San Jose State University

Project Report

URead - A News Recommender System

Authors:
Kunal Goswami 010772541
Parag Swami 010698675
Kajal Chauhan 010740496

Under:
Prof. Chandrasekhar Vuppalapati

Contents

1 What is URead? 4
- 1.1 Introduction . . . 4
- 1.2 Intuition . . . 4
- 1.3 Motivation . . . 4
- 1.4 Applications . . . 4

2 Requirements 5
- 2.1 Functional Requirements . . . 5
  - 2.1.1 Client Side . . . 5
  - 2.1.2 Server Side . . . 5
- 2.2 Non-Functional Requirements . . . 5

3 UI Design Principles 6

4 High Level Architecture Design 7
- 4.1 Architecture Overview . . . 7
  - 4.1.1 User Interface Module . . . 7
  - 4.1.2 Recommendation Engine . . . 7
- 4.2 Infographic . . . 8

5 Dataset 9
- 5.1 Explicit Data Collection . . . 9
- 5.2 Implicit Data Collection . . . 9

6 Dataflow Diagrams 10

7 Data Mining Principles 13
- 7.1 Vector Space Model . . . 13
- 7.2 Similarity Measures . . . 14
  - 7.2.1 Euclidean Distance . . . 14
  - 7.2.2 Pearson Co-relation Co-efficient . . . 15
  - 7.2.3 Cosine Similarity . . . 15
- 7.3 Neighbourhood . . . 15
- 7.4 K Nearest Neighbours . . . 15
- 7.5 User Based Recommendation Engine . . . 16

8 Knowledge Discovery in Databases 17
- 8.1 Data Cleansing and Selection . . . 17
- 8.2 Data Preprocessing . . . 17
- 8.3 Data Transformation . . . 17
- 8.4 Data Mining and Machine learning . . . 17
- 8.5 Knowledge Discovery . . . 18

9 Apache Mahout 19

10 Client side Design 20

11 Testing 22

Abstract

URead is an article recommender system, based on user based recommendation approach. This document first deals with the high level architecture of the system followed by the data flow in our project. Further more the document then deals with the technical details behind the implementation of our recommender system, the algorithms that we have used to recommend new articles to the users as well as the methods used in it. We have also covered the tools that we have used for the purpose of our recommender system, the detailed architecture has also been provided. Lastly the document deals with the testing results for the user interface for our recommendation system.

Acknowledgements

We sincerely thank Prof. Chandrasekhar Vuppalapati for the continuous support and guidance during the tenure of this project, he has provided valuable insights as well as guided us throughout this project for the successful implementation of the same. We would also like to thank the Computer Engineering Department of the San Jose State University for the resources it provided us to work on this project.

1 What is URead?

1.1 Introduction

URead is what one can popularly refer to as a recommendation system. However, it is much more than that in technical terms, it is a system which involves the high level abstraction of machine learning and data mining components which upon bring invoked deal with the multitude of text and user data to generate effective and efficient suggestions to the user based on the past history.

1.2 Intuition

The intuition behind the recommender system is very simple, if a user buys a set of n items regularly from a store. Consider another user which buys m items from the same store and the case is where m is less than n. In such a case, it is very likely that the user would like the items the prior user is purchasing from the store, it could be possible that they're of the liking to this user as well.

1.3 Motivation

The main reason behind selecting this project is to obtain a better understanding of recommender systems, the applications of which are scaling dynamically with the increase in the data available through the internet. As well as the increase in the competition in the market, also to be familiar with the popular recommender system frameworks such as Apache Mahout.

1.4 Applications

With the growing amount of data that is digitised, newspapers are also digitised. Along with that, users who read such articles on the web want more and more personalised articles for them so as to lessen the amount which they spend surfing for the articles which they specifically like. Such a system would not only boost the user's experience of reading articles but also would build the customer business relationship to a personal level. The same recommender system when tweaked a bit can be implemented as a book recommender system as well as a movie recommender system. As long as there's data available on different abstractions of items, recommender system applications would continue to grow.

2 Requirements

2.1 Functional Requirements

2.1.1 Client Side

1. Authentication: The client application shall provide an authentication interface that makes it possible for the user to get access to the recommended articles.
2. Request Recommendations: The client application shall make it possible to request recommendations of the news articles and to send the requests to the server application.
3. Evaluate list: The client application shall make it possible to handle the incoming data and evaluate it in order to display it on the user interface.

2.1.2 Server Side

The server application receives information from the client application, and provides the client application with recommendations. The requirements for the server application are:

1. The server application shall receive and handle requests for recommendations.
2. The server application shall evaluate the stored database in order to extract the required data for it.
3. The server application shall be capable of producing recommendations by interpreting the content and evaluations provided by the database.
4. The server application shall send the extracted data by the recommendation engine back to the client application.

2.2 Non-Functional Requirements

1. Accuracy: The server application shall produce accurate recommendations that match the users preference.
2. Intrusiveness: The client application shall minimize intrusiveness and at the same time capture user attention so that an acceptable amount of evaluation data is received.
3. Scale potential: The recommender system shall have the potential of being scalable both with respect to size and geography.

3 UI Design Principles

The overall process for designing a user interface begins with the creation of different models of system function. The human and computer-oriented tasks that are required to achieve system function are then outlined; design issues that apply to all interface designs are considered; tools are used to prototype and ultimately implement the design model; and the result is evaluated for quality. All the above can be achieved by following the design principles:

1. Effectiveness and Visibility each users action should produce an effect. Effective UI increases user satisfaction and productivity, so we take into consideration both frontend and backend usability. This project adds maximum visibility to objects, so that users can.

2. Performance in addition to careful software architecture design which is important to the application performance, we have tuned the application interface for quick response by implementing JSP, Xml Java technologies and features.

3. Consistency and Predictability the user interface in our project is similar to what real interface uses. Therefore, the user wont have to work more to learn and use it. This allow the user to develop consistent interfaces that allow users to develop usage patterns theyll learn what the different buttons, tabs, icons and other interface elements look like and will recognize them and realize what they do in different contexts.

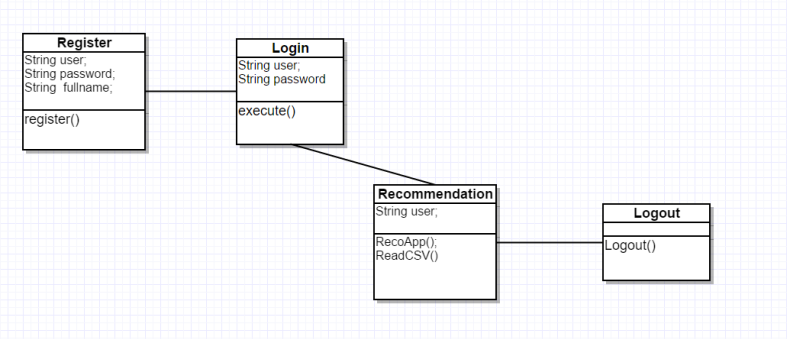

Figure 1: The diagram illustrates the basics of how different components of the User interface interact with each other.

4 High Level Architecture Design

4.1 Architecture Overview

4.1.1 User Interface Module

The user interface module contains the client side of the recommender system which involves the user login and regisration processes discussed later in the document. This section provides the high level abstraction of the architecture of our recommendation system. The details of the modules are covered later in the document.

4.1.2 Recommendation Engine

The recommendation engine module of the system contains the J2EE application which communicates with the MySQL Database and the JSP. This engine contains the implementation of the core functionality of our recommendation system. It deals with preprocessing and data transformation of the user data set as well as the recommendations generated at the end of the system. The diagram gives you a block view of the entire architecture.

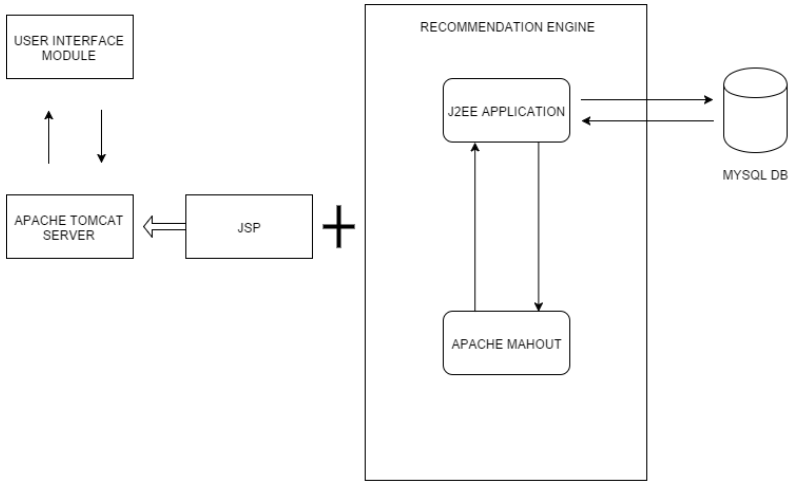

Figure 2: Figure illustrates the block diagram for the news recommender system under consideration.

4.2 Infographic

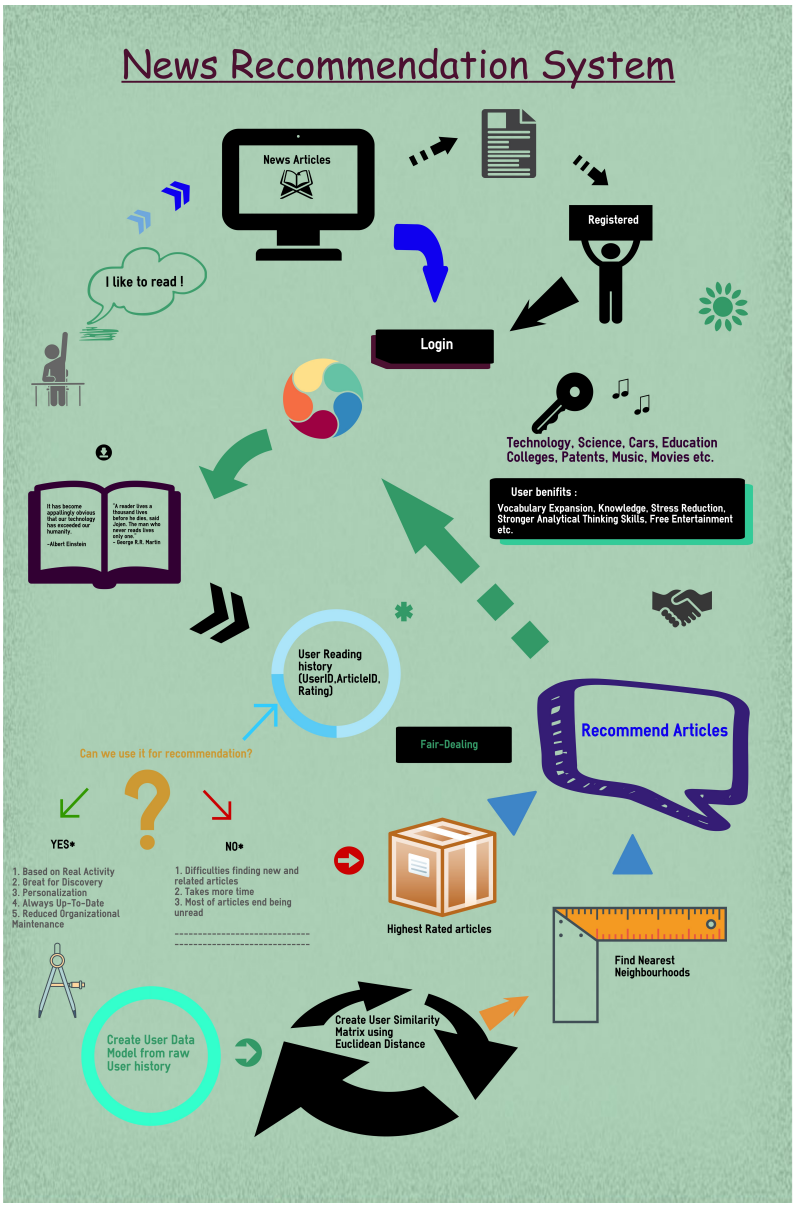

Figure 3: The diagram depicts the infographic concerning our recommender system.

5 Dataset

The data set that we are using is the YOW User Study Data which was originally collected by the PI at Carnegie Mellon University. The goal of this user study was to collect implicit and explicit data from the users participating in the study on a heterogenous set of documents and a wide range of information about documents and topics. Twenty one subjects participated in this study for four weeks, the implicit and explicit data was collected in the following format:

5.1 Explicit Data Collection

After the user finished reading an article the browser would redirect user to a form where the user would suggest the class for the article, rate the article basically by providing a scale of the user's liking for the article. If the article was assigned classes already, the user was asked to determine the relevance of the article with the assigned classes. The user also had the opporunity to assign new classes to the article, the articles were also rated based on measures such as *readable, authorative, novel* and *relevant*.

5.2 Implicit Data Collection

The browser used for this user study recorded the user's mouse and keyboard activites, the number of seconds the user spent on the article as well as the number of times the user scrolled while reading an article. These methods of implicit data collection were only at work when the mouse pointer was focued within the article, whenever it went out of focus the recording stopped.

From the original data set we considered only the attributes *User id, Doc id, relevant and classes* for the purpose of our recommendation. Furthermore we divided this data set into two models, the first part contained the *User id, doc id* and the *rating* given by the user and the second partition of the data contained the *doc id* and *classes*.

The data contained over 7000 article ratings from the given 21 users. We have used all such ratings for the purpose of our recommendation system.

6 Dataflow Diagrams

This section contains the data flow diagrams for our recommender system.

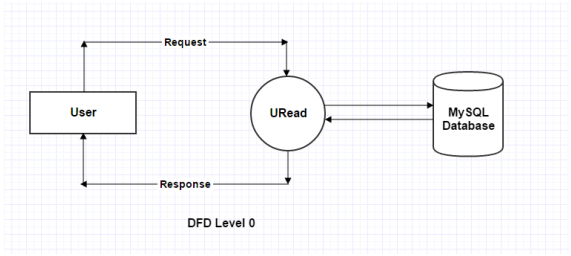

Figure 4: This figure represents the highest abstraction for our recommendation system. The user sends requests to the recommender system and the system responds with the appropriate output to the user.

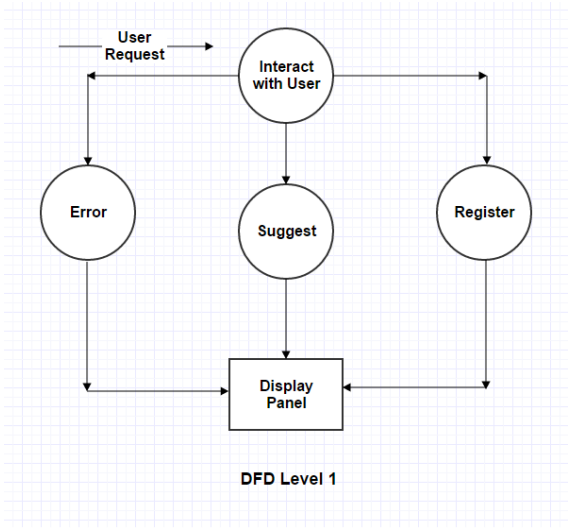

Figure 5: This figure represents the basic processes that revolve around the recommender system. The details of every single such process are incorporated in the further diagrams.

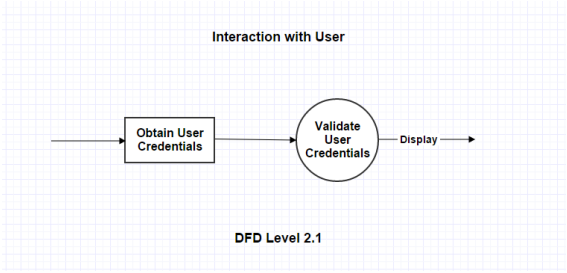

Figure 6: This data flow diagram illustrates the interaction of the system with the user.

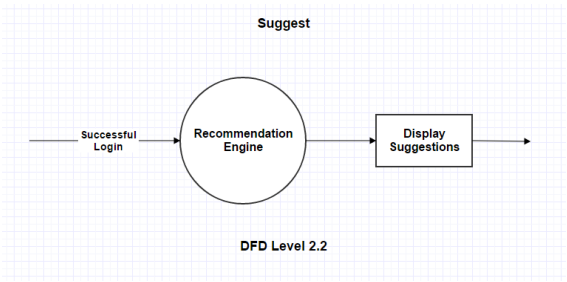

Figure 7: The Data flow diagram illustrates the working of the suggest module of the recommender system which basically deals with recommending articles to the user.

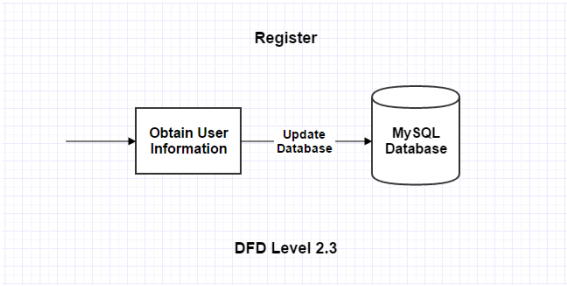

Figure 8: This data flow diagram illustrates the usage of the registration module for the particular user.

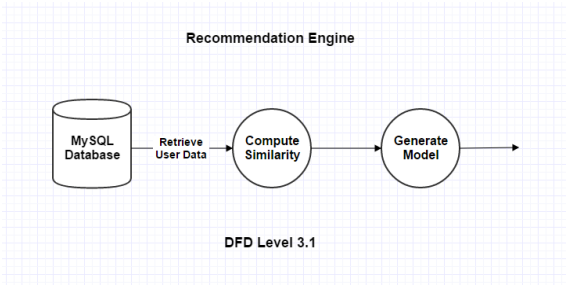

Figure 9: This data flow diagram illustrates the usage of the recommender engine for suggesting new articles to the user.

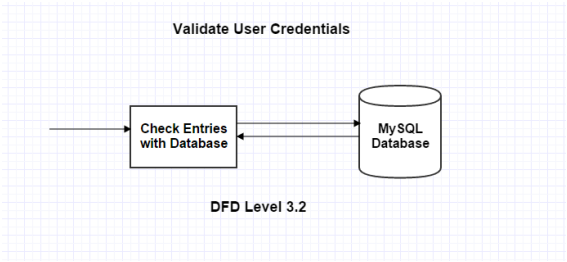

Figure 10: This data flow diagram illustrates the flow of data as the system evaluates the credentials a user enters while logging in.

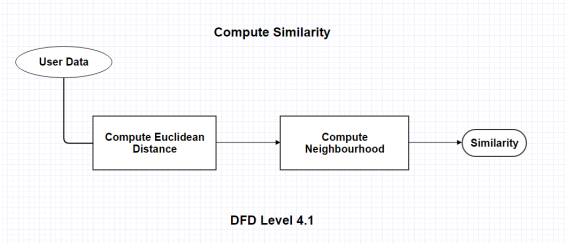

Figure 11: This data flow diagram illustrates the call to the compute similarity module from the recommender engine so as to determine the euclidean distance between different users.

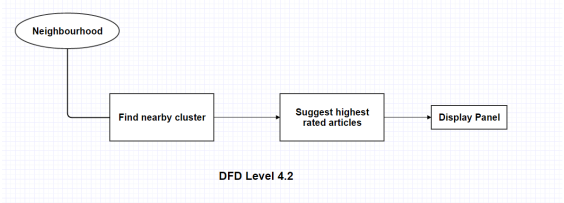

Figure 12: This data flow diagram represents the flow of the data as the system calculates the nearest neighbours for the user.

7 Data Mining Principles

The recommender system that weve developed is an article recommender system, it revolves around many algorithms and data mining principles for the effective implementation of the same. As we mentioned earlier the dataset contains both explicit as well as implicit feedback from the users, the explicit user feedback provides us the basic overview of how the user relates with the article, whether or not the article is to the users liking. A quick look at the ratings on various articles would be enough to determine whether or not the user enjoyed those. However, whenever theres a case of human subject under consideration theres a window of implicit data as well, data that is not something we as humans normally notice but a machine can capture it and with the help of advanced algorithms and data mining principles the behaviour of the user can be grasped from the same. The implicit data includes the time spent by the user on every article, the number of times the user used the scroll bar as well as the mouse and keyboard entires. The user is more likely to engage with an article which he/she enjoys a lot, this would automatically be reflected in some manner through the explicit and implicit features.

From the data that we have, that is the dataset, we first create unique user profiles. Therere more than 7000 entires in the dataset and 21 users contributed for the same, each user has a distinct profile and contains the article ids and topic which he/she liked as well as rated. Based on this information the user profile can be generated which can have different attributes per every article, may it be the topic on which the article was. These user profiles then become the utility matrix upon which we compute the similarity between different users, also we cluster the users based on their ratings on common articles with the concept of neighbourhood.

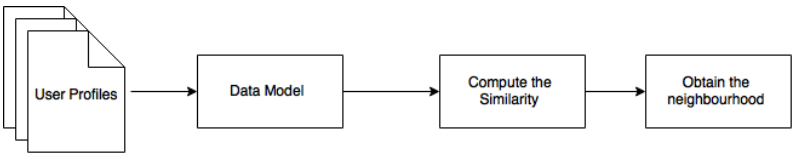

Figure 13: The above figure presents the steps through which the user neighbourhood is generated.

The basic principle used here is the similarity measure, the user profile here is showcased as a vector in an n dimensional space, n being the number of attributes in the user profile. For e.g. *(User id, article id, rating, time spent, number of scrolls)* has 4 attributes for the user. The attribute values are associated with a numeric measure which augments the computation of the vector for the same, the categories of the article can be mapped to unique keys so as to refer them easily.

7.1 Vector Space Model

The presentation of objects as vectors in n dimensional space is in general known as the vector space model. Popularly used for text documents, converting them into vectors for better and more effective computation with regard to data mining, in our recommender system we employ this model to generate the vectors of the user profiles that we have. Every single dimension of the vector represents an attribute that the user posseses, to be more precise every dimension of the vector would represent a feature for the corresponding user and the corresponding article. Consider the following example for a better understanding of the same.

$$U_i = (12, 4.5, 120, 5)$$
$$\text{where i} = \text{The ith user in the dataset.}$$

Here the value 12 in the vector maps to the article id, the entry followed by it, that is 4.5 indicates the user's rating on the same article and so on. What we considered here is a very small subset of the actual data but which provides enough information to understand the basic working of the Vector Space Model.

Every such vector is stored in a matrix of $m \times n$ where m indicates the total number of users in the system and n indicates the number of features for every single user. Once such a matrix is generated, it is very easy to compute the similarity between different users.

7.2 Similarity Measures

Similarity basically means the distance between two given vectors, which specifies the nearness between them in the n-dimensional space. In our case, this concept directly relates to the fact that the less distance between the vectors of two users, the more similar to each other they are and hence are more likely to enjoy similar articles or more likely to enjoy the article which the other user found interesting.

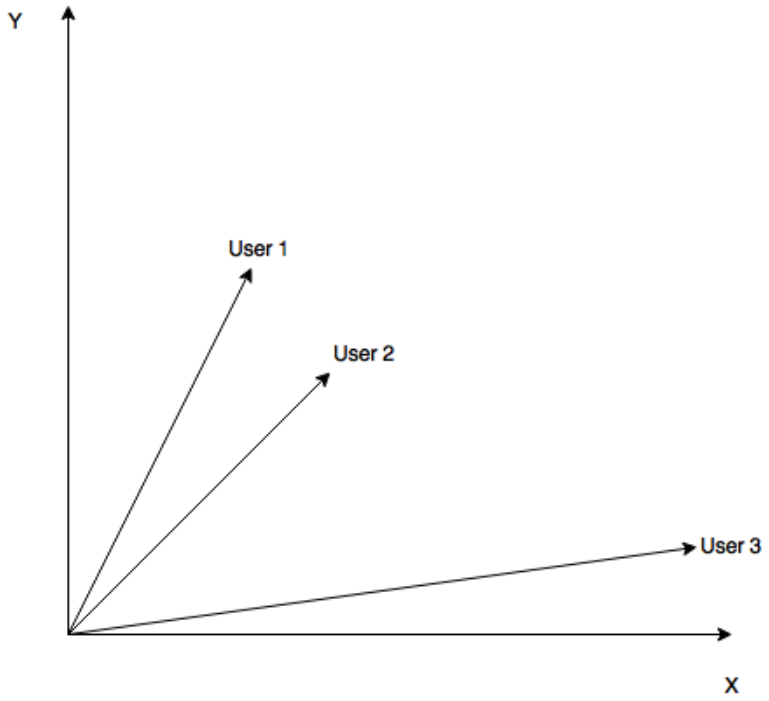

Figure 14: Consider the 3 users, user 1, user 2 and user 3 as shown in the figure in a 2-dimensional space. It is clear from the diagram itself that users 1 and 2 are nearer to each other than users 1 and 3. This nearness can be obtained using various similarity measures, which we'll discuss in detail shortly. The point to grasp from the figure is, if there's an article which user 1 finds interesting, it is much more likely that user 2 would find the same article interesting rather than user 3. That's the intuition our recommender system revolves around.

Let us now explore the different methods to obtain the similarties between different users. Also known as distance corelation methods.

7.2.1 Euclidean Distance

Given two vectors p and q, the Euclidean distance between the two vectors is defined as follows:

$$\mathrm{p} = (p_1, p_2, p_3, p_4, ..., p_n)$$
$$\mathrm{q} = (q_1, q_2, q_3, q_4, ..., q_n)$$

The Euclidean distance for the same is given as,

$$d(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + ... + (p_n - qn)^2}$$

The interesting property about Euclidean distance is the fact that even if the vectors are scaled up to more dimensions or scaled down to fewer dimensions, the Euclidean distance remains constant.

7.2.2 Pearson Co-relation Co-efficient

Pearson's coefficient is the classical measure of corelation between two objects, the value of the same ranges from -1 to + 1 where a +1 indicates completely co-related objects, 0 indicates no co-relation between the objects what so ever and -1 indicates a complete negative co-relation between the objects. The pearson corelation coefficient for two vectors p and q is given as follows:

$$\rho = \frac{cov(p,q)}{\sigma_p \sigma_q}$$

where cov = The covariance between p and q.

and σ = The standard deviation within the vector under consideration.

7.2.3 Cosine Similarity

Cosine similarity as the name suggests is the measure of the cosine angle between two vectors in the vector space model. It is very much similar to the pearson coefficient in terms of the output. The cosine similarity returns a +1 if the vectors are equal to each other and a -1 if the two vectors are complete opposite of each other. The values are seldom equal to the extremes but the nearer it is to either, the easier it is for us to quantify the user profiles. The cosine similarity measure is obtained with the help of the inner dot product between the two vectors. The cosine similarity between two vectors p and q is given as,

$$cos(\theta) = \frac{p \cdot q}{||p||||q||}$$

where θ is the angle between them.

7.3 Neighbourhood

The neighbourhood as the name suggests determines the nearest p users given the user similarity based on the measures we discussed above. Once the neighbourhood is determined, the data is then passed on to the user based recommendation engine. We've used the Apache mahout implementation for determining the neighbourhood as well as generating recommendations from the neighbourhood data, as we mentioned earlier. The neighbourhood concept basically makes use of the K- nearest neighbour algorithm to determine the k nearest users based on the similarity measure we provide it.

7.4 K Nearest Neighbours

As the name of the algorithm suggests, it computes the nearest data points in the given dataset. Given the features, this algorithm intially assigns random data points depending on the data set. Once these random points are assigned, the distance of every single data point is computed with these so called assigned random points to determine the nearest neighbours by calculating the Euclidean distance between the assigned point and the random point, according to the distance displayed by such points the random data points are then updated to obtain the nearest neighbours based on some parameters. The value of k is best determined heuristically, there's no right or wrong choice for k as it is different based on different data.

One can obtain an understanding that the data is then clustered into groups according to the values that are stored in the data points, the values are basically the features of the data. In our case the features of the users' profiles, once the nearest users are determined, every user can be assigned to a class. The class in our case is determined with the help of the articles the user prefers, the preferance is obtained from the ratings the user gives on different articles.

7.5 User Based Recommendation Engine

User Based Recommendation Engine is the classical approach to recommendation engine, more precisely the conventional way to recommendations. To explain it in a simple manner, consider a set of items and a set of users, for this sample dataset we have the transactions of every single user. The basic way to recommend an item to a user is to suggest that, 'User B' used to buy these items and he/she found this item very useful. Also, 'User B' regularly buys the same items as the user under consideration. In such cases the user might relate to the recommendation based on the similarity with 'User B'. The same module is now mathematically implemented in the User based recommendation engine, with the help of the neighbourhood computed on the data set of all the users then is used to determine the articles the new user or the user under consideration might find interesting. We have limited the number of recommendations to 20, that is the user can be suggested 20 new articles based on the user's history of reading and rating articles. There can be less than 20 recommendations but no more than 20. In the further section we discuss the architecture of the data tool that we used, that is Apache Mahout and discuss it's details.

8 Knowledge Discovery in Databases

The goal of knowledge discovery in databases is to extract useful information from the data. As the name suggests it's a process which identifies the useful information from the given data. The KDD process consists of the following steps:

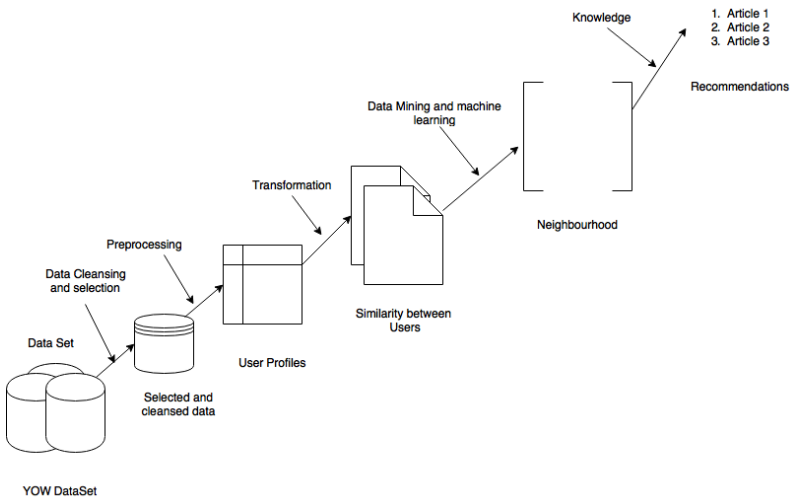

Figure 15: The above figure illustrates the steps of a KDD process for the purpose of our recommender system, that is the article recommender system. The steps in this process are discussed within the text.

8.1 Data Cleansing and Selection

As a part of this step in the KDD process, we have removed the entries with null values under consideration of the ratings. The data set also contained a lot many attributes as a part of the user study from all those attributes we have selected the *User id, doc id, rating, classes* and *relevance* attributes for every entry in the data. The output of this step is a clean dataset, containing just the attributes that we want.

8.2 Data Preprocessing

Under this step of the KDD process, before performing computations on the data, it is preprocessed to the desirable format. For our case the preprocessing of the data means that the data is stored in a data model according to the user id and document id which the user has rated, for the sake of understanding this can be considered as an m $\times$ n matrix, where m is the total number of entires and n is the total number of attributes per entry.

8.3 Data Transformation

In this step of the KDD process, we have applied the Euclidean Distance measure discussed in the previous section of the document, this process computes the similarity of the user with the other user and is accomplished with the help of Apache Mahout's similarity module.

8.4 Data Mining and Machine learning

Once the user similarity has been obtained, the data is then passed on to a K Nearest Neighbours algorithm to compute the nearest neighbours in between the total users. That is the inference we

obtain from the dataset, the nearest users based on the similarity measure.

8.5 Knowledge Discovery

The neighbourhood data is then passed as the input to the recommender module of Apache Mahout, also the user id is provided as a part of the API call and the recommendation engine based on the co-relation between the users suggests new articles for the user to read.

In this way, the knowledge discovery in databases process relates with our recommendation system.

9 Apache Mahout

Apache Mahout as the name suggests is a Apache Software Foundation which contains the open source implementation of scalable machine learning algorithms, algorithms that can work effectively even with large amounts of data. We have used Mahout as the basis of our recommendation engine, the basic recommender framework in Mahout is depicted by the following figure. Apache

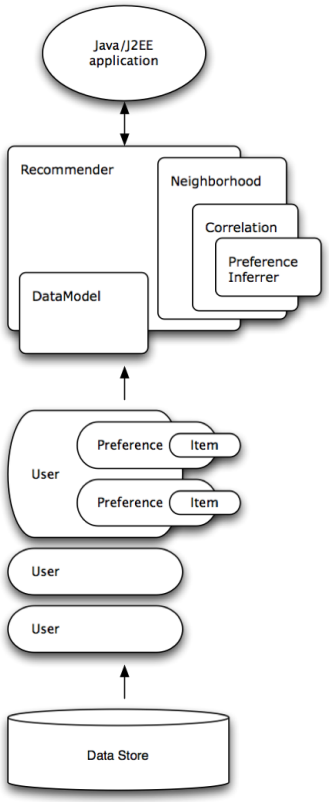

Figure 16: This figure displays the overall outline of the User based recommendation system implementation in Apache Mahout. Image Courtesy: http://mahout.apache.org

Mahout makes it really easy to implement a recommender system, it contains the libraries which can be imported to the J2EE application and with the instances initiated from such classes, computation can be carried out on the data under consideration, which is the YOW data set in our case. Apache Mahout has java as well as hadoop based implementations for the algorithms it contains for machine learning, the hadoop based approach focuses on the distributed implementation of the algorithms. However, the java classes contain the implementations for various algebraic functions and other statistical utilities.

The recommendation engine is a part of the Taste CF component of Apache Mahout, which is written by Sean Owen and then was donated to Mahout. CF stands for collaborative filtering here, we used the GenericUserBasedRecommender module as a part of our recommendation engine.

10 Client side Design

The recommender system has been hosted on the url http://sgasoft.co.in:8080/LoginEx/, therefore not much is included under the client side design other than some modules which are accessible through the web browser. All the computations are done on the server side and only the output is displayed on the client side. The following figures display the User interface of our recommender system:

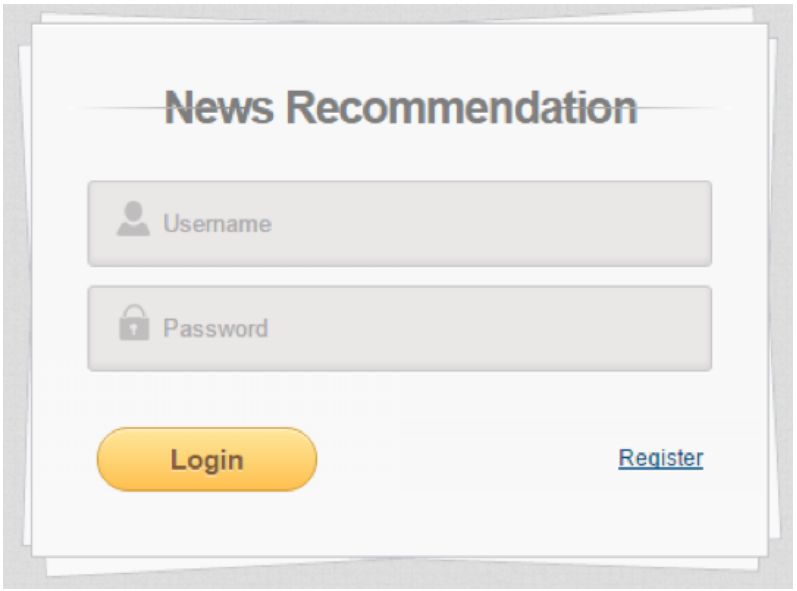

Figure 17: This figure diplays the basic login page for a user in the recommender system. This is basically the first screen that a user will encounter upon clicking the link of the system.

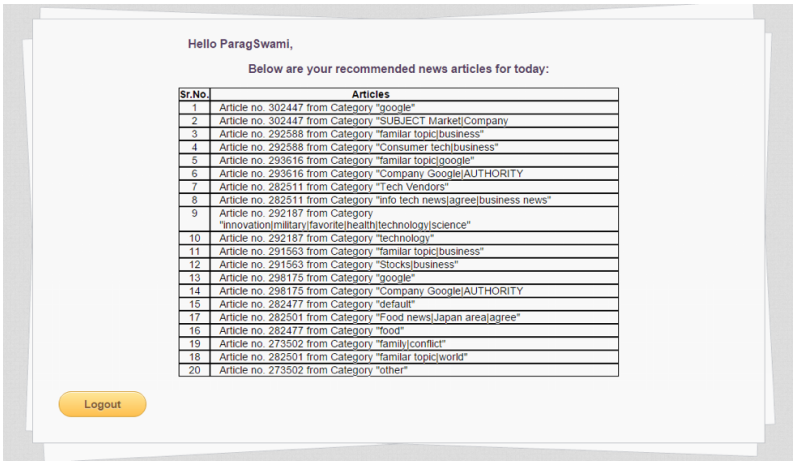

Figure 18: This figure portrays the basic recommender system at work, for a given user this system is recommending 20 articles to read based on the user's past history of rating articles. The system would recommend the top rated 20 articles in the case of a new user.

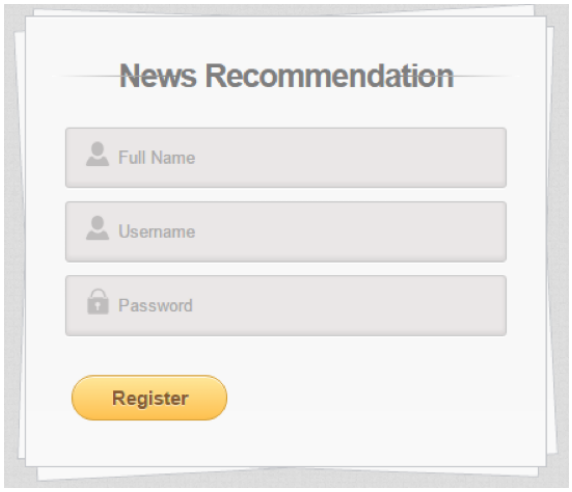

Figure 19: For a new user, the client side would appear like this in the browser, the user would be asked to provide the credentials which would be stored in the MySQL database for future preferances.

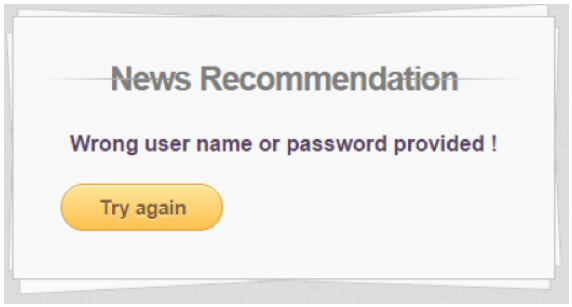

Figure 20: In the case of incorrect login credentials, this is the screen that the user would encounter. The login credentials aren't disclosed for security purposes and the user will be redirected to the login page through this page. From there the user can either register for a new account or can enter the correct login credentials if already registered.

11 Testing

Test case ID	Test case	Input data	Expected results	actual results	Result
1	User should be able to login with details present in the database as well as datasets.	username - 84 & password - test	User should get recommendations as per user history	User logged in successfully and got the recommendations as per user history	Pass
2	User should not be able to login with incorrect password and correct username	username - 84 & password - zest	User should get an error message	User redirects to error page and doesn't able to log in.	Pass
3	User should not be able to login with correct password and incorrect username	username - 8 & password - test	User should get an error message	User redirects to error page and doesn't able to log in.	Pass
4	User should not be able to login with incorrect password and incorrect username	username - 8 & password - pest	User should get an error message	User redirects to error page and doesn't able to log in.	Pass
5	User should be able to register with userID which doesn't exists in the database.	Fullname - Prof Chandra, username - 123 & password - web	User should be able to register.	User registered correctly and can login with new credentials.	Pass
6	User should not be able to register with the userID which already exists in the database	Fullname - Rohan, username - 123 & password - red	User should not be able to register.	User not registered with error message saying userID already exists	Pass
7	User who's history is present in the dataset should get recommendations	username - 84 & password - test	User should get recommendations as per user history	User got the recommendations as per user history	Pass
8	User who's history is not present in the dataset should also get recommendations	Fullname - Prof Chandra, username - 123 & password - web	User should get recommendations	User got the recommendations the articles which were highest rated by other users.	Pass

Figure 21: This figure includes the testing details for the user interface for our recommender system.