2 \|r_v\|_2} = \frac{\sum_i r_{u,i} \cdot r_{v,i}}{\sqrt{\sum_i r_{u,i}^2 \cdot \sum_i r_{v,i}^2}} \quad (3)$$

Unknown ratings are considered to be 0; this causes them to effectively drop out of the numerator. If the user mean baseline is subtracted from the ratings prior to computing the similarity, cosine similarity is equivalent to Pearson correlation when

the users have rated the same set of items and decreases as $\frac{|I_u \cap I_v|^2}{|I_u| |I_v|}$ decreases [2].

1.5 Examples of Recommender System

Here we present some of the e-commerce businesses which use the recommendation system as a tool to boost their business areas and to get an idea of about the customers' likings. Here we explain how each different websites works with their type of recommendation system.

1.5.1. Flipkart.com

We focus here on the mobile and tablet section of the website. Like every other e-commerce websites, flipkart.com also maintains a list of customers who have brought the product in their every mobile and tablet product listed page. This way they are able to recommend products in two ways. First way recommends mobiles and tablets frequently purchased by customers who purchased the selected mobile. The second recommends company of the mobile whose phones and tablets are frequently purchased by customers who purchased the phone or tablet of same company. Another feature which helps flipkart.com to recommend products is its content matcher feature. According to this feature, customers have to rate those phones and tablets which are being used by the customer in a 5-point-scale, from dissatisfactory to excellent. After rating various phones and tablets used by the customer, now they want to get recommended some new phones. This recommendation is achieved by showing phones which are correlated with the users' likings and choice. The customer feedback or say the customer rating is the most important thing for an accurate recommendation, as they form the base for future customers to buy the same phone.

1.5.2. Netflix.com

Netflix.com is the most used website which deals with movies and TV series. Their widespread collection of movies and TV series are incredible which also makes their task to manage these difficult. A user is generally confused as to which movies they

should watch which would suit their taste from thousands of the listed movies and TV series. They help us to support our decision in watching the movie with help of the mood, genre or theme of the movie demanded by the user. They have two types of recommendations. Firstly the recommendation is based on the global scale. This means that they recommend movies which are being frequently watch and liked by the user currently on a global scale irrespective of their genre, theme and all. This are like top 10 movies of the day and their data may vary from hour to hour as per the demand of particular movie. Second recommendation method used by Netflix.com is the personalized recommendation technique. This technique considers the multiple factors for getting movies recommended which include the customer's rating, IMDB rating of the movie, genre of the movie, mood of the person, and theme of the movie.

1.5.3 Levis Style Finder

Style Finder allows customers of the Levi Straus™ (www.levis.com) website to receive recommendations on articles of Levi's clothing. Customers indicate whether they are male or female, then view three categories -- Music, Looks, Fun -- and rate a minimum of 4 "terms" or "sub-categories" within each. They do this by providing a rating on a 7-point scale ranging from "leave it" to "love it." They may also choose the rating of "no opinion." Once the minimum number of ratings is entered customers may select "get recommendations." Here, they are provided with thumbnails of 6 items of recommended clothing. Customers may provide feedback by use of the "tell us what you think feature" which allows them to enter an opinion rating for the recommended article of clothing. Feedback may change one or all of the six items recommended.

CHAPTER -2

LITERATURE REVIEW

This chapter gives an overview of literature survey made in context to the proposed thesis work. From the study of research paper various conclusions are made:-

2.1 Base Paper [1]

J. Ben Schafer, Joseph Konstan, John Riedl, “Recommender Systems in E-Commerce”

Recommender systems are changing from novelties used by a few E-commerce sites, to serious business tools that are re-shaping the world of E-commerce. Many of the largest commerce Web sites are already using recommender systems to help their customers find products to purchase. A recommender system learns from a customer and recommends products that she will find most valuable from among the available products. In this paper we present an explanation of how recommender systems help E-commerce sites increase sales, and analyze six sites that use recommender systems including several sites that use more than one recommender system. We conclude with ideas for new applications of recommender systems to E-commerce.

Recommender systems enhance E-commerce sales in three ways:

Browsers into buyers: Visitors to a Web site often look over the site without ever purchasing anything. Recommender systems can help customers find products they wish to purchase.

Cross-sell: Recommender systems improve cross-sell by suggesting additional products for the customer to purchase. If the recommendations are good, the average

order size should increase. For instance, a site might recommend additional products in the checkout process, based on those products already in the shopping cart.

Loyalty: In a world where a site's competitors are only a click or two away, gaining customer's loyalty is an essential business strategy. Recommender systems improve loyalty by creating a value-added relationship between the site and the customer. Sites invest in learning about their users, use recommender systems to operationalize that learning, and present custom interfaces that match customer needs. Customers repay these sites by returning to the ones that best match their needs.

Finding Recommendations

Just as sites can utilize different methods for calculating or displaying recommendations, so can they utilize different methods for allowing customers to access the recommendations? Through our recommender system examples we have identified four different methods for finding recommendations each of which may provide access to more than one recommendation interface and/or technology. These four methods are ordered in the amount of customer effort required to find the recommendations.

Organic Navigation: Requiring the least amount of work to actually access recommendations is the organic navigation process. In applications such as Album Advisor, Movie Matches, and Feedback Profile, customers do nothing extra in order to receive recommendations. In each of these applications, recommendations appear as part of the item information page. These recommendations can consist of additional items to consider, average ratings, or a list of other customer comments. However, the underlying similarity is that through the course of normal navigation of the site, customers are provided with a recommendation.

Selection Options: In the selection options process customers must truly interact with the system in order to receive recommendations. Typically, customers choose from a set of predefined criterion/options upon which to base their recommendations. For example, users of Amazon.com Delivers have a choice from nearly 50 pre-defined categories in which to receive periodic recommendations. Even more involved, users of Moviefinder.com's We Predict system can select from a finite list of title, format, length and genre options to define a search, as well as customizing options such as ranking method and display features.

Keyword/Freeform: Arguably, the keyword/freeform option requires the most interaction from the customer. In applications such as Eyes, customers provide a set of textual keywords upon which to retrieve future recommendations. A version of Album Advisor takes the freeform input of multiple artists upon which to make recommendation matches. The We Predict and Movie Map applications produce recommendations from the results of a query conducted using the keywords provided. While each uses the keywords in very different manners, each requires the user to know specifically what types of things they are interested in.

2.2 Base Paper [2]

Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl, "Item-based collaborative filtering recommendation algorithms"

The amount of information in the world is increasing far more quickly than our ability to process it. All of us have known the feeling of being overwhelmed by the number of new books, journal articles, and conference proceedings coming out each year. Technology has dramatically reduced the barriers to publishing and distributing information. Now it is time to create the technologies that can help us sift through all the available information to and that which is most valuable to us. One of the most

promising such technologies is collaborative Filtering. Collaborative Filtering works by building a database of preferences for items by users. A new user, Neo, is matched against the database to discover neighbors, which are other users who have historically had similar taste to Neo. Items that the neighbors like are then recommended to Neo, as he will probably also like them. Collaborative filtering has been very successful in both research and practice, and in both information Filtering applications and E-commerce applications. However, there remain important research questions in overcoming two fundamental challenges for collaborative filtering recommender systems.

Challenges of User-based Collaborative Filtering Algorithms

User-based collaborative filtering systems have been very successful in past, but their widespread use has revealed some potential challenges such as:

Sparsity. In practice, many commercial recommender systems are used to evaluate large item sets (e.g., Amazon.com recommends books and CDnow.com recommends music albums). In these systems, even active users may have purchased well under 1% of the items (1% of 2 million books is 20,000 books). Accordingly, a recommender system based on nearest neighbor algorithms may be unable to make any item recommendations for a particular user.

Scalability. Nearest neighbor algorithms require computation that grows with both the number of users and the number of items. With millions of users and items, a typical web-based recommender system running existing algorithms will suffer serious scalability problems.

CHAPTER - 3

HARDWARE AND SOFTWARE SPECIFICATION

3.1 Hardware Requirements:

Factors	Specification
RAM	1GB
Hard Disk	80GB
Processor	1GHz

3.2 Software Requirements:

Factors	Specification
Operating System	Windows XP, 7, 8
Languages	PHP, HTML
Server	Apache Web Server (XAMPP Control Panel)
Database	MySQL
IDE	Aptana Studio

CHAPTER-4

PROPOSED WORK

4.1 Existing Problem

- 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 real time, 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.

Obtaining recommendations from trusted sources is a critical component of the natural process of human decision making. With burgeoning consumerism buoyed by the emergence of the web, buyers are being presented with an increasing range of choices while sellers are being faced with the challenge of personalizing their advertising efforts. In parallel, it has become common for enterprises to collect large volumes of transactional data that allows for deeper analysis of how a customer base interacts with the space of product offerings.

For example, movie watchers on Netflix frequently provide ratings on a scale of 1 (disliked) to 5 (liked). Such a data source records the quality of interactions between users and items. Additionally, the system may have access to user-specific and item-specific profile attributes such as demographics and product descriptions respectively.

Recommender Systems have evolved to fulfill the natural dual need of buyers and sellers by automating the generation of recommendations based on data analysis.

Good product recommendations are key to successful online retailing, they help customers' find products and induce cross-selling. In practice, many recommendations are based on counting co-occurrences between (combinations of) products. In such an approach it is difficult to use the entire purchase history of a customer as a whole, due to the sparse nature of purchase data. In addition, it is hard to incorporate additional information at the customer level [4].

4.2 Existing Technique:

A Pure Collaborative filtering bases its predictions and recommendations on the ratings or behavior of other users in the system. The fundamental assumption behind this method is that other users' opinions can be selected and aggregated in such a way as to provide a reasonable prediction of the active user's preference. Intuitively, they assume that, if users agree about the quality or relevance of some items, then they will likely agree about other items — if a group of users likes the same things as Mary, then Mary is likely to like the things they like which she hasn't yet seen. For this pure collaborative filtering component uses a *neighborhood-based algorithm*. In neighborhood-based algorithms, a subset of users is chosen based on their similarity to the active user, and a weighted combination of their ratings is used to produce predictions for the active user. The algorithm we use can be summarized in the following steps:

- a. Weight all users with respect to similarity with the active user.
 - Similarity between users is measured as the Pearson correlation between their ratings vectors.

b. Select n users that have the highest similarity with the active user.

- These users form the *neighborhood*.

c. Compute a prediction from a weighted combination of the selected neighbors' ratings.

In step 1, similarity between two users is computed using the Pearson correlation coefficient, defined by

$$s(u, v) = \frac{\sum_{i \in I_u \cap I_v} (r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I_u \cap I_v} (r_{u,i} - \bar{r}_u)^2 \times \sum_{i \in I_u \cap I_v} (r_{v,i} - \bar{r}_v)^2}} \quad (1)$$

where $r_{a,i}$ is the rating given to item i by user a ; \bar{r}_a is the mean rating given by user a ; and m is the total number of items.

In step 3, predictions are computed as the weighted average of deviations from the neighbor's mean:

$$P_{a,i} = \bar{r}_a + \frac{\sum_{u=1}^n (r_{u,i} - \bar{r}_u) \times P_{a,u}}{\sum_{u=1}^n P_{a,u}} \quad (4)$$

Where $p_{a,i}$ is the prediction for the active user a for item i ; $P_{a,u}$ is the similarity between users a and u ; and n is the number of users in the neighborhood.

It is common for the active user to have highly correlated neighbors that are based on very few co-rated (overlapping) items. These neighbors based on a small number of overlapping items tend to be bad predictors. To devalue the correlations based on few co-rated items, we multiply the correlation by a *Significance Weighting* factor. If two users have less than 50 co-rated items we multiply their correlation by a factor $S_{ga,u} =$

$n/50$, where n is the number of co-rated items. If the number of overlapping items is greater than 50, then we leave the correlation unchanged i.e. $s_{ga,u} = 1$ [2].

The Complete task of Collaborative Filtering process can be explained using the figure

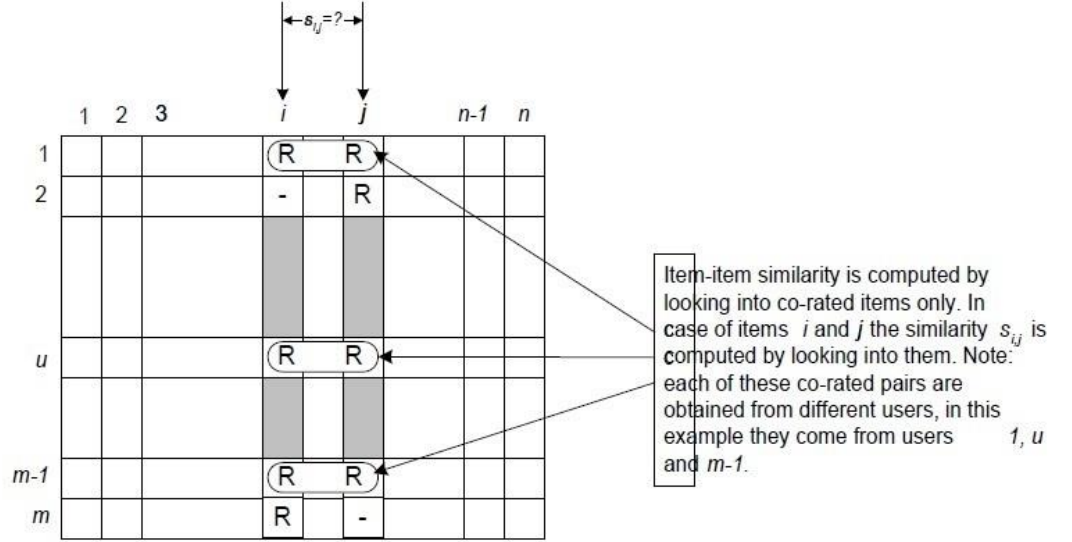


Figure 4.1: Isolation of the co-rated items and similarity computation

4.3 Proposed Solution:

Overcoming the First-Rater Problem In pure CF a prediction cannot be made for an item, for the active user, unless it was previously rated by other users. However, we can make such a prediction using a content based predictor for the user. Using collaborative filtering we can further improve the predictions by utilizing the content-based predictions of other users as well. If the neighbors of the active user are highly correlated to it, then their predictions should also be very relevant to the user. This is particularly true if neighbors have rated many more items than the active user; because their predictions are likely to be more accurate than the active user's.

4.3.1 Tackling Sparsity

In collaborative filtering, since we use a pseudo ratings matrix, which is a full matrix, we eliminate the root of the sparsity problem. Pseudo user-ratings vectors contain ratings for all items; and hence all users will be considered as potential neighbors. This increases the chances of finding similar users. Thus the sparsity of the user-ratings matrix affects collaborative filtering to a smaller degree than CF.

4.3.2 Finding Better Neighbors

A crucial step in CF is the selection of a neighborhood. The neighbors of the active user entirely determine his predictions. It is therefore critical to select neighbors who are most similar to the active user. In pure CF, the neighborhood comprises of the users that have the best n correlations with the active user. The similarity between users is only determined by the ratings given to co-rated items; so items that have not been rated by both users are ignored. However, in collaborative filtering, the similarity is based on the ratings contained in the pseudo user-ratings vectors; so users do not need to have a high overlap of co-rated items to be considered similar. Our claim is that this feature of collaborative filtering makes it possible to select a better, more representative neighborhood.

4.3.3 Improving Collaborative Filtering

Due to the nature of our hybrid approach, we believe that improving the performance of the individual components would almost certainly improve the performance of the whole system. In other words, if we improved our pure content-based predictor or the CF algorithm, we would be able to improve our system's predictions. A better content based predictor would mean that the pseudo ratings matrix generated would more accurately approximate the actual full user-ratings matrix. This in turn, would

improve the chances of finding more representative neighbors. And since the final predictions in our system are based on a CF algorithm, a better CF algorithm can only improve our system's performance. In our current implementation of the content-based predictor, we use a naive Bayesian text-classifier to learn a six way classification task. This approach is probably not ideal, since it disregards the fact that classes represent ratings on a linear scale. This problem can be overcome by using a learning algorithm that can directly produce numerical predictions [4].

4.4 Proposed Algorithm:

The complete proposed work can be classified into five main broad categories. They include:

- a. Getting user rating from user and other factors.
- b. Create Pseudo user-rating vector.
- c. Finding the individual similarity coefficient for both the factors.
- d. Finding a cumulative similarity coefficient.
- e. Finally getting the prediction or recommendation.

4.4.1. Getting user rating from user and other factors.

The most important factor which is needed in the Collaborative filtering process of recommendation is the rating of the active user. These ratings are the main backbone of the process. The information domain for a collaborative filtering system consists of *users* which have expressed preferences for various *items*. A preference expressed by a user for an item is called a *rating* and is frequently represented as a (*User*, *Item*, *Rating*) triple. These ratings can take many forms, depending on the system in question. Some systems use real- or integer-valued rating scales such as 0–5 stars, while others use binary or ternary (like/dislike) scales.¹ Unary ratings, such as “has purchased”, are particularly common in e-commerce deployments as they express

well the user's purchasing history absent ratings data. When discussing unary ratings, we will use "purchased" to mean that an item is in the user's history, even for non-commerce settings such as web page views. Also we can add the factors such as the cost and category of the product for the better and more sensible recommendation. Generally the factors considered are of positive type. We can also increase the accuracy of the recommendation system by using negative factors such as 'item bought and returned back', 'technical problems' etc.

4.4.2. Create Pseudo user-rating vector.

In the proposed work, we create a *pseudo user-ratings vector* for every user u in the database. The user-ratings database, which is a list of users versus items, where each cell is the rating given by a user to an item. We call each row of this list as a *user-ratings vector*. The user-ratings matrix is very sparse, since most items have not been rated by most users. The proposed work aims on each user-ratings vector and a thus pseudo user-ratings vector is created. A pseudo user-ratings vector contains the user's actual ratings and for the unrated items the rating is usually taken in corresponding to a fixed factor such as cost of the product. All pseudo user-ratings vectors put together form the pseudo ratings matrix, which is a full matrix. Now given an active user's ratings, predictions are made for a new item using Collaborative Filtering on the full pseudo ratings matrix. The pseudo user-ratings vector, v_u , consists of the item ratings provided by the user u , where available, and those predicted by the corresponding other factors otherwise.

$$v_{u,i} = \begin{cases} r_{u,i} : \text{if user } u \text{ rated item } i \\ c_{u,i} : \text{otherwise} \end{cases} \quad (5)$$

In the above equation $r_{u,i}$ denotes the actual rating provided by user u for item i , while $c_{u,i}$ is the rating predicted in corresponding to other factor's values. The pseudo user-

ratings vectors of all users put together give the dense pseudo ratings matrix V . We now perform collaborative filtering using this dense matrix. The similarity between the active user a and another user u is computed using the Pearson correlation coefficient described in Equation 1. Instead of the original user votes, we substitute the votes provided by the pseudo user-ratings vectors v_a and v_u .

The complete work is then also dealt with other factors taken into consideration, i.e. we create a complete list of items and user with all the items being rated in their own factors. Thus, now the complete work to be carried out on these set of pseudo user-rating values of the products instead of the original ratings rated by the user.

4.4.3. Finding the individual similarity coefficient for both the factors.

This step is important as most of the recommendation works depends on the successful completion of the step. The whole and sole concept of collaborative filtering process is completely dependent on finding the suitable similarity coefficient between the users. This is done by computing the neighborhood algorithm to compute a neighborhood $N \subseteq U$ of neighbors of u . The basic idea in similarity computation between two items i and j is to first isolate the users who have rated both of these items and then to apply a similarity computation technique to determine the similarity s_{ij} . Here too we use the same concept of finding the similarity known as the Pearson Correlation Similarity Coefficient. The Pearson Correlation Similarity Coefficient is computed by Equation 1.

Now this algorithm is used to find the similarity coefficient for all the factors which we have considered in step 1. The similarity coefficient of all the factors is then recorded for our future transactions.

4.4.4. Finding a cumulative similarity coefficient.

After computing the various similarity coefficients of the many factors taken under consideration, the main problem arises as which coefficient to be used to find the recommendations. As all the factors constitute an important significance in the prediction so it is better that we take all of them into consideration but in a cumulative manner. Thus we base the cumulative structure of the similarity coefficient for finding the predictions for the users.

The cumulative value of similarity coefficient of x factors for n users and I items can be found by simply finding the mean of the similarity coefficients. This is given by:

$$\bar{x} = \frac{\sum_{f=1}^x sim(i,j)}{x} \quad (6)$$

Where, f is the number of factors taken in account, $sim(i,j)$ is the similarity coefficient of i users and j items for a single factor.

4.4.5. Finally getting the prediction or recommendation.

The most important step in this complete recommendation system is to generate output interface in terms of predictions. Here we recommend the product for a user X by computing the similarity distance of the user with all the other users. After getting the similarity coefficient we check to find the user with closest distance to it. This can be done by isolating the user X with respect to other users. This isolation means that we find those users who have also rated the products with more or less same rating value. Once we isolate the users then the simple task remains that the products of those isolated user with more or less same value is then being recommended for user X .

CHAPTER – 5

METHODOLOGY

Today, with the increase of e commerce websites and even more marketing led the foundation stone for starting the research work on this domain. The complete work of the project report began with the describing the problem. While researching the content on it we, went through the collaborative filtering technique and more precisely the neighborhood based technique for recommendation in these e-commerce websites. The problem which is considered in the complete research paper is the problems which were faced by the collaborative filtering technique (considered as one of the best recommendation technique), which include the cold start problem and the sparsity problem. Due to these problems there was difficulty in predicting the products to the user, in non-common sense way. Then to resolve these two main problems and to get a more common sensible prediction we developed a model approach to solve the problem. To complete the model, the first and foremost thing which is necessary is to collect data of the product, its cost and the user rating of user u for an item i . In our model we have dealt with the data part using the book catalogue. We had the book's name entered along with ISBN number, author of the book, publisher and the cost of the book entered. Once we have our complete data on the book we then create different user account and allow the users to rate the books which ever they have read. Instead of dealing only with customer's feedback as the basis for find the neighbor we considered other factors too both positive and negative ones. This way it not only resolved the problem of cold start but also gave more accuracy to the prediction technique. In our model we considered only two factors namely the user rating and the cost of the book. Now to solve the sparsity problem we developed a pseudo user rating vector. This vector is a virtual matrix, in which all the

data of all items by all users are created, even if they have not rated some items. This done by predictions and involvement of other factors. And then we predict the neighboring users using individual factors in consideration. Then we find a cumulative factor using these factor similarity coefficients. Using this information in our hand we predict the recommendations for user. After the complete model is designed and implemented it was put under the testing. Initially we didn't get the desired results, so we had to modify the model a bit and then again put the model to test. Once the model solution properly resulted the desired output, we then implemented the complete model for verification. The complete process can be depicted by:-

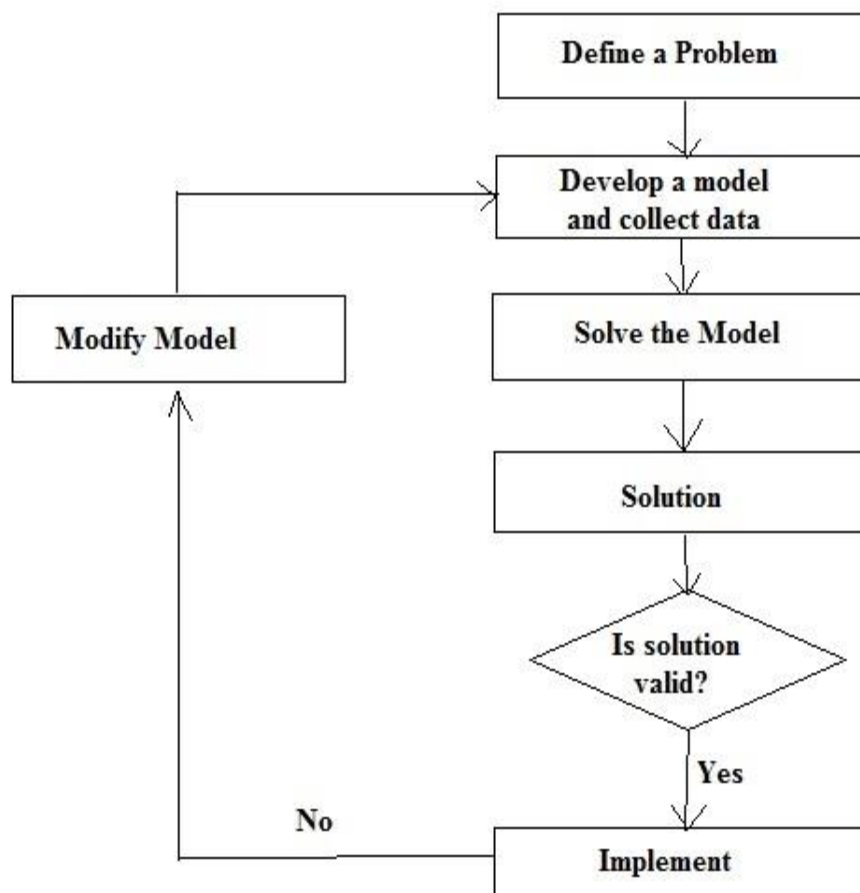


Fig 5.1: Methodology Diagram of the Project

CHAPTER – 6

IMPLEMENTATION

The Smart Product Recommendation performs several operations like getting the user rating from user, finding similarity distance and getting the prediction of items. The implementation process follows the different algorithms defined for these operations. All these algorithms mentioned here help us to provide a better and accurate recommendation system. The algorithms for these operations are given below.

6.1 Algorithms Used

6.1.1. Pearson Correlation Similarity Algorithm

```
1. PearsonCorrelation (xarr[], yarr[])
/*Computes the similarity distance between the users for a preferred item*/
2. {
3.   for i= 0 to xarr.size
4.     {
5.       meanX += xarr.elementAt(i)
6.       meanY += yarr.elementAt(i)
7.     }
8.   meanX /=xarr.size
9.   meanY /=yarr.size
10.  for i=0 to xarr.size
11.    {
12.      sumXY += ((xarr.elementAt(i) - meanX) *
(yarr.elementAt(i) - meanY))
13.      sumX2 += square(xarr.elementAt(i) - meanX
14.      sumY2 += square(yarr.elementAt(i) - meanY
15.    }
16.  sim=sumXY / (squareroot(sumX2) * squareroot(sumY2))
17.  return(sim)
18. }
```

6.1.2 Get Recommendation Algorithm

```
function getRecommendations(preferences, person)
/*Computes and find the recommendations*/
{
  Foreach (preferences= start_user till end_user)
  {
    if(otherPerson != person)
    {
      sim = PearsonCorrelation(person,otherperson)
    }
    if(sim > 0)
    {
      foreach(preferences=otherPerson] till keyvalue)
      {
        if(!array_key_exists(key, preferences[person]))
        {
          if(!array_key_exists(key, total))
          {
            total[key] = 0;
          }
          total[key] += preferences[otherPerson][key] * sim;
          if(!array_key_exists(key, simSums))
          {
            simSums[key] = 0;
          }
          simSums[key] += sim;
        }
      }
    }
  }
  foreach(total = till keyvalue)
  {
    ranks[key] = value / simSums[key];
  }
  array_multisort(ranks, SORT_DESC);
  return ranks;
}
```

6.2 Context Diagrams

6.2.1 UML Diagram:

The UML Diagram consists of 5 major blocks i.e. Request Handler, Connection Handler, Response Handler, Rating Model and Rating Manager. The purpose of Request Handler is to send request to the Connection Handler for extracting the required data from the database which then process a remote response to Response Handler which notifies to the Rating Model. It also helps in getting the ratings of the product. Rating Manager helps in managing all the operations of rating a product. After the rating of all the products is being done, the Request Handler gets the required products which are recommended to the user [5].

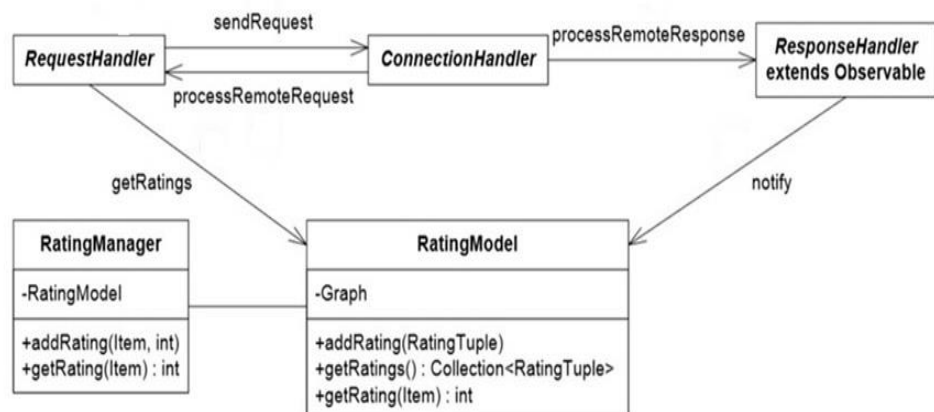


Figure 6.1: UML Diagram

6.2.2 State Diagram:

The State Diagram of the project consists of the following blocks like Enter Query, Recommender System and Data Extracted from the Dataset. Initially the user enters the query i.e. he/she asks for the books to be recommended after the usual operations of rating them. Then the flow of the diagram moves to Recommender System which in turn checks whether the data is present in the dataset according to the needs of the user. If the data is not available, then it moves to No Data [5].

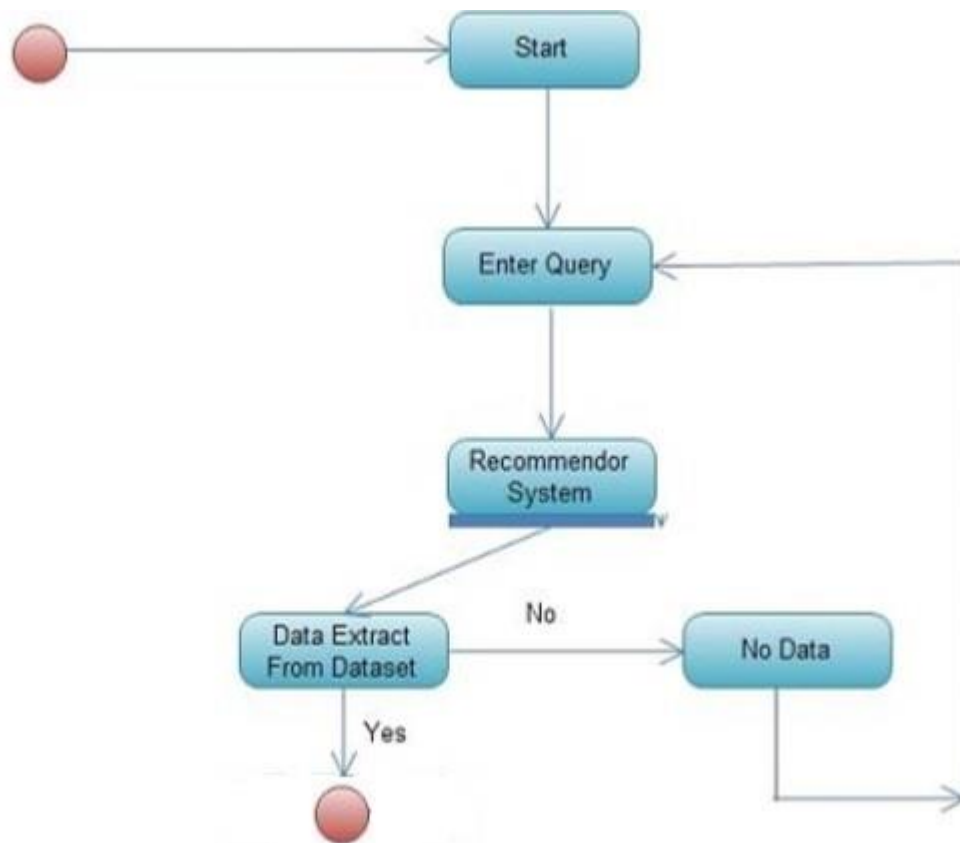


Figure 6.2: State Diagram

6.2.3 Use Case Diagram:

The Use Case Diagram of the project involves 4 set of operations which consists of Enter Query, Receive Recommendation and Data Extraction. Initially, the user interacts with Enter Query module, whose main objective is to take the necessary steps in order to get recommendation. In Receive Recommendation, the user gets the required recommendation according to the ratings of all the users. Data Extraction involves two major sub blocks namely AOL Data Sets and Image DataSet [6].

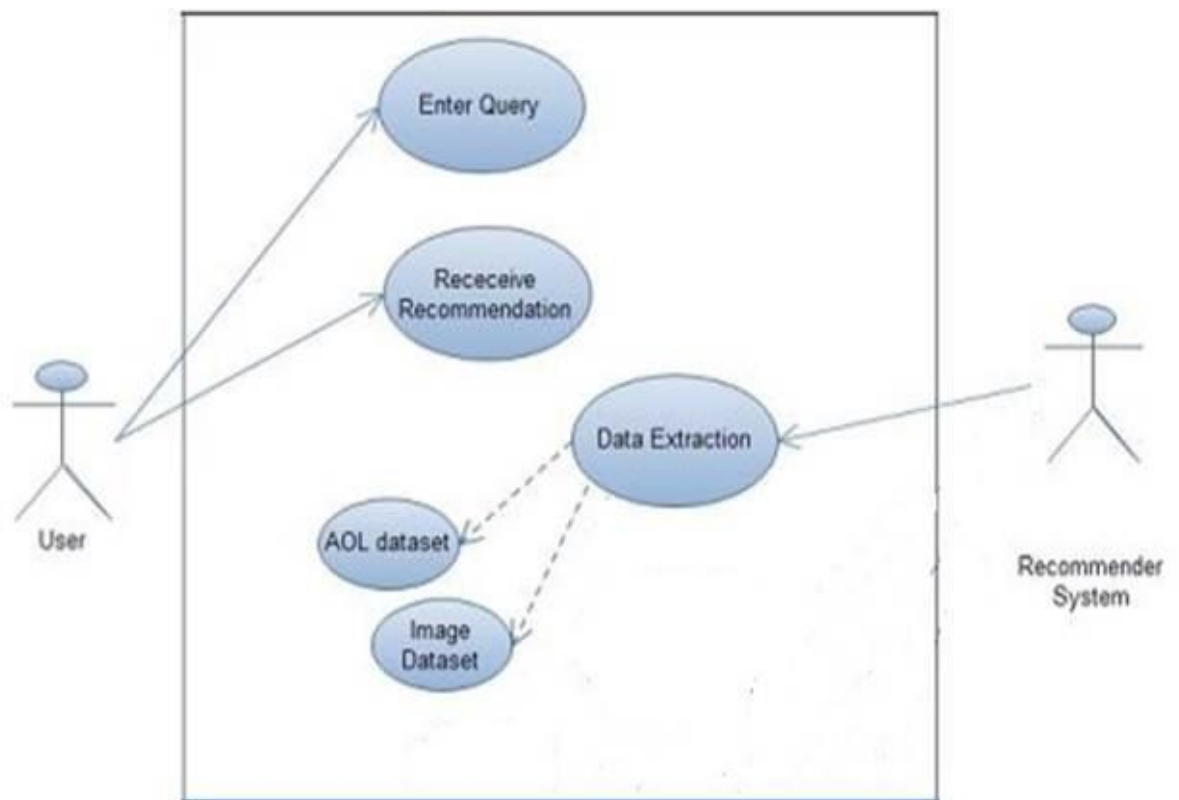


Figure 6.3: Use Case Diagram

6.2.4 Data flow diagram

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. DFDs can also be used for the visualization of data processing (structured design). DFD shows what kinds of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored.

6.2.4.1) Level 0 Diagram

Level 0 diagram shows all the processes that comprise the overall system; it also shows how information moves from and to each process and finally Adds data stores. Following figure shows all the processes which are in our recommender system, namely user, recommender system in middle and recommended query and finally a click-through data, which is a historical data stored in a database.

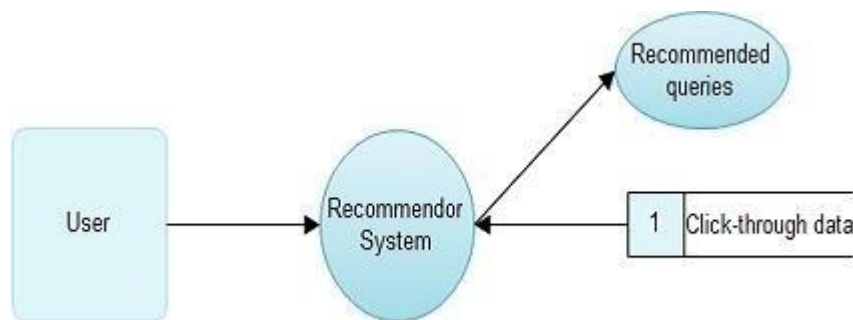


Fig 6.4 : Level 0 DFD

6.2.4.2) Level 1 Diagram

Level 1 diagram shows all the processes that comprise a single process on the level 0 diagram. It also shows how information moves from and to each of these processes and more detail the content of higher level process. Level 1 diagram may not be needed for all level 0 processes.

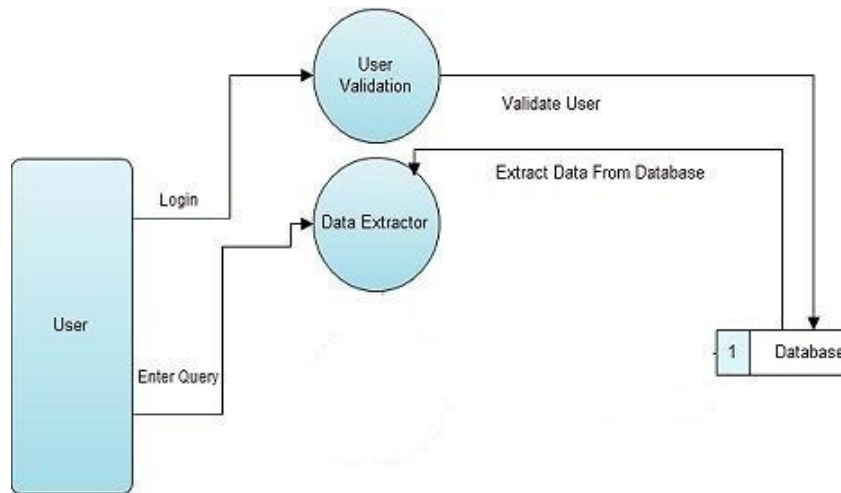


Fig 6.5: Level 1 DFD

6.2.4.3) Level 2 Diagram

There can be one level 2 DFD for each process of the Level 1 DFD. Level 2 shows a process broken down into greater detail. Level 2 Diagrams are only necessary where the Level 1 process is more complex, and where the particular process is relevant to the analysis. Following figure shows a level 2 diagram of our system.

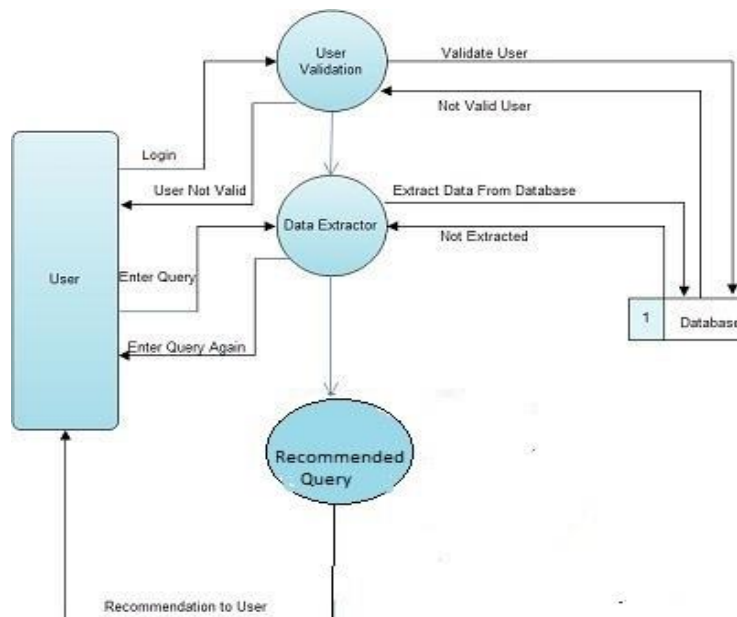
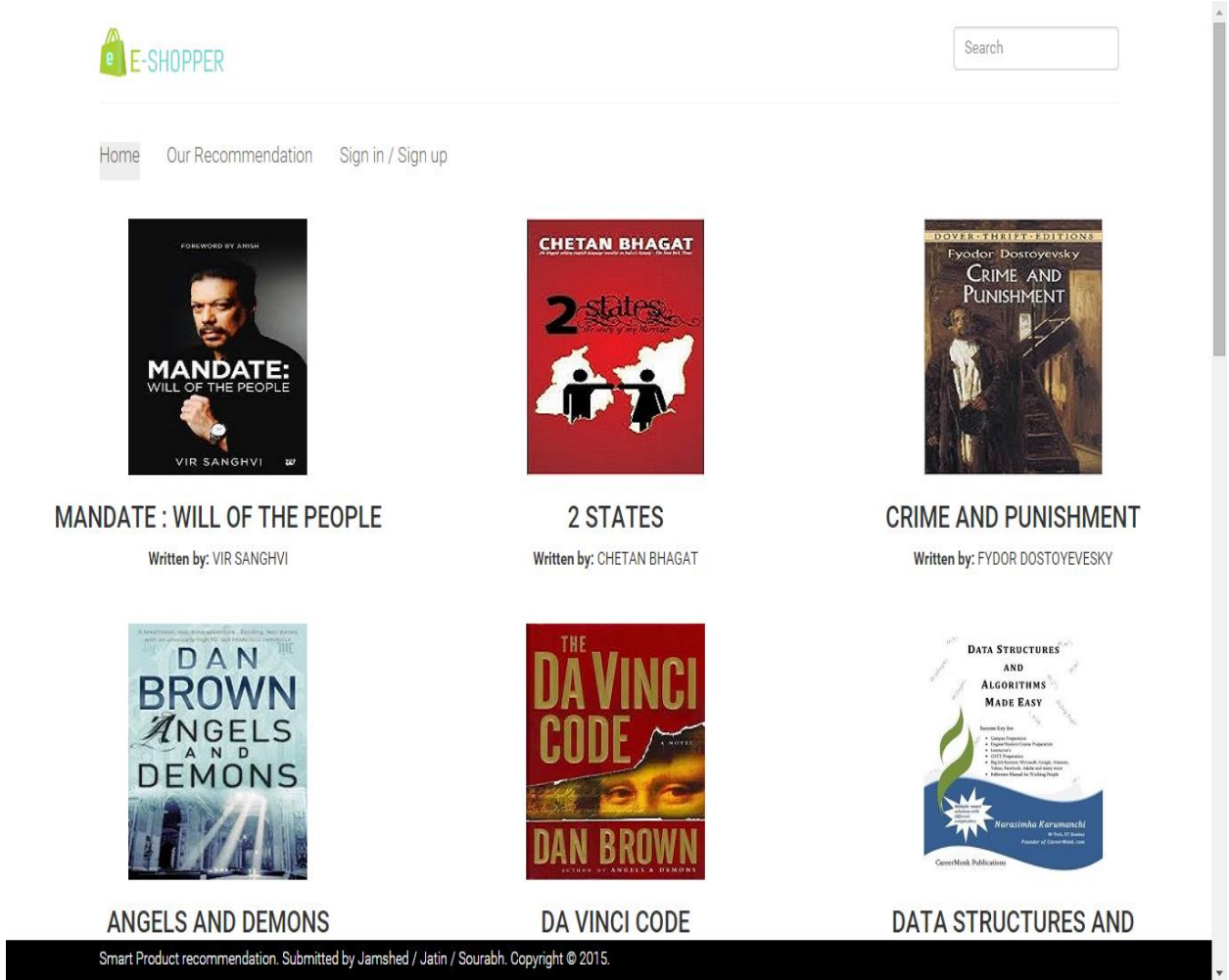


Fig 8: Level 6.6 DFD

CHAPTER -7

RESULT

7.1 Home Page:




The screenshot displays the 'e-SHOPPER' website's home page. At the top left is the 'e-SHOPPER' logo, and at the top right is a search bar. Below the header, there are navigation links: 'Home', 'Our Recommendation', and 'Sign in / Sign up'. The main content area features a grid of six book covers with their titles and authors listed below them:

- MANDATE: WILL OF THE PEOPLE** by VIR SANGHVI
- 2 STATES** by CHETAN BHAGAT
- CRIME AND PUNISHMENT** by FYDOR DOSTOYEVSKY
- ANGELS AND DEMONS** by DAN BROWN
- DA VINCI CODE** by DAN BROWN
- DATA STRUCTURES AND ALGORITHMS MADE EASY** by Narasimha Karumanchi

At the bottom of the page, a black banner contains the text: 'Smart Product recommendation. Submitted by Jamshed / Jatin / Sourabh. Copyright © 2015.'

This page shows the available number of books on the e-Shopper portal. Only from these set of books, you will be recommended select books based on your rating as well as the price of the book.

7.2 Sign Up Page:



[Home](#) [Our Recommendation](#) [Sign in / Sign up](#)

Login to your account

User ID*

Password*

Login

OR

New User Signup!

jatin

jatin

.....

.....

Sign up

This page simply urges you to register to E-shopper in order to get the books recommended. It also helps you to Sign In with your current account.

7.3 Add Products:


Name

Name of Product*
ISBN*
Author*
Publisher*
Add

This page helps the administrators to add new products to the E-Shopper portal. The details to be involved are Name of the Product, ISBN, Author and Publisher of the Book. After the details are being done, the product are being added to the database.

7.4 User 1:

Login Page



Search

[Home](#) [Our Recommendation](#) [Sign in / Sign up](#)

Login to your account

jatin

...

Login

OR

New User Signup!

Name*

User ID*

Password*

Confirm Password*

Sign up

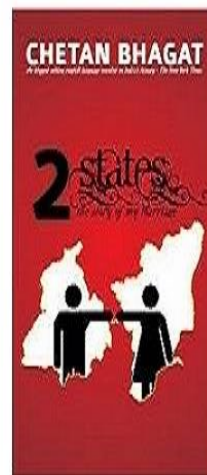
This page makes user 1 enter his/her account credentials after Signing Up with E-Shopper. After you have logged in, then he/she can give ratings to books of his/her choice and then get the recommendations accordingly.

Rating Page:

[Home](#)[Our Recommendation](#)[Sign in / Sign up](#)

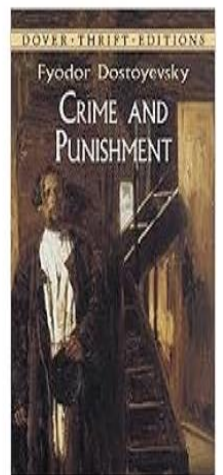
MANDATE : WILL OF THE PEOPLE

Written by: VIR SANGHVI



2 STATES

Written by: CHETAN BHAGAT




CRIME AND PUNISHMENT

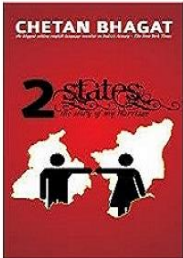
Written by: FYDOR DOSTOYEVESKY

After Login, user 1 is allowed to rate each and every product in the E-Shopper portal in order to get accurate results. These ratings given by the users will help in recommending the products which suits the likeness of the user.

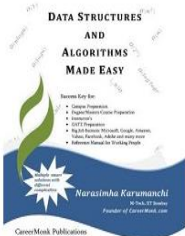
Recommendation Page:



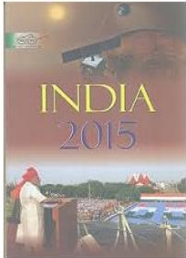
[Home](#) [Our Recommendation](#) Welcome jatin ▾



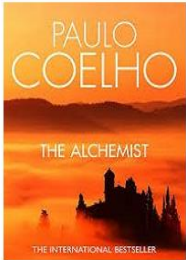
2 STATES
Written by: CHETAN BHAGAT



DATA STRUCTURES AND ALGORITHMS MADE EASY
Written by: NARASIMHA KARUMANCHI



INDIA 2015
Written by: PUBLICATIONS DIVISION



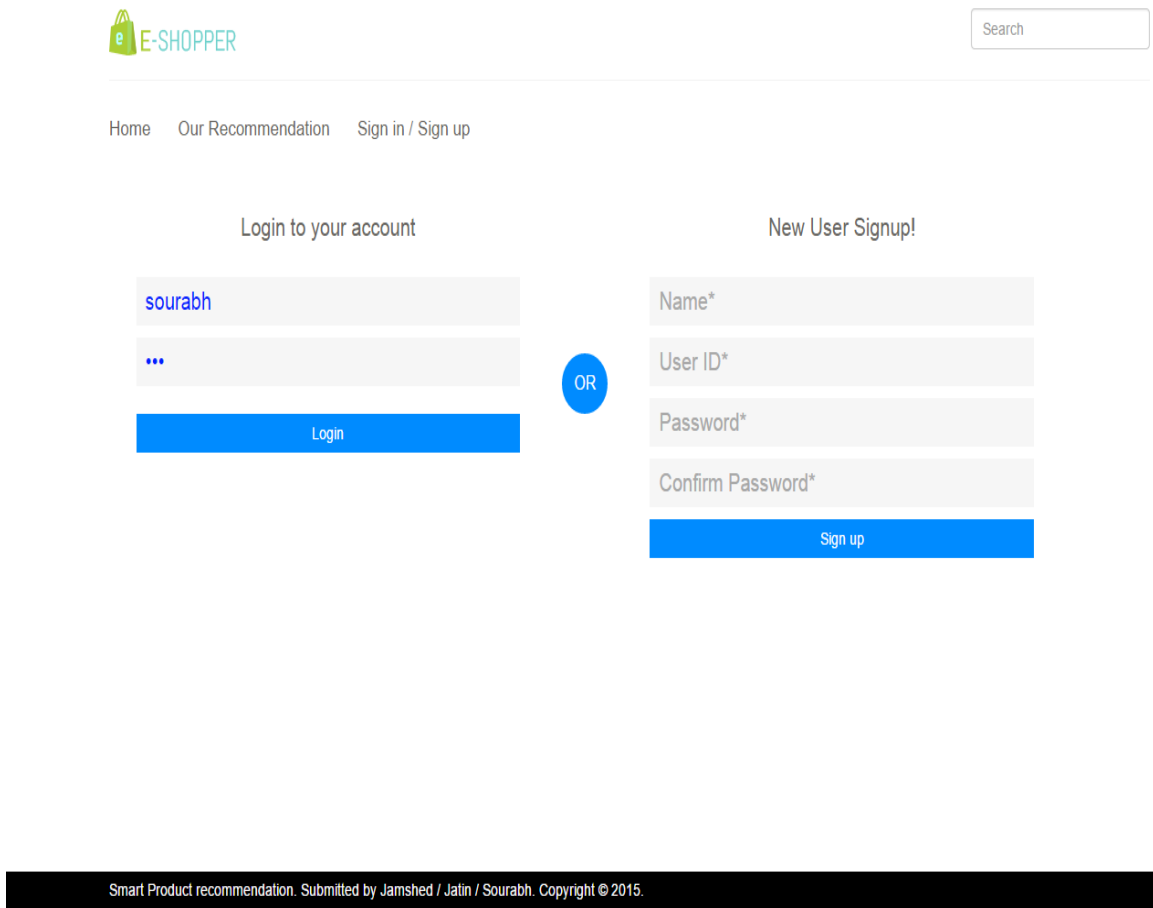
THE ALCHEMIST
Written by: PAULO COELHO

Smart Product recommendation. Submitted by Jamshed / Jatin / Sourabh. Copyright © 2015.

After all the products are being rated in the E-Shopper portal by user 1, then according to the algorithm, it then recommends the following products based on the ratings of all the users registered along with the price.

7.5 User 2:

Login Page



The screenshot displays the login and registration interface for 'E-SHOPPER'. At the top left is the logo, and at the top right is a search bar. Below the header is a navigation menu with links to Home, Our Recommendation, and Sign in / Sign up. The main content area is split into two columns. The left column, titled 'Login to your account', contains input fields for a username (pre-filled with 'sourabh') and a password (masked with dots), followed by a blue 'Login' button. The right column, titled 'New User Signup!', contains input fields for Name*, User ID*, Password*, and Confirm Password*, followed by a blue 'Sign up' button. A blue circle with the text 'OR' is positioned between the two columns. At the bottom of the page is a black footer bar with white text: 'Smart Product recommendation. Submitted by Jamshed / Jatin / Sourabh. Copyright © 2015.'

E-SHOPPER

Search

Home Our Recommendation Sign in / Sign up

Login to your account

sourabh

...

Login

OR

New User Signup!

Name*

User ID*

Password*

Confirm Password*

Sign up

Smart Product recommendation. Submitted by Jamshed / Jatin / Sourabh. Copyright © 2015.

This page makes user 2 enter his/her account credentials after Signing Up with E-Shopper. After you have logged in, then he/she can give ratings to books of his/her choice and then get the recommendations accordingly.

Rating Page:



MANDATE : WILL OF THE PEOPLE

Written by: VIR SANGHVI

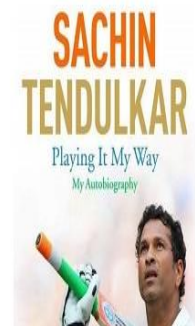
Rating: 3



CRACKING THE CODE: MY JOURNEY

Written by: AYUSHMANN KHURRANA

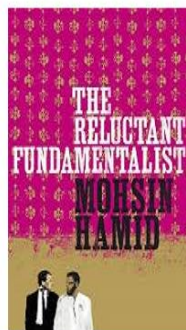
Rating: 10



SACHIN TENDULKAR - Playing It My Way

Written by: SACHIN TENDULKAR

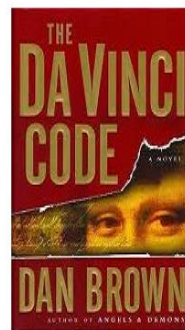
Rating: 9



THE RELUCTANT FUNDAMENTALIST

Written by: MOHSIN HAMID

Rating: 7



DA VINCI CODE

Written by: DAN BROWN

Rating: 2



JAVA THE COMPLETE REFERENCE

Written by: HERB SCHILDT

Rating: 5

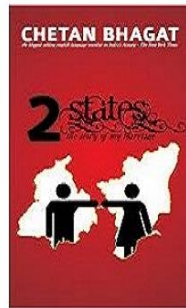
Smart Product recommendation. Submitted by Jamshed / Jatin / Sourabh. Copyright © 2015.

After Login, user 2 is allowed to rate each and every product in the E-Shopper portal in order to get accurate results. These ratings given by the users will help in recommending the products which suits the likeness of the user.

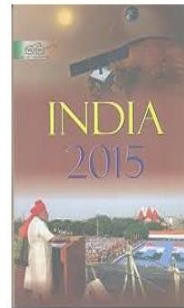
Recommendation Page:

[Home](#)[Our Recommendation](#)

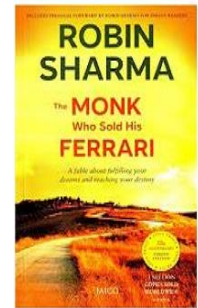
Welcome Sourabh ▾

**2 STATES**

Written by: CHETAN BHAGAT

**INDIA 2015**

Written by: PUBLICATIONS DIVISION

**THE MONK WHO SOLD HIS FERRARI**

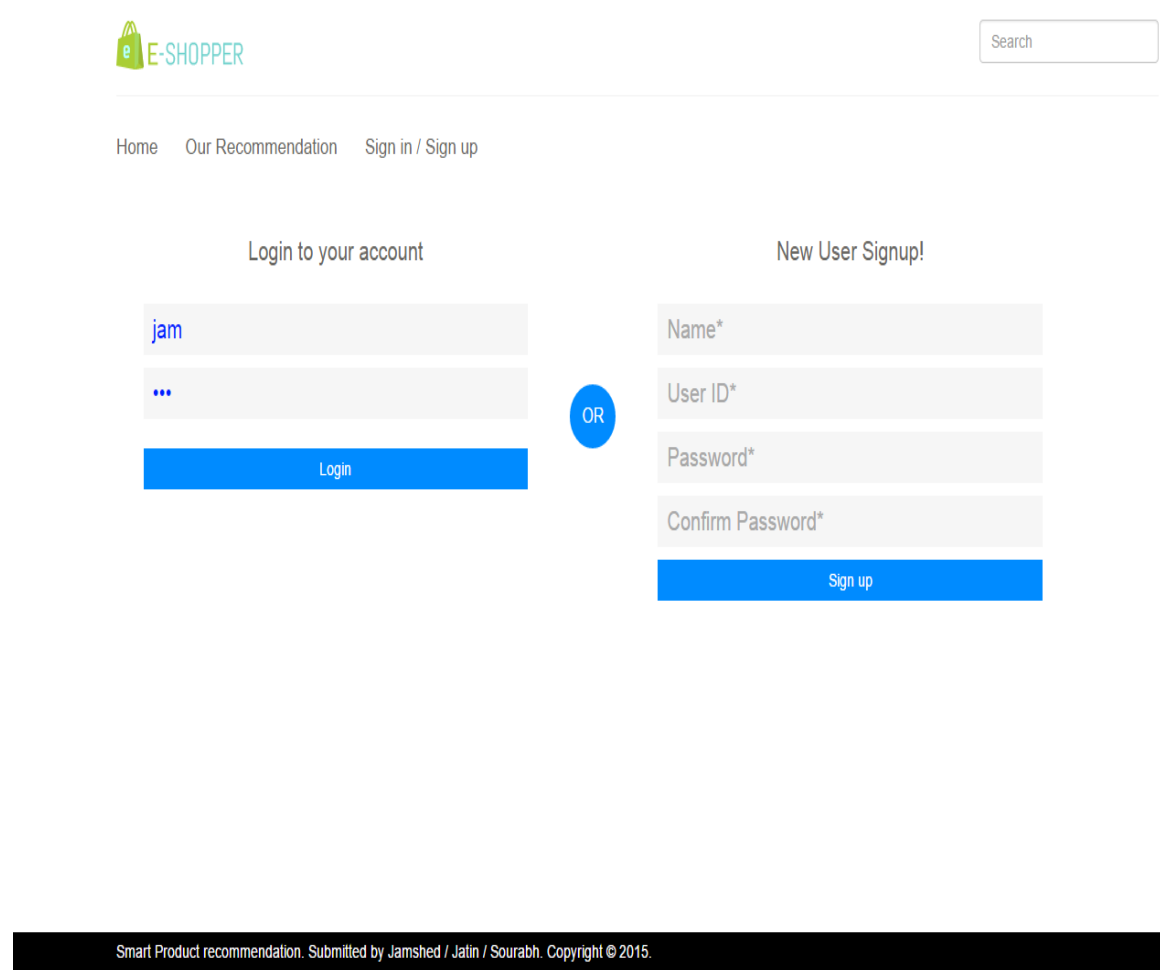
Written by: ROBIN SHARMA

Smart Product recommendation. Submitted by Jamshed / Jatin / Sourabh. Copyright © 2015.

After all the products are being rated in the E-Shopper portal by user 2, then according to the algorithm, it then recommends the following products based on the ratings of all the users registered along with the price.

7.5 User 3:

Login Page



The screenshot displays the 'E-SHOPPER' login and registration interface. At the top left is the logo, and at the top right is a search bar. Below the header, navigation links for 'Home', 'Our Recommendation', and 'Sign in / Sign up' are visible. The main content area is split into two columns. The left column, titled 'Login to your account', contains two input fields: the first has the text 'jam' and the second has three dots. Below these is a blue 'Login' button. The right column, titled 'New User Signup!', contains four input fields: 'Name*', 'User ID*', 'Password*', and 'Confirm Password*'. Below these is a blue 'Sign up' button. A blue circle with the text 'OR' is positioned between the two columns. At the bottom of the page, a black footer bar contains the text: 'Smart Product recommendation. Submitted by Jamshed / Jatin / Sourabh. Copyright © 2015.'

This page makes user 3 enter his/her account credentials after Signing Up with E-Shopper. After you have logged in, then he/she can give ratings to books of his/her choice and then get the recommendations accordingly.

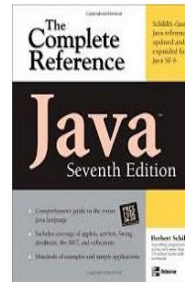
Rating Page:



OBJECT ORIENTED PROGRAMMING WITH C++

Written by: E. BALAGURUSAMY

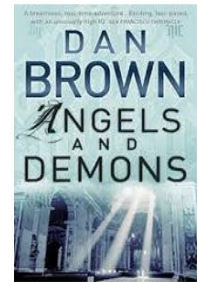
Rating: 7



JAVA THE COMPLETE REFERENCE

Written by: HERB SCHILDT

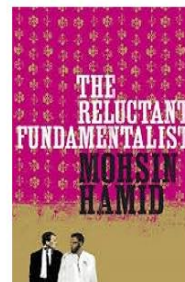
Rating: 5



ANGELS AND DEMONS

Written by: DAN BROWN

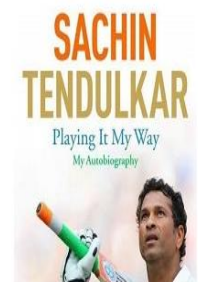
Rating: 3



THE RELUCTANT FUNDAMENTALIST

Written by: MOHSIN HAMID

Rating: 5



SACHIN TENDULKAR - Playing It My Way

Written by: SACHIN TENDULKAR

Rating: 5

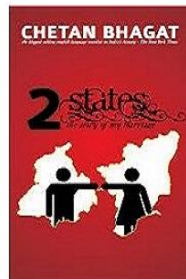
Smart Product recommendation. Submitted by Jamshed / Jatin / Sourabh. Copyright © 2015.

After Login, the user 3 is allowed to rate each and every product in the E-Shopper portal in order to get accurate results. These ratings given by the users will help in recommending the products which suits the likeness of the user.

Recommendation Page:

[Home](#)[Our Recommendation](#)

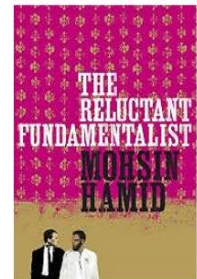
Welcome jam ▾

**2 STATES**

Written by: CHETAN BHAGAT

**DATA STRUCTURES AND
ALGORITHMS MADE EASY**

Written by: NARASIMHA KARUMANCHI


**THE RELUCTANT
FUNDAMENTALIST**

Written by: MOHSIN HAMID

Smart Product recommendation. Submitted by Jamshed / Jatin / Sourabh. Copyright © 2015.


After all the products are being rated in the E-Shopper portal by user 3, then according to the algorithm, it then recommends the following products based on the ratings of all the users registered along with the price.

7.6 Search:



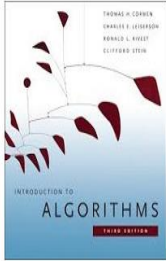
[Home](#) [Our Recommendation](#) Welcome jam ▾

Search results for: alg [2 result]



DATA STRUCTURES AND ALGORITHMS MADE EASY

Written by: NARASIMHA KARUMANCHI



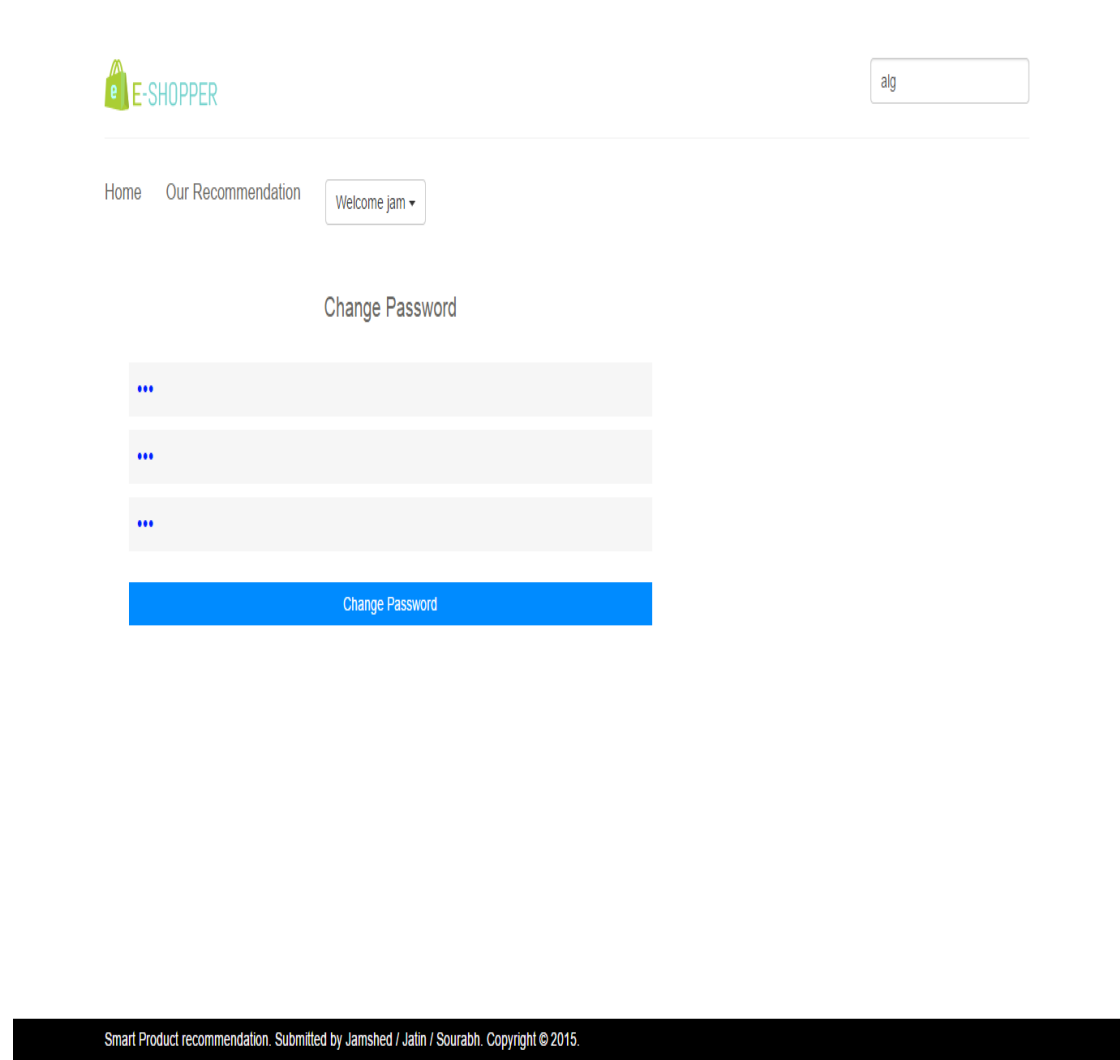
INTRODUCTION TO ALGORITHMS

Written by: AL. CORMEN

Smart Product recommendation. Submitted by Jamshed / Jatin / Sourabh. Copyright © 2015.

The E-Shopper Portal also has a feature of searching books instantly by typing just a few letters and then the book is in front of you.

7.7 Change Password:



The screenshot displays the 'E-SHOPPER' website interface. At the top left is the logo, and at the top right is a search bar containing the text 'alg'. Below the header, there is a navigation bar with links for 'Home' and 'Our Recommendation', followed by a user profile dropdown menu showing 'Welcome jam'. The main heading of the page is 'Change Password'. Below this heading are three password input fields, each represented by a light gray rectangle with three blue dots on the left. At the bottom of the form is a solid blue button labeled 'Change Password'. A black footer bar at the very bottom contains the text: 'Smart Product recommendation. Submitted by Jamshed / Jatin / Sourabh. Copyright © 2015.'

It also helps to change the password for a user if he/she is willing to do so. The Change Password feature ask you your current password and then the new password twice in order to complete process.

CHAPTER -8

CONCLUSION AND FUTURE WORKS

8.1 Conclusion

Recommendation algorithms provide an effective form of targeted marketing by creating a personalized shopping experience for each customer. For large retailers like Amazon.com, a good recommendation algorithm is scalable over very large customer bases and product catalogs, requires only sub-second processing time to generate online recommendations, is able to react immediately to changes in a user's data, and makes compelling recommendations for all users regardless of the number of purchases and ratings.

In the future, we expect the retail industry to more broadly apply recommendation algorithms for targeted marketing, both online and offline. While e-commerce businesses have the easiest vehicles for personalization, the technology's increased conversion rates as compared with traditional broad-scale approaches will also make it compelling to offline retailers for use in postal mailings, coupons, and other forms of customer communication.

8.2 Future Works

Although the techniques which are implied in our product recommendation system gives a perfect mix of products according to the likeness of the user. However, its performance can surely be improved by adding more factors to compare like products added to cart, number of times product sold, products added to wish list and so on. This would definitely help in narrowing down the products recommended and making it efficient.

REFERENCES

- [1] J. Ben Schafer, Joseph Konstan, John Riedl, **“Recommender Systems in E-Commerce”** in Proceeding of the 1st ACM Conference on Electronic Commerce ,1999.
- [2] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl, “Item-based collaborative filtering recommendation algorithms” in Proceedings of the 10th international conference on World Wide Web,2001.
- [3] Michael D. Ekstrand, John T. Riedl and Joseph A. Konstan **“Collaborative Filtering Recommender Systems”** in Foundations and Trends in Human–Computer Interaction Vol. 4, No. 2, 2011.
- [4] Greg Linden, Brent Smith, and Jeremy York, **“Item-to-Item Collaborative Filtering”** in an Industrial report Magazine of Amazon.com, Jan/Feb 2003.
- [5] Sonal Patil and Ankush Mahajan, **“Design and Architecture for Web Graph Mining Base Recommender System for Query, Image and Social Network using Query Suggestion Algorithm and Heat Diffusion Method”** in International Journal of Advanced Research in Computer Science and Software Engineering, 2014.
- [6] Alexandre de Spindler, Moira C. Norrie, Michael Grossniklaus and BeatSigner, **“Spatio-Temporal Proximity as a Basis for Collaborative Filtering in Mobile Environments”** in Institute for Information Systems, ETH Zurich 8092 Zurich, Switzerland, 2005.s