DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY PATNA

A Project Report

on

CONTENT BASED USER-USER AND USER-ITEM SIMILARITY AND CONTENT BASED RECOMMENDER SYSTEM

Submitted in Partial Fulfillment of
The Requirements for the Award of the

Degree

of

Bachelor of Technology

In

Information Technology

By

Ankit Kumar Agrawal
Md Musaib Naseem
Manish Kumar

Under the Supervision of:

ASST. PROF. A.S.TEWARI

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY PATNA

CERTIFICATE

This undersigned certify that **ANKIT KUMAR AGARWAL** Roll No. **1307018, MD MUSAIB NASEEM** Roll No. **1307075, MANISH KUMAR** Roll No. **1307025** are registered students for the Bachelor's Program in **Department of Computer Science and Engineering** with specialization in **Information Technology** under my supervision. They all have successfully completed Major Project under my supervision in which they developed a **Recommender System** based on Content-Based User-Item Similarity.

PROF. A.S.TEWARI
(Assistant Professor)
Computer Science and Engineering
National Institute Of Technology
Patna

Dr. PRABHAT KUMAR
(Head Of Department)
Computer Science and Engineering
National Institute Of Technology
Patna

DECLARATION

We hereby declare that this project work entitled **“Content based user-user and user-item Similarity and Content based Recommendation System”** has been carried out by us in the department of **Computer Science & Engineering** of **National Institute of Technology Patna** as a requirement of major project (8th Semester) for the award of degree B.Tech, under the supervision of **PROF.A.S.TEWARI.**

Name and Roll Number	**Signature**
ANKIT KUMAR AGARWAL(1307018)	_____________________
MD MUSAIB NASEEM(1307075)	_____________________
MANISH KUMAR(1307025)	_____________________

Place: Patna

Date: __/__/____

ACKNOWLEDGEMENT

We express our deepest gratitude towards our project guide Asst. Prof. **A.S.TEWARI** (Department of Computer Science & Engineering, NIT PATNA) for his valuable suggestion, insightful criticism and direction throughout the development of project. He has always encouraged us to get out of the shell and do something innovative despite our limitations.

We are also thankful to all the non-teaching staff & other faculties of the department of Computer Science and Engineering, NIT PATNA for their precious support.

Finally, we thank our friend and family members for providing constant encouragement, support and valuable suggestion during the development of the project.

1. ANKIT KUMAR AGRAWAL (1307018)

2. MD MUSAIB NASEEM(1307075)

3. MANISH KUMAR (1307025)

TABLE OF CONTENTS

OBJECTIVE

--> To find Content Based Similarity between distinct user Profiles based on Past History.

--> To find Content Based Similarity between ‘User Profile’ and ‘Item Profile’ based on Past History.

--> To Calculate the Average Rating of each Movie based on User’s Ratings.

--> To Provide Recommendations to Users Based on both ‘User-Item Profile Similarity’ and ‘Average Rating’.

DEVELOPMENT TOOLS & TECHNOLOGY

1. JDK (Java Development Kit)
2. XAMPP Server
3. Eclipse IDE
4. NetBeans IDE
5. JSP (Java Server Page)
6. MySQL
7. Browser (Chrome/Firefox)

CONTENT BASED RECOMMENDATION

System implementing a content-based recommendation approach analyze a group of users and/or descriptions of items previously rated by a user, and build a model or profile of user interests based on the features of the objects rated by that user. The profile is a structured representation of user interests, adopted to recommend new interesting items. The recommendation process basically consists in matching up the attributes of the user profile against the attributes of a content object.

The result is a relevance judgment that represents the user’s level of interest in that object. If a profile accurately reflects user preferences, it is of tremendous advantage for the effectiveness of an information access process. For instance, In could be used to filter search results by deciding whether a user is interested in a specific Web page or not and, in the negative case, preventing it from being displayed.

# OBJECTIVE - 1

CONTENT BASED USER-USER SIMILARITY BASED ON THEIR PAST HISTORY:

Content-based user-user similarity, filters items based on a comparison between the content of an user profile with other user's profile. The content of each user is represented as a set of descriptors or terms, typically the words that occur in the profile. The user profile is represented with the same terms and built up by analysing the content of items which have been seen by the user. Several issues have to be considered when implementing a content-based filtering system. First, terms can either be assigned automatically or manually. When terms are assigned automatically a method has to be chosen that can extract these terms from items. Second, the terms have to be represented such that both of the user profiles can be compared in a meaningful way. Third, a learning algorithm has to be chosen that is able to learn the user profile based on seen items and can filters based on this user profile. The information source conveys that content-based filtering systems are mostly used with the text documents. A standard approach for term parsing selects single words from documents.The vector space model use these terms to represent documents as vectors in a multi-dimensional space.

CHOOSING A LEARNING METHOD

The efficiency of a learning method does play an important role in the decision of which method to choose. The most important aspect of efficiency is the computational complexity of the algorithm, although storage requirements can also become an issue as many user profiles have to be maintained. Neural networks and genetic algorithms are usually much slower compared to other learning methods as several iterations are needed to determine whether or not a profile is relevant. Instance based methods slow down as more training examples become available because every example has to be compared to all the unseen profiles. Among the best performers in terms of speed is Cosine Similarity Methed(Vector Space Model).

EXPLORATION STRATEGIES

The learning methods applied to content-based filtering try to find the most relevant documents based on the user's behaviour in the past. Such approach however restricts the user to documents similar to those already seen. This is known as the over-specialization problem. As stated before the interests of a user are rarely static but change over time. Instead of adapting to the user's interests after the system has received feedback one could try to predict a user's interests in the future and recommend documents that contain information that is entirely new to the user. A recommender system has to decide between two types of information delivery when providing the user with recommendations:

Exploitation. The system chooses documents similar to those for which the user has already expressed a preference.

Exploration. The system chooses documents where the user profile does not provide evidence to predict the user's reaction.

TERM USED

1. Items **-->** Movies rated by an user.

2. Genre of a Movie **-->** provides content data.

3.Target User **-->** An Active User who is currently logged in to the Website.

4. User Profile **-->** Consists a list of Genres of all the movies rated by the user.

5. Profile Vector **-->** It is created from the user profile.

6. Cosine Similarity Method **-->** Compute cosine/closeness between the profile vectors.

7. User-User Similarity **-->** Two user profiles are said to be more closer/similar if the cosine value between their profile vectors is very less.

PROCEDURE

1. We have Prepared User Profile of every user based on the Genre of the Items (i.e. movies) rated by him.

2. We have build the Contents of User Profile(using, **Tree Map data structure**) which includes 'Genre' of all the movies rated by him.

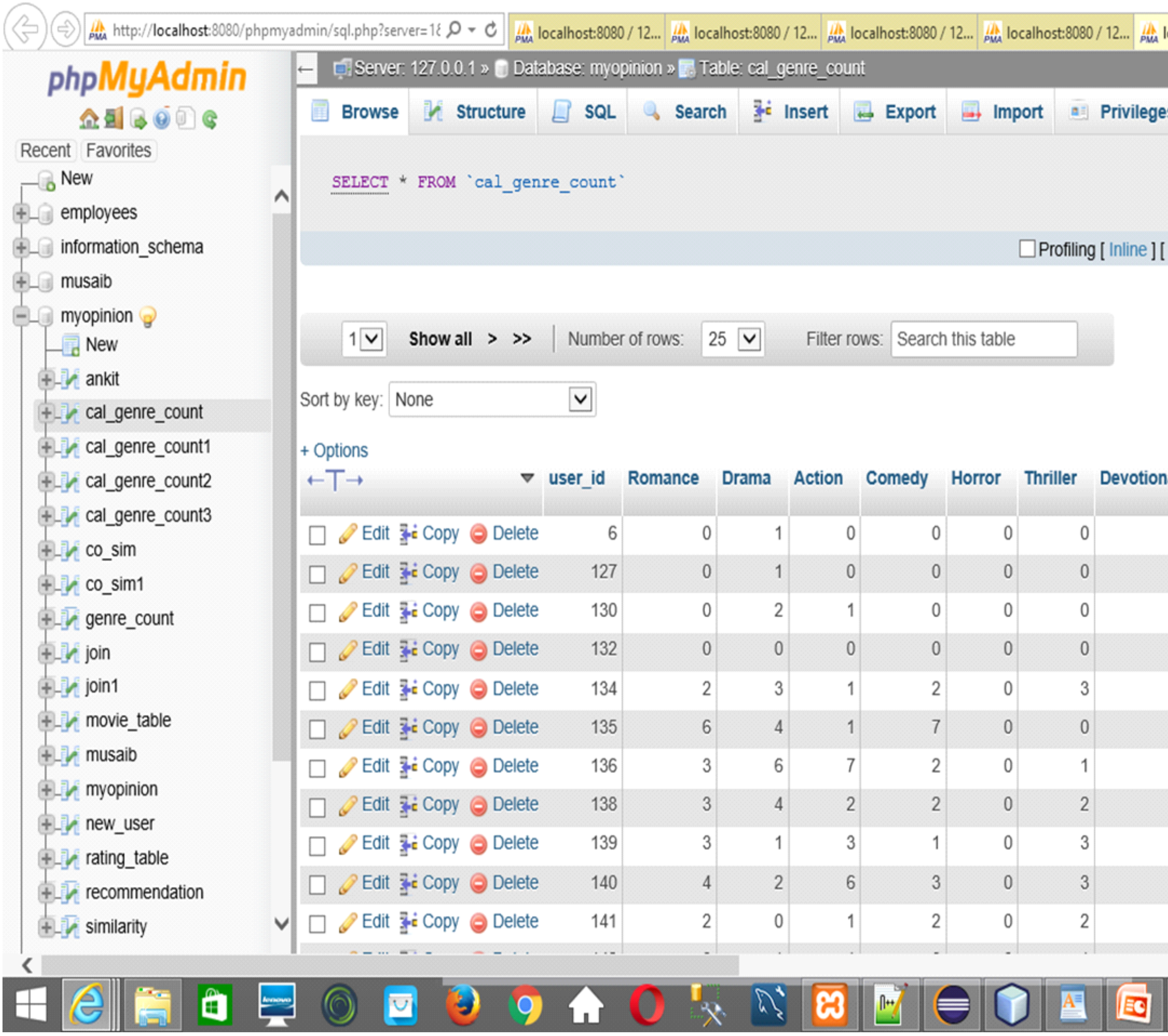
3. Visit User Profile and count Genres of each type (say, Comedy, Drama, Romance, Thriller, etc.) using, **Collection Class** in java.

4. For further process, we have stored User_Id and each Genre Count of all the Users into a table (named, cal genre Count) in the Database (named, myopinion).

5. Corresponding to every user_id in a table we have generated a **n-dimensional vector** of Genre Types through java code using **Array List**, where n denotes the total number of Genre Types.

6. We now calculate the **Vector Product**(i.e. Similarity) of Target User Profile with all the other Existing User's Profile on the Website using, **Cosine Similarity Method**.

CONTINUED...



```
SELECT * FROM `cal_genre_count`
```

1 Show all > >> Number of rows: 25 Filter rows: Search this table

Sort by key: None

	user_id	Romance	Drama	Action	Comedy	Horror	Thriller	Devotion
□ Edit Copy Delete	6	0	1	0	0	0	0	
□ Edit Copy Delete	127	0	1	0	0	0	0	
□ Edit Copy Delete	130	0	2	1	0	0	0	
□ Edit Copy Delete	132	0	0	0	0	0	0	
□ Edit Copy Delete	134	2	3	1	2	0	3	
□ Edit Copy Delete	135	6	4	1	7	0	0	
□ Edit Copy Delete	136	3	6	7	2	0	1	
□ Edit Copy Delete	138	3	4	2	2	0	2	
□ Edit Copy Delete	139	3	1	3	1	0	3	
□ Edit Copy Delete	140	4	2	6	3	0	3	
□ Edit Copy Delete	141	2	0	1	2	0	2	

Table name: **cal_genre_count**

CONTINUED...

Cosine Similarity Method

Cosine similarity is a measure of similarity between two non-zero vectors of an inner product space that measures the cosine of the angle between them. The cosine of 0° is 1, and it is less than 1 for any other angle. It is thus a judgment of orientation and not magnitude: two vectors with the same orientation have a cosine similarity of 1, two vectors at 90° have a similarity of 0, and two vectors diametrically opposed have a similarity of -1, independent of their magnitude. Cosine similarity is particularly used in positive space, where the outcome is neatly bounded in [0,1].

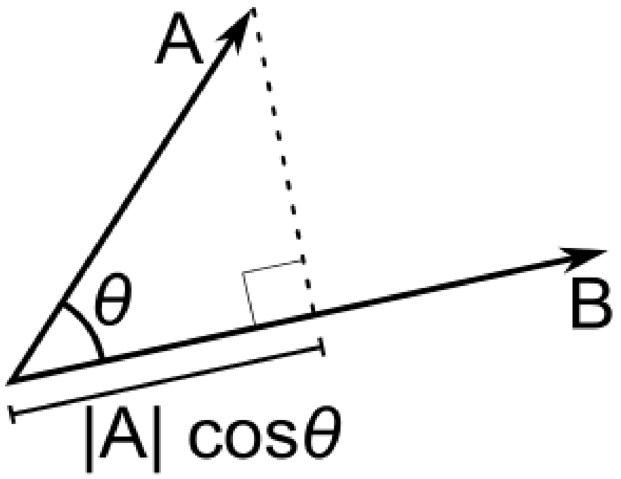


Fig.- The projection of the vector A into the vector B.

The cosine of two non-zero vectors can be derived by using the Euclidean dot product formula:

d1·d2 = ‖d1‖*‖d2‖*cos(θ) ;

Mathematically the cosine similarity is expressed by the following formula:

[similarity = cos(θ) = **cos(d1, d2)** = (d1·d2) / ‖d1‖*‖d‖ **]**;

where "·" indicates vector dot product of d1 & d2 and ‖d‖ denotes the length of vector d.

Example: Find similarity between documents 1 and 2.

d1 = (5, 0, 3, 0, 2, 0, 0, 2, 0, 0)

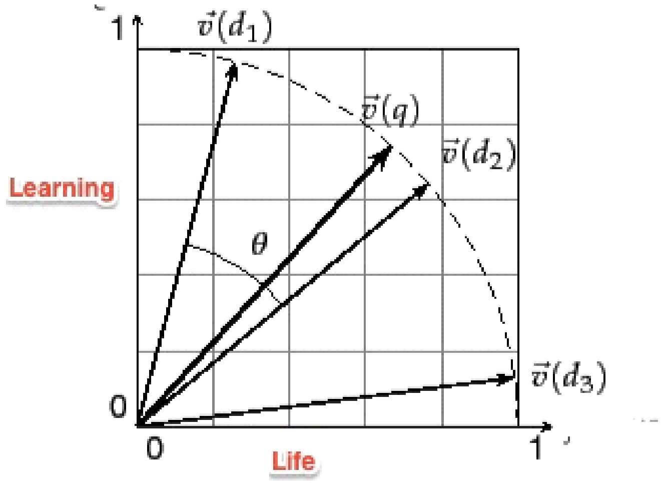
d2 = (3, 0, 2, 0, 1, 1, 0, 1, 0, 1)

d1 . d2 = 5*3+0*0+3*2+0*0+2*1+0*1+0*1+2*1+0*0+0*1 = 25

||d1|| = (5*5+0*0+3*3+0*0+2*2+0*0+0*0+2*2+0*0+0*0)0.5 = (42) 0.5 = 6.481

||d2|| = (3*3+0*0+2*2+0*0+1*1+1*1+0*0+1*1+0*0+1*1)0.5 = (17) 0.5 = 4.12

cos(d1, d2) = 0.94

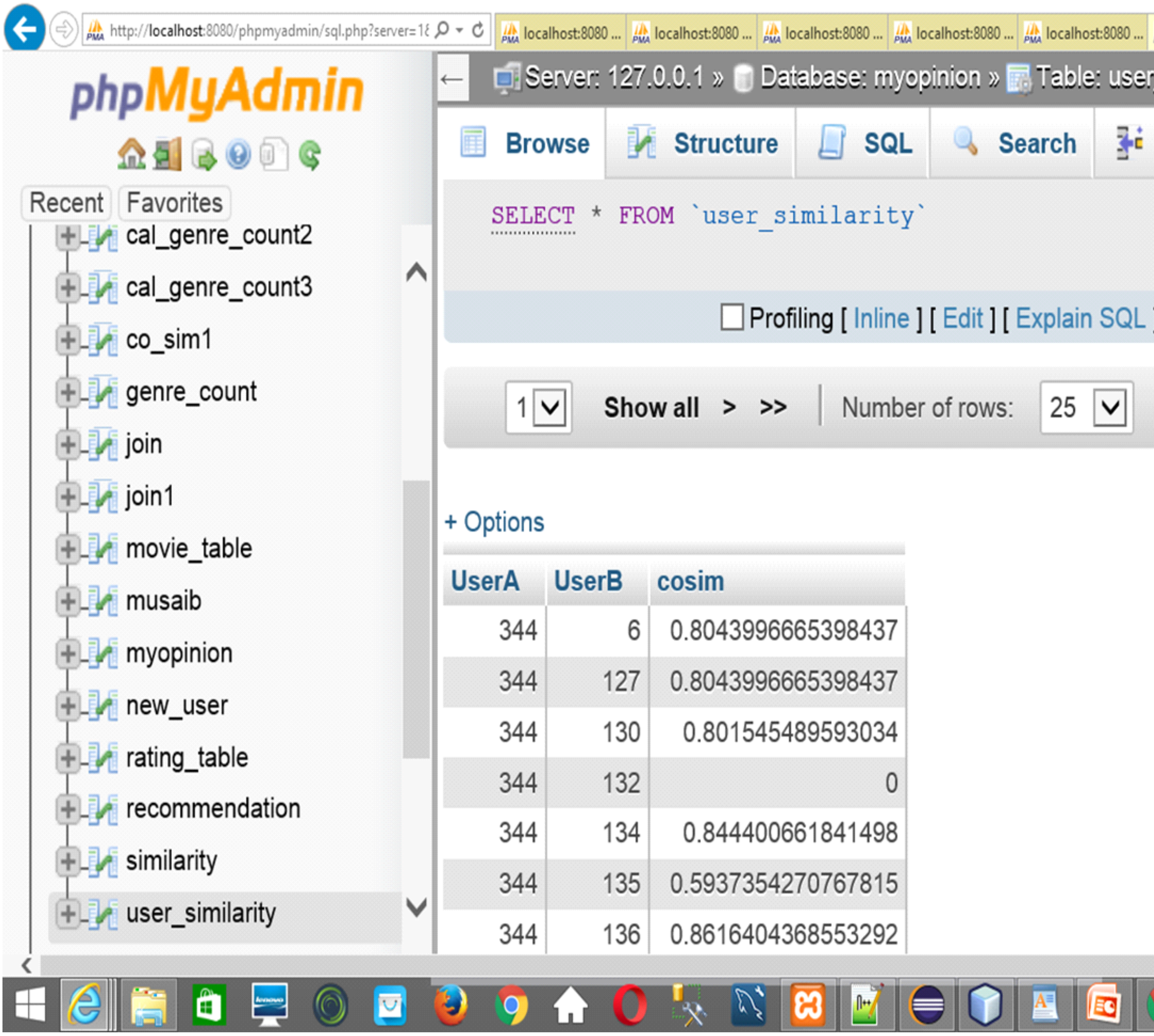


# Clearly, it can be seen from the graph that:- The closeness/similarity between the vectors is inversely proportional to the cosine of the angle between them, i.e. **the profiles will be more similar at lesser angles**.

7. For future reference, Store the Cosine/Similarities of Target user_id with the corresponding user_id into a table (named, user similarity). Now, sort the table in the decreasing order of their Similarity with the Target User.

8. Now, sort the table in the decreasing order of their Similarity with the Target User.

9. Clearly, the User appears at the top of the table is said to be the most similar user with the Target User. More clearly, they likes almost same type of movies.



```
SELECT * FROM `user_similarity`
```

UserA	UserB	cosim
344	6	0.8043996665398437
344	127	0.8043996665398437
344	130	0.801545489593034
344	132	0
344	134	0.844400661841498
344	135	0.5937354270767815
344	136	0.8616404368553292

Table name: **user_similarity**

# Objective - 2,3 & 4

Content based similarity between User profile and Item profile and Recommender System

Introduction

Recommender systems have the effect of guiding users in a personalized way to interesting movies in a large space of possible options. Content-based Recommendation Systems try to recommend items similar to those a given user has rated in the past. Indeed, the basic process performed by a content-based recommender consists in matching up the contents of a user profile in which preferences and interests are stored, with the attributes of a content object (item), in order to recommend to the user new interesting items.

The abundance of information available on the Web and in Digital Libraries, has determined a rapidly increasing difficulty in finding what we want, when we need it and in a manner which best meets our requirements.

As a consequence, the role of user modeling and personalized information access is becoming crucial: users need a personalized support in sifting through large amounts of available information, according to their interests and tastes.

Many information sources embody recommender systems as a way of personalizing their content for users. Recommender systems have the effect of guiding users in a personalized way to interesting or useful objects in a large space of possible options Recommendation algorithms use input about a customer's interests to generate a list of recommended items. At Amazon.com, recommendation algorithms are used to personalize the online store for each customer, for example showing programming titles to a software engineer and baby toys to a new mother.

First, we present the basic concepts and terminology related to content-based recommenders. A classical framework for providing content-based recommendations is described, in order to understand the main components of the architecture, the process for producing recommendations and the advantages and drawbacks of using this kind of recommendation technique. provides a thorough review of the state of the art of content-based systems, by providing details about the classical and advanced techniques for representing items to be recommended, and the methods for learning user profiles.

Item Representation

Items that can be recommended to the user are represented by a set of features, also called attributes or properties. For example, in a movie recommendation application, features adopted to describe a movie are: actors, directors, genres, subject matter, . . .). When each item is described by the same set of attributes, and there is a known set of values the attributes may take, the item is represented by means of structured data.

In most content-based filtering systems, item descriptions are textual features extracted from Web pages, emails, news articles or product descriptions. Unlike structured data, there are no attributes with well-defined values. Textual features create a number of complications when learning a user profile, due to the natural language ambiguity.

- POLYSEMY, the presence of multiple meanings for one word;

- SYNONYMY, multiple words with the same meaning.

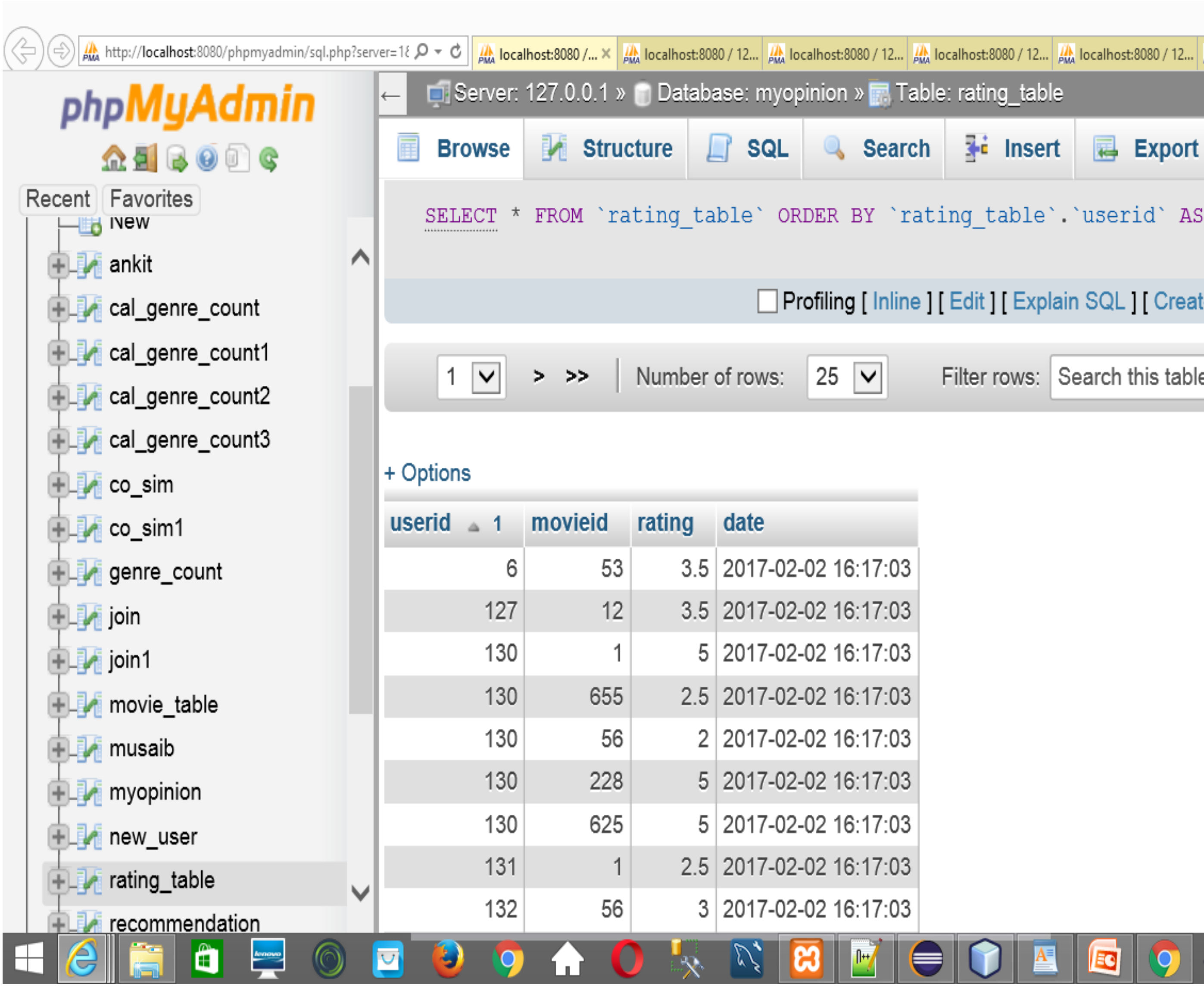
Clearly, due to synonymy, relevant information can be missed if the profile does not contain the exact keywords in the documents while, due to polysemy, wrong documents could be deemed relevant.

Some basic terminologies

- Item **-->** Movie
- Item Profile **-->** Consists of Genres of an Item(i.e. Movie)
- Item/Movie **-->** Identified by movie_id
- User_Id **-->** Identified by user_id
- Target User **-->** An Active User who is currently logged in to the Website.
- User Profile **-->** Consists a list of Genres of all the movies rated by the User.

Two basic tables

- rating_table **-->** It consists user id, movie id,
  rating, date as the 'Attributes'.
  It contains all the entries of every Active
  User as 'tuples'.
- movie_ table **-->** It consists movie id, title, genre
  avg rating, image and other 'Attributes'.



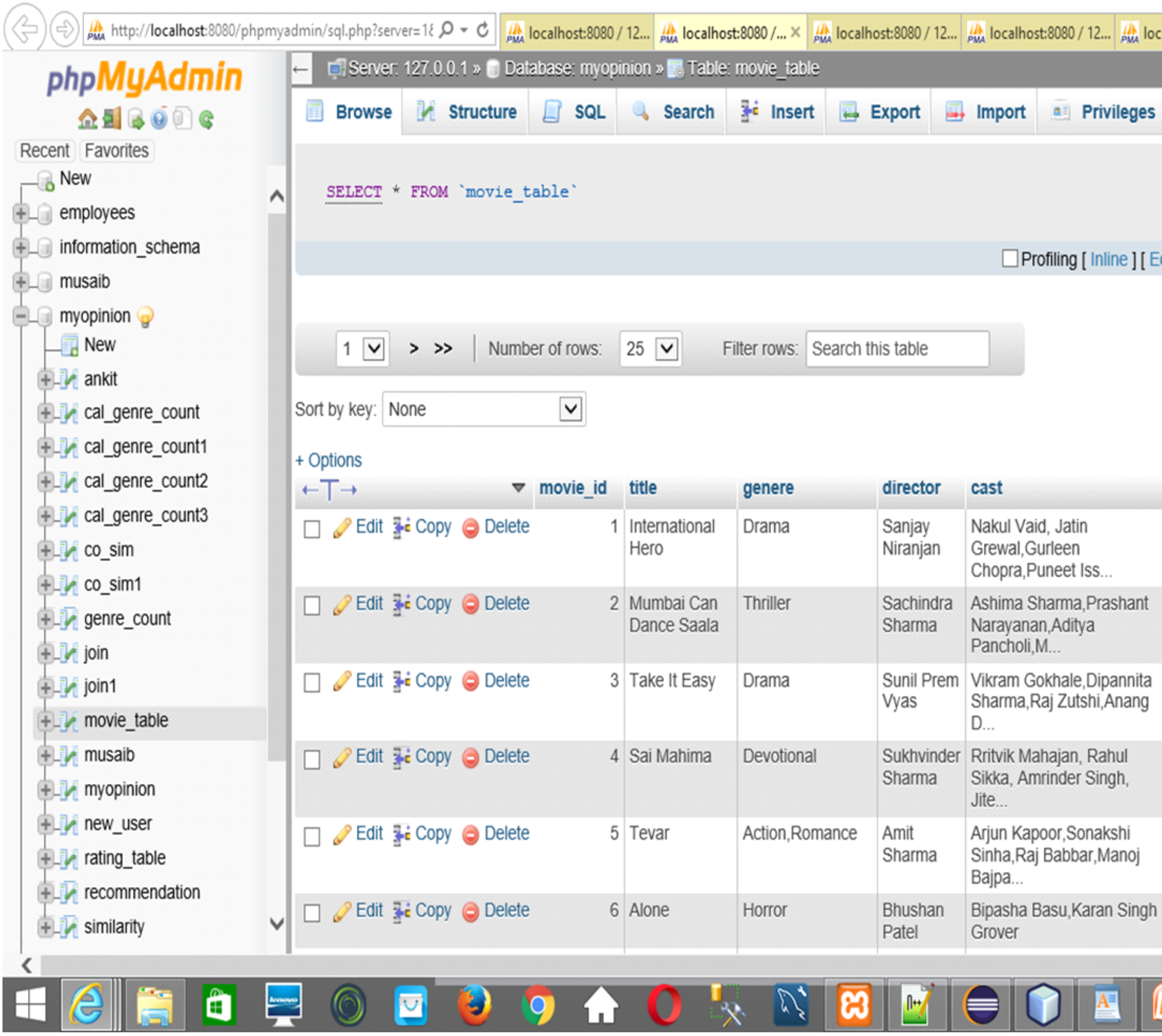
```
SELECT * FROM `rating_table` ORDER BY `rating_table`.`userid` AS
```

□ Profiling

1 | Number of rows: 25 | Filter rows: Search this table

userid ▲ 1	movieid	rating	date
6	53	3.5	2017-02-02 16:17:03
127	12	3.5	2017-02-02 16:17:03
130	1	5	2017-02-02 16:17:03
130	655	2.5	2017-02-02 16:17:03
130	56	2	2017-02-02 16:17:03
130	228	5	2017-02-02 16:17:03
130	625	5	2017-02-02 16:17:03
131	1	2.5	2017-02-02 16:17:03
132	56	3	2017-02-02 16:17:03

Tablename: **rating_table**



```
SELECT * FROM `movie_table`
```

Number of rows: 25

Filter rows: Search this table

Sort by key: None

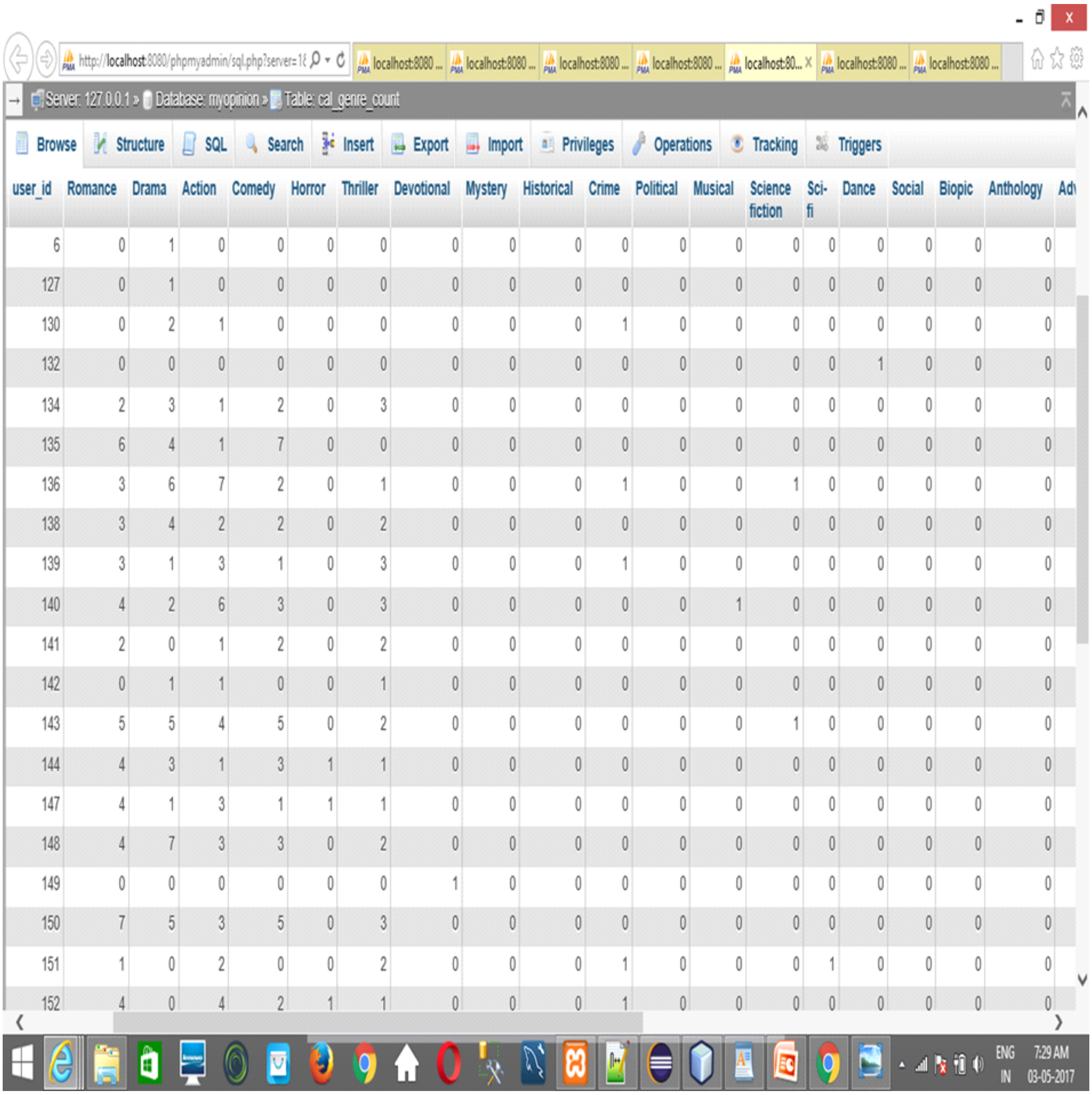
movie_id	title	genere	director	cast
1	International Hero	Drama	Sanjay Niranjan	Nakul Vaid, Jatin Grewal,Gurleen Chopra,Puneet Iss...
2	Mumbai Can Dance Saala	Thriller	Sachindra Sharma	Ashima Sharma,Prashant Narayanan,Aditya Pancholi,M...
3	Take It Easy	Drama	Sunil Prem Vyas	Vikram Gokhale,Dipannita Sharma,Raj Zutshi,Anang D...
4	Sai Mahima	Devotional	Sukhvinder Sharma	Rritvik Mahajan, Rahul Sikka, Amrinder Singh, Jite...
5	Tevar	Action,Romance	Amit Sharma	Arjun Kapoor,Sonakshi Sinha,Raj Babbar,Manoj Bajpa...
6	Alone	Horror	Bhushan Patel	Bipasha Basu,Karan Singh Grover

Table name: **movie_table**

PROCEDURE

- We have fetched all the records of each User from the join of rating table with movie table.
- Build User Profile of every user based on his Content (i.e. Genre of his rated Items).
- Create User Profile Vector which is the vector representation of the counts of each of the Type Genre present in the User Profile.
- Similarly, we have fetched all the Items from movie Table and then build Item Profile of each item based on it's Genre.
- Create Item Profile Vector which is the vector representation of the counts of each of the Type Genre present in the Item Profile.

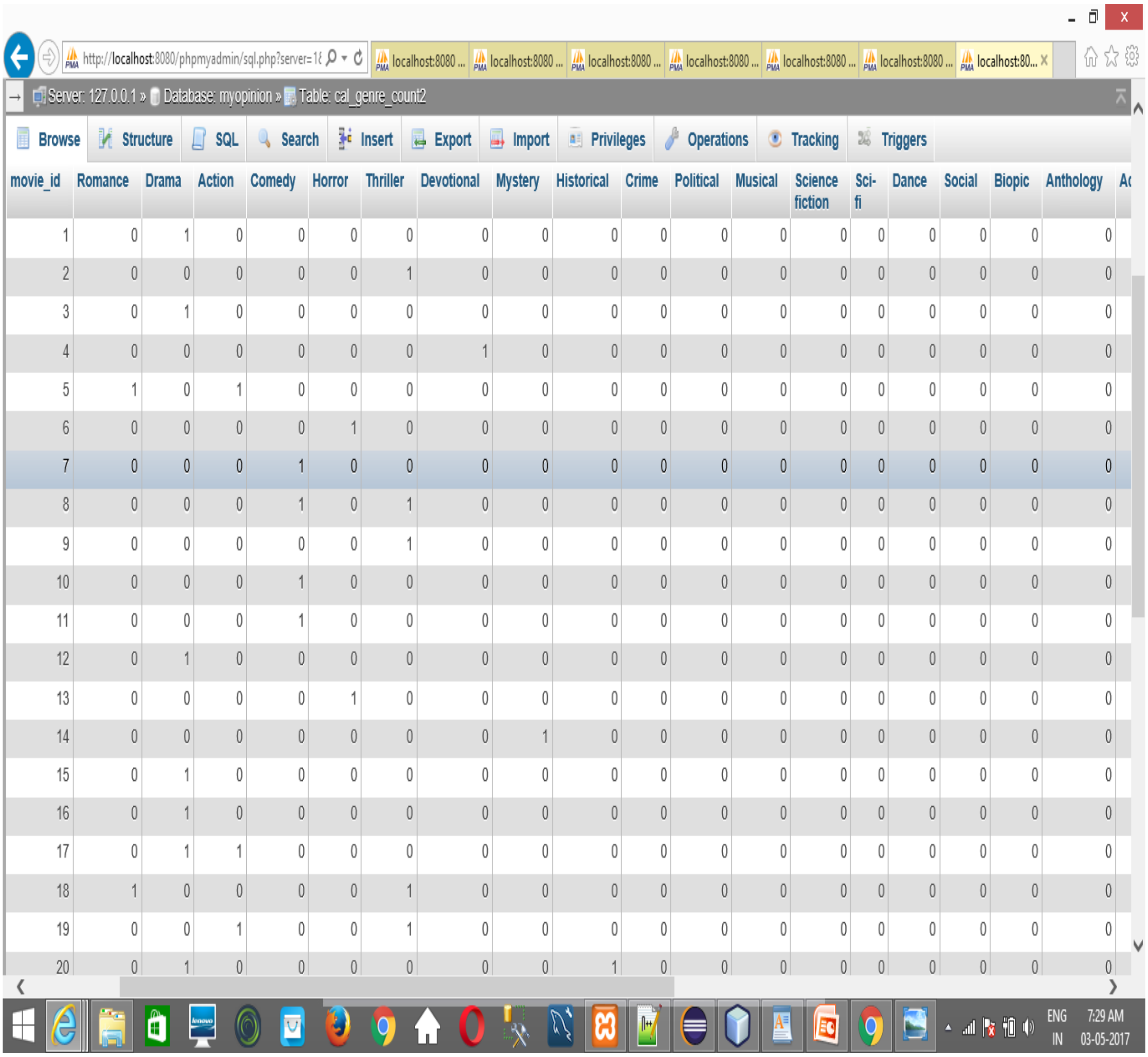
# Note: Dimension of the Vector will be the total number of available Type Genres. If there are n Type Genres, the Vector will be a n-dimensional vector.



Server: 127.0.0.1 » Database: myopinion » Table: cal_genre_count

user_id	Romance	Drama	Action	Comedy	Horror	Thriller	Devotional	Mystery	Historical	Crime	Political	Musical	Science fiction	Sci-fi	Dance	Social	Biopic	Anthology	Adv
6	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
127	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
130	0	2	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	
132	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	
134	2	3	1	2	0	3	0	0	0	0	0	0	0	0	0	0	0	0	
135	6	4	1	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
136	3	6	7	2	0	1	0	0	0	1	0	0	1	0	0	0	0	0	
138	3	4	2	2	0	2	0	0	0	0	0	0	0	0	0	0	0	0	
139	3	1	3	1	0	3	0	0	0	1	0	0	0	0	0	0	0	0	
140	4	2	6	3	0	3	0	0	0	0	0	1	0	0	0	0	0	0	
141	2	0	1	2	0	2	0	0	0	0	0	0	0	0	0	0	0	0	
142	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
143	5	5	4	5	0	2	0	0	0	0	0	0	1	0	0	0	0	0	
144	4	3	1	3	1	1	0	0	0	0	0	0	0	0	0	0	0	0	
147	4	1	3	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	
148	4	7	3	3	0	2	0	0	0	0	0	0	0	0	0	0	0	0	
149	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	
150	7	5	3	5	0	3	0	0	0	0	0	0	0	0	0	0	0	0	
151	1	0	2	0	0	2	0	0	0	1	0	0	0	1	0	0	0	0	
152	4	0	4	2	1	1	0	0	0	1	0	0	0	0	0	0	0	0	

Fig.: **User Profile**



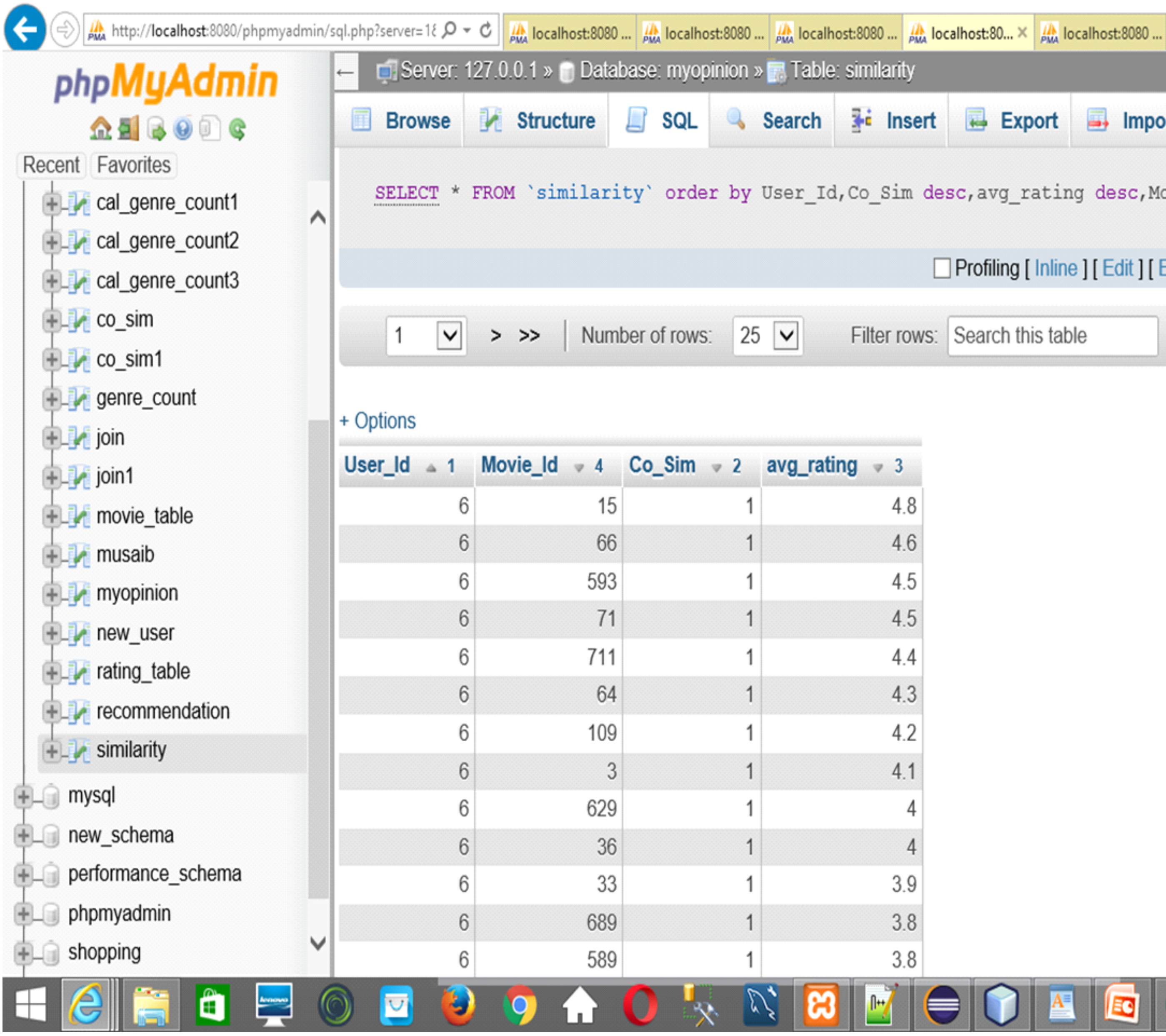
Server: 127.0.0.1 » Database: myopinion » Table: cal_genre_count2

movie_id	Romance	Drama	Action	Comedy	Horror	Thriller	Devotional	Mystery	Historical	Crime	Political	Musical	Science fiction	Sci-fi	Dance	Social	Biopic	Anthology	Ac
1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
2	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
3	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
4	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	
5	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
6	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	
7	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
8	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
9	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
10	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
11	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
12	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
13	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	
14	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	
15	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
16	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
17	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
18	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
19	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
20	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	

Fig.: **Item Profile**

COMPUTE USER-ITEM PROFILE SIMILARITY USING COSINE SIMILARITY

- Cosine Similarity Method: It gives the Vector Product (i.e. Similarity) between two vectors (i.e. User Profile Vector and Item Profile Vector). It has already been discussed in the previous Objective.
- Using this method, we will calculate the Similarity of Target User Profile with all the Item Profiles (excluding those which are rated by him).
- Finally, for the Future Reference, Store the Similarities of the Target user id with the corresponding movie id into a table (named, similarity).



```
SELECT * FROM `similarity` order by User_Id,Co_Sim desc,avg_rating desc,Mo
```

User_Id ▲ 1	Movie_Id ▼ 4	Co_Sim ▼ 2	avg_rating ▼ 3
6	15	1	4.8
6	66	1	4.6
6	593	1	4.5
6	71	1	4.5
6	711	1	4.4
6	64	1	4.3
6	109	1	4.2
6	3	1	4.1
6	629	1	4
6	36	1	4
6	33	1	3.9
6	689	1	3.8
6	589	1	3.8

Table name: **similarity table**

COMPUTE AVERAGE RATING

- Average Rating is the average of all the ratings of an Item.
- As soon as a new record is available in the rating table, firstly, we fetch the movie id from that record.
- Then, fetch all the records (which contains this movie id) from the rating table. And finally, calculate the Average using the Aggregate Function (AVG).
- Lastly , Update the calculated Average in the movie table in the database.

CONTENT BASED RECOMMENDATION SYSTEM

- Content based recommendation systems try to recommend items similar to those a given user has rated in the past.
- Each time when a User logged in to the Website, he will be provided updated recommendation of items based on his previous past history.
- In further slides, we have discussed in detail that -- How to produce updated recommendation.

TABLE USED

- **rating_table -->** It consists user_id, movie_id, rating and date as the 'Attributes'.

  It contains all the entries of every Active User as 'tuples'.
- **movie_table -->** It consists movie_id, title, genre avg_rating, image and other 'Attributes'.
- **similarity table -->** It consists User_Id, Movie_Id, Co_Sim, avg_rating as the 'Attributes'.
- **recommendation table -->** It consists User_Id, Movie_Id, title, genre, avg_rating, image as the "Attributes".

ROLE OF RECOMMENDER

- As soon as a User (i.e. Target User) logged in to the website, the Recommender will get his user id.
- Just after getting the user id, it looks for the new entries (if available) in the rating table corresponding to that Target user id.
- If found, it will make an update on the Target User Profile followed by Profile Vector.

# Note:- In case the Target User is a New User,
the Recommender will create a New
Profile for him.

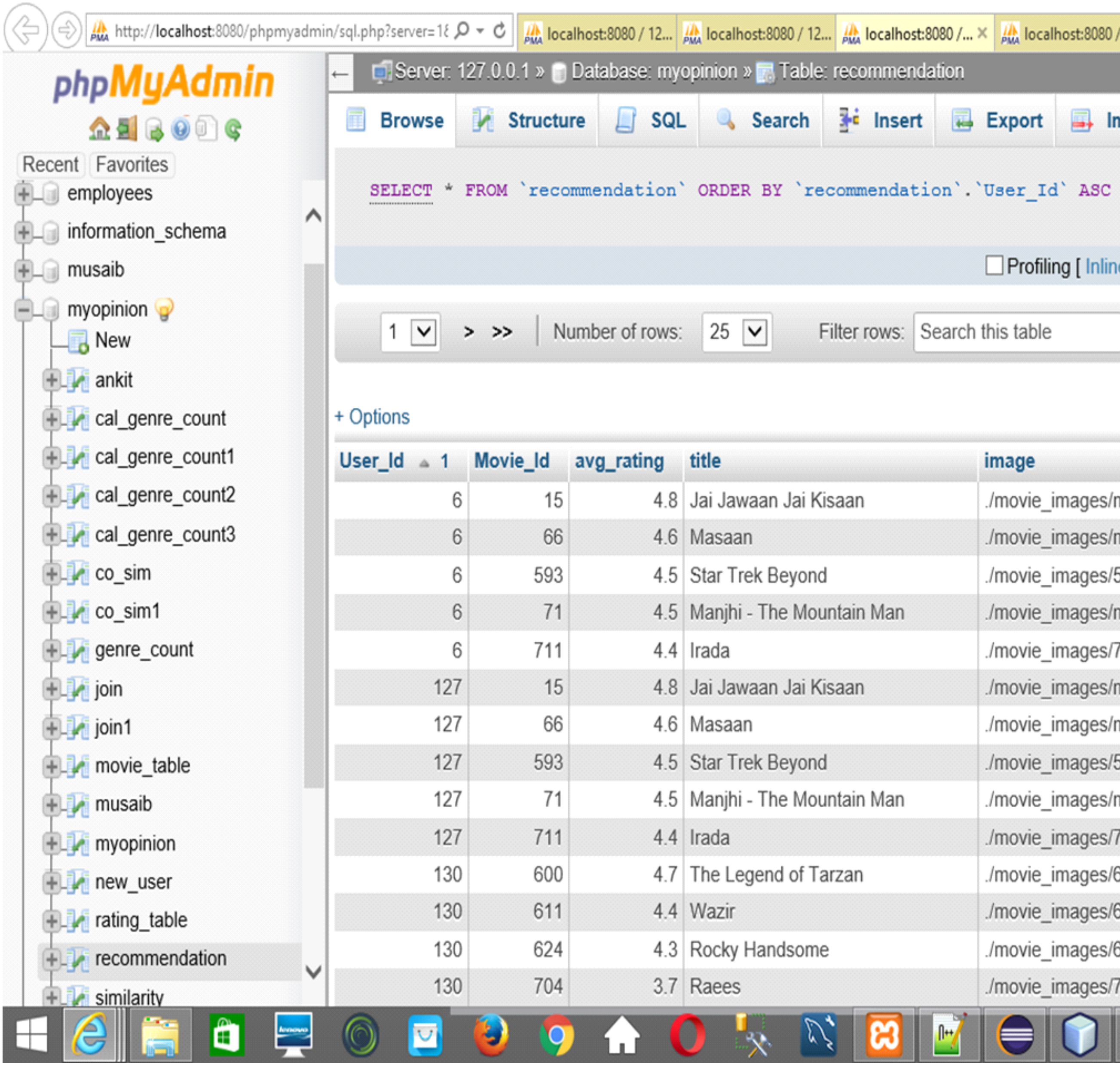
- In the mean time , his records in the similarity table will also be updated.
- Now, the Recommender will sort the similarity table in the descending order of these Attributes--

1. Co_Sim(i.e. similarity)
2. avg_rating
3. Movie_Id

- Clearly, Items with the maximum similarities (w.r.t Target User Profile) followed by highest average ratings followed by latest movies will be appeared at the Top of the Table.
- All these Three Attribute Properties forms the '**Basis of Recommendation**'.
- Now, the Recommender will select the Top Five Items from the similarity table and Update the previous records of the corresponding Target User with these items in the recommendation table.

- Hence, our recommendation table gets Updated.
- Finally, the Target User will be provided this updated recommendations on his next Login Session.
- **On the other hand,** if the Recommender can't find any new entry in the rating_table corresponding to the Target user_id, then in that case--
 --> What our Recommrnder will do ???
- In that case, User Profile and Similarity factor remains unaffected.
- So, avg_rating will only be the basis of recommendation because avg_rating is regularly modified by the Existing Active Users, not only the Target User.
- Hence, the Recommender will select Items with higher avg_rating from the similarity table corresponding to the Target user_id and Update them in the recommendation table.
- Finally, he too will get the updated recommendations by the Recommender.

Continued...



```
SELECT * FROM `recommendation` ORDER BY `recommendation`.`User_Id` ASC
```

□ Profiling [Inlin

1 > >> Number of rows: 25 Filter rows: Search this table

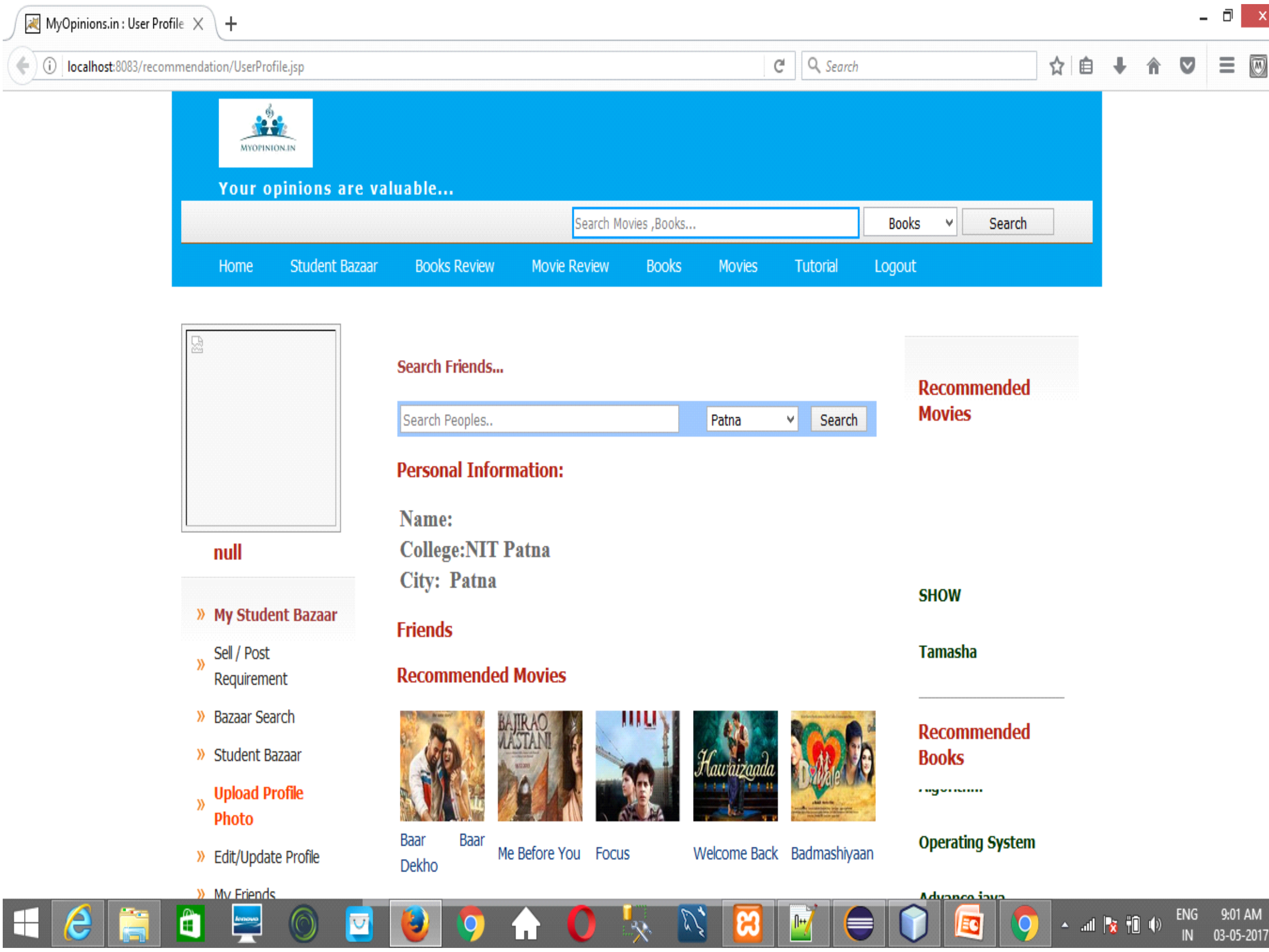
User_Id ▲ 1	Movie_Id	avg_rating	title	image
6	15	4.8	Jai Jawaan Jai Kisaan	./movie_images/n
6	66	4.6	Masaan	./movie_images/n
6	593	4.5	Star Trek Beyond	./movie_images/5
6	71	4.5	Manjhi - The Mountain Man	./movie_images/n
6	711	4.4	Irada	./movie_images/7
127	15	4.8	Jai Jawaan Jai Kisaan	./movie_images/n
127	66	4.6	Masaan	./movie_images/n
127	593	4.5	Star Trek Beyond	./movie_images/5
127	71	4.5	Manjhi - The Mountain Man	./movie_images/n
127	711	4.4	Irada	./movie_images/7
130	600	4.7	The Legend of Tarzan	./movie_images/6
130	611	4.4	Wazir	./movie_images/6
130	624	4.3	Rocky Handsome	./movie_images/6
130	704	3.7	Raees	./movie_images/7

Table name: **recommendation table**

RECOMMENDATIONS GENERATED ON THE WEBPAGE OF THE TARGET USER

- Once a User logged in to the Website, we will get his user id from his login.
- We then pass that user id to the Recommender through the JSP Code.
- The Recommender then fetched the recommendations for that User from the recommendation table and redirect them to the Web Page(UserProfile.jsp) of that User via the Browser(Mozilla Firefox/Google Chrome).
- The Screenshot of the UserProfile.jsp page of the Target User which includes recommendations for him is shown in the next page...

Continued...



Your opinions are valuable...

null

- My Student Bazaar
- Sell / Post Requirement
- Bazaar Search
- Student Bazaar
- Upload Profile Photo
- Edit/Update Profile
- My Friends

Search Friends...

Personal Information:

Name:
College:NIT Patna
City: Patna

Friends

Recommended Movies

Baar Baar Dekho | Me Before You | Focus | Welcome Back | Badmashiyaan

Recommended Movies

SHOW

Tamasha

Recommended Books

Algorithm...

Operating System

Advance java

Web Page: **UserProfile.jsp**

ADVANTAGES OF CONTENT BASED APPROACH FOR RECOMMENDATION

The Content-based Recommendation paradigm has several advantages :

- USER INDEPENDENCE - Content-based recommenders exploit solely ratings provided by the active user to build his own profile. Instead, some other methods need ratings from other users in order to find the "nearest neighbors" of the active user, i.e., users that have similar tastes since they rated the same items similarly.

- TRANSPARENCY - Explanations on how the recommender system works can be provided by explicitly listing content features or descriptions that caused an item to occur in the list of recommendations

- NEW ITEM - Content-based recommenders are capable of recommending items not yet rated by any user. As a consequence, they do not suffer from the first-rater problem, i.e. until the new item is rated by a substantial number of users, the system would not be able to recommend it.

CONCLUSION

- Similarity found between user-user profiles is verified and correct.
- Profiles and Profile Vectors of the Target User and the items rated by him/her are properly constructed and verified.
- Similarity found between user-item profile is verified and correct.
- Cosine Similarity Method is working well in recommending items by our Recommender.
- Any entry of a New User is soon recognized and passed on to our Recommender System.
- Recommendations provided to Users(Existing User/New User) is based on both User-Item Similarity and Average Rating of Items.

BIBLIOGRAPHY

The following book were used for the completion of the goal of the project :-

1. JAVA 7 PROGRAMMING BLACK BOOK.

2. SERVELET AND JSP BLACK BOOK.

3. WEB TECHNOLOGY BLACK BOOK.

4. RECOMMENDER SYSTEM BOOK BY CHARU AGARWAL

5. PROGRAMMING IN JAVA BOOK BY SACHIN MALHOTRA

Also , the following websites were consulted for relevant information :-

1. http://www.javatpoint.com

2. http://www.tutorialspoint.com

3. http://www.google.co.in