2013

Recommender System Using Collaborative Filtering Algorithm

Ala Alluhaidan
Grand Valley State University

Recommender System Using Collaborative Filtering Algorithm

By

Ala S. Alluhaidan

April, 2013

Recommender System Using Collaborative Filtering Algorithm

By

Ala S. Alluhaidan

A project submitted in partial fulfillment of the requirements for the degree of

Master of Science in

Computer Information Systems

at

Grand Valley State University

April, 2013

Jerry Scripps **Date**

CONTENTS

ABSTRACT

"More Data Beats Better Algorithms"

Omar Tawakol, CEO, BlueKai, 2012

With the vast amount of data that the world has nowadays, institutions are looking for more and more accurate ways of using this data. Companies like Amazon use their huge amounts of data to give recommendations for users. Based on similarities among items, systems can give predictions for a new item's rating. Recommender systems use the user, item, and ratings information to predict how other users will like a particular item.

Recommender systems are now pervasive and seek to make profit out of customers or successfully meet their needs. However, to reach this goal, systems need to parse a lot of data and collect information, sometimes from different resources, and predict how the user will like the product or item. The computation power needed is considerable. Also, companies try to avoid flooding customer mailboxes with hundreds of products each morning, thus they are looking for one email or text that will make the customer look and act.

The motivation to do the project comes from my eagerness to learn website design and get a deep understanding of recommender systems. Applying machine learning dynamically is one of the goals that I set for myself and I wanted to go beyond that and verify my result. Thus, I had to use a large dataset to test the algorithm and compare each technique in terms of error rate. My experience with applying collaborative filtering helps me to understand that finding a solution is not enough, but to strive for a fast and ultimate one. In my case, testing my algorithm in a large data set required me to refine the coding strategy of the algorithm many times to speed the process.

In this project, I have designed a website that uses different techniques for recommendations. User-based, Item-based, and Model-based approaches of collaborative filtering are what I have used. Every technique has its way of predicting the user rating for a new item based on existing users' data. To evaluate each method, I used Movie Lens, an external data set of users, items, and ratings, and calculated the error rate using Mean Absolute Error Rate (MAE) and Root Mean Squared Error (RMSE). Finally, each method has its strengths and weaknesses that relate to the domain in which I am applying these methods.

INTRODUCTION

Recommender systems are now pervasive in consumers' lives. They aim to help users in finding items that they would like to buy or consider based on huge amounts of data collected. Amazon, Facebook, LinkedIn, and other commercial and social networking websites use these systems. Parsing a huge amount of data to predict a user's preference or his or her similarity with other group of users is the core of a recommender system. Collaborative Filtering, Content-based Filtering, and Hybrid filtering are all approaches to apply a recommender system. My goal is to apply a collaborative filtering algorithm in a rating website that collects users' information, such as location and gender, item's information, such as category and description, as well as ratings for items by users.

There are many algorithms that could be applied on data to predict a user preference. User-based, Item-based, and Model-based methods are ways of predicting a user preference. The number of users, items, or clusters in each one respectively will determine the function performance. However, the most well-known and common one is User-based Collaborative Filtering. In this algorithm, we predict an item's rate for a user by collecting information about this user and similar users (Candillier et al., 2007).

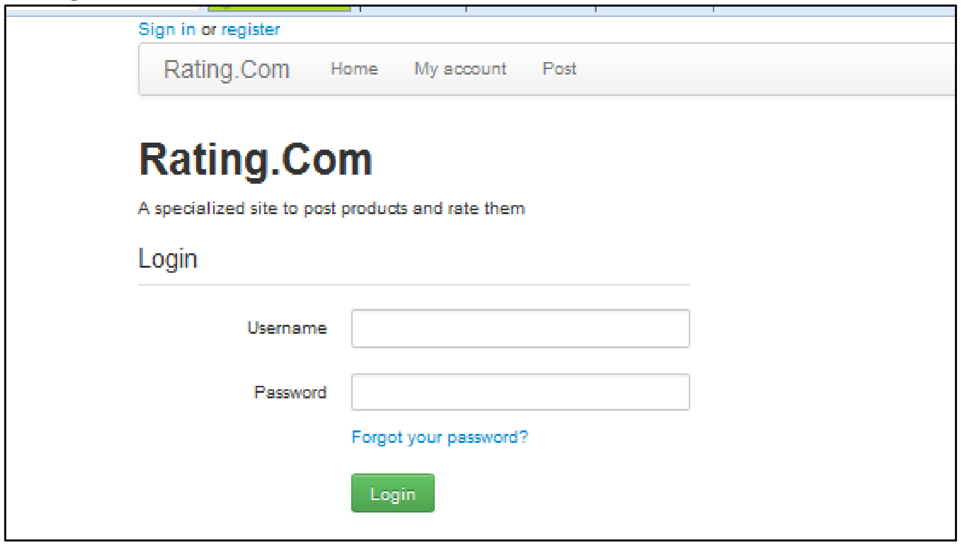
THE VEHICLE (THE WEBSITE)

A simple website, Rating.Com, was designed that contains a registration form, login form, search bar, and post form. Users can register to the system and then browse items and rate them. Also, they are able to post new items that do not exist in the system, rate them, and let other users rate them as well. When the user logs into the system, the collaborative filtering is activated to look for items that are predicted to be highly rated by the user. Then, the user can go and rate those items.

MOTIVATION

The motivation behind the project is to gain a deeper understanding of how a recommender system can be applied. Also, I want to build a website that is able to collect data and be able to parse that data dynamically using a machine learning algorithm such as collaborative filtering. Designing forms and analyzing the type of information needed to apply such a system are challenges that I could not resist. Knowing the details of the collaborative filtering algorithm and how it works is another interesting goal that I want to master. Figures 1, 2, and 3 show some of the screens that make up Rating.com.

Figures 1 and 2 show the user login and user registration forms. The post product form is displayed in Figure 3.



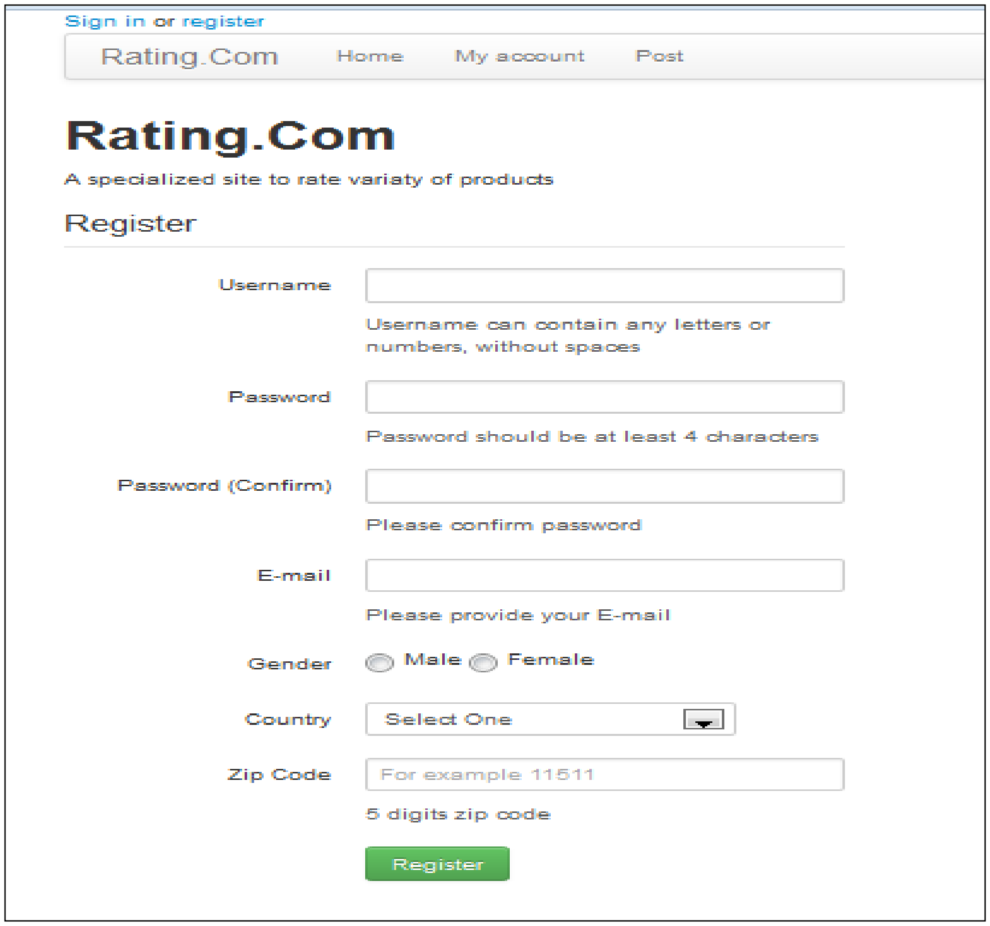
Rating.Com

A specialized site to post products and rate them

Login

Username	
Password	

Figure 1: user login form



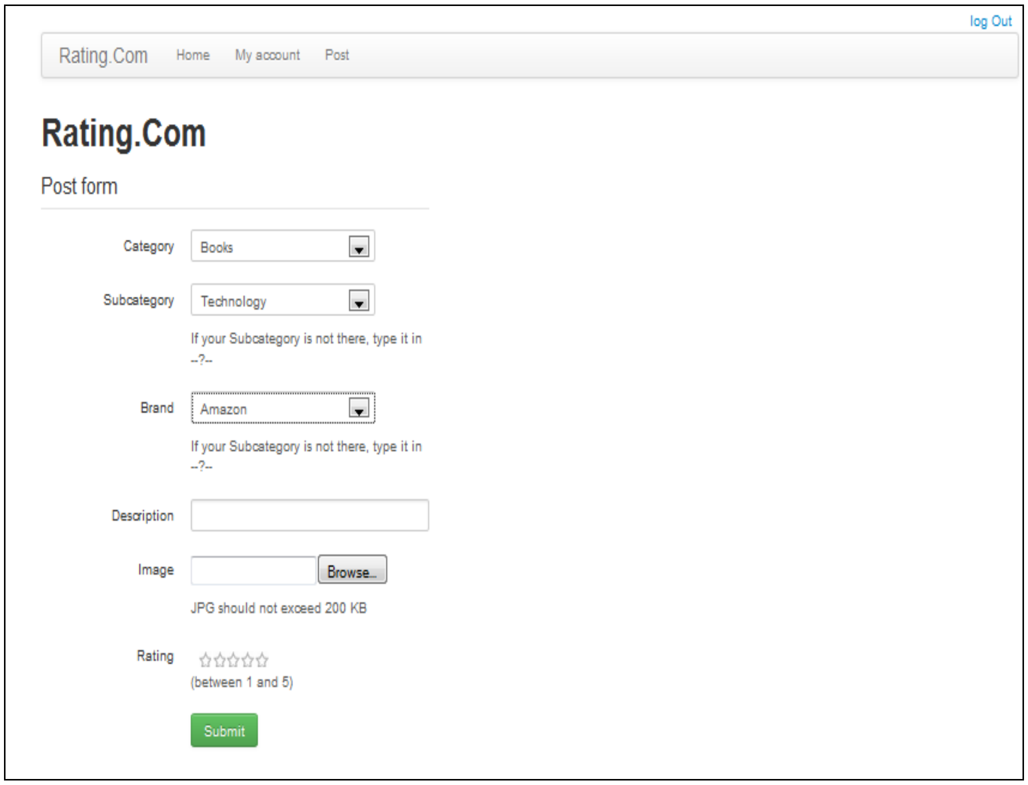
Rating.Com

A specialized site to rate variaty of products

Register

Username	Username can contain any letters or numbers, without spaces
Password	Password should be at least 4 characters
Password (Confirm)	Please confirm password
E-mail	Please provide your E-mail
Gender	Male Female
Country	Select One
Zip Code	For example 11511 5 digits zip code

Figure 2: user registration form


log Out

Rating.Com Home My account Post

Rating.Com

Post form

Category	Books
Subcategory	Technology If your Subcategory is not there, type it in --?--
Brand	Amazon If your Subcategory is not there, type it in --?--
Description	
Image	Browse... JPG should not exceed 200 KB
Rating	☆☆☆☆☆ (between 1 and 5)
	Submit

Figure 3: post product form

WHY IS IT IMPORTANT?

In the course CIS 656 Distributed Systems, we learned how to use the Hadoop framework to parse large amounts of data. However, we had to be off-line on a cluster to parse all the data. After getting the results we could inject it back into the system. Therefore, I wanted to build a system that can parse data in real-time in a more dynamic way. Also, the amount of data that systems store is huge, and recommender systems can give users a narrow scope of items that they might like or rate highly. Of course, this will save users time and make more profit for system owners if it succeeds in predicting what users want and motivates them to buy it.

GOALS AND OBJECTIVES

- **Web Programming:**
  - Learning CSS.
  - Designing an Interface using HTML5 and bootstrap 2.2.2.

- **Machine Learning**
    - Applying machine learning in real-time using Collaborative Filtering.
    - Parsing data retrieved from a database and predicting user preference.
    - Evaluating different approaches of recommender systems.

What I was trying to do was to build a system that collects information and then uses the stored data in a machine learning algorithm. Predicting users' preferences using data may give more accurate results than any algorithm that does not use previous data. Most systems like Amazon, eBay, and others suggest things to users based on similarities among users, items, or both. This will make those systems more personalized and efficient from a user's perspective. Commercial and trading systems gain trust and profits using such systems if they successfully predict what users want at what time and where.

BACKGROUND AND RELATED WORK

In the collaborative filtering algorithm, the system has a recorded set of items and users and how the users rated those items. Then algorithm is used to predict the rating for a user who has not rated the item yet. A rating for an item can be predicted from the ratings given to the item by users who are similar in taste to the given user. The available frameworks for recommender systems were made in Java like Mahout. Mahout is a well-known framework that is flexible and scalable. Also, Mahout has been integrated as a web service (What is Apache Mahout?, 2011). However, a lot of attempts were made to solve the dependency of Mahout on Maven and make the integration into web applications easy. Still, the real implementation is in Java. In this project we wanted a more flexible independent PHP library to apply the recommender system.

TRADITIONAL COLLABORATIVE FILTERING

The traditional collaborative filtering algorithms include User-based, Item-based, and Model-based methods. To explain how these methods works we are going to use the following notations. "Let U be a set of N users and I a set of M items. Vui denotes the rating of user $u \in U$ on item $i \in I$, and $S \subseteq I$ stands for the set of items that user u has rated" (Candillier et al., 2007).

USER-BASED COLLABORATIVE FILTERING:

In this method, we predict the user behavior against a certain item using the weighted sum of deviations from mean ratings of users that previously rated this item and the user mean rate. First, we calculate the user mean rate using the following formula:

$$\overline{v_u} = \frac{\sum_{i \in S_u} v_{ui}}{|S_u|}$$

(Candillier et al., 2007)

The weight that we previously mentioned can be calculated using Pearson correlation according to the following formula:

$$w(a,u) = \frac{\sum_{i \in S_a \cap S_u} (v_{ai} - \overline{v_a})(v_{ui} - \overline{v_u})}{\sqrt{\sum_{i \in S_a \cap S_u} (v_{ai} - \overline{v_a})^2 \sum_{i \in S_a \cap S_u} (v_{ui} - \overline{v_u})^2}}$$

(Candillier et al., 2007)

The prediction formula is stated as below:

$$p_{ai} = \overline{v_a} + \frac{\sum_{\{u \in U | i \in S_u\}} w(a,u) \times (v_{ui} - \overline{v_u})}{\sum_{\{u \in U | i \in S_u\}} |w(a,u)|}$$

(Candillier et al., 2007)

ITEM-BASED COLLABORATIVE FILTERING:

In reviewing the recommendation system that Amazon used, we found that they did not use the traditional collaborative filtering algorithm that was previously mentioned. To explain, User-based and Cluster models are not used in Amazon recommender system due to many reasons. Due to expensive computation O (MN), where M is the number of similar users and N is the number of common items among those users, Amazon decided not to use these methods (Linden et al., 2003).

Using clusters to reduce the number of items and users was suggested to solve the large computation problem; however, this will reduce the quality of recommendations. In other words, if the method will compare the user to a small sample, the similarity will not be accurate. Also, partitioning items to item-space will limit the recommendations to specific types of products. Additionally, if the cluster does not include the popular or unpopular items, they will never be recommended to users. Consequently, if the user already bought these items, then they will never be recommended to him or her (Linden et al., 2003).

However, in my website I decided to apply Item-based Collaborative Filtering to display similar items for the user once he or she selected a particular item using the adjusted cosine formula:

$$sim(i,j) = \frac{\sum_{\{u \in U | i \in S_u \& j \in S_u\}} (v_{ui} - \overline{v_u})(v_{uj} - \overline{v_u})}{\sqrt{\sum_{\{u \in U | i \in S_u \& j \in S_u\}} (v_{ui} - \overline{v_u})^2 \sum_{\{u \in U | i \in S_u \& j \in S_u\}} (v_{uj} - \overline{v_u})^2}}$$

(Candillier et al., 2007)

Additionally, we predicted how the user will rate this item using the previous similarity:

$$p_{ai} = \frac{\sum_{\{j \in S_a | j \neq i\}} sim(i,j) \times v_{aj}}{\sum_{\{j \in S_a | j \neq i\}} |sim(i,j)|}$$

(Candillier et al., 2007)

In this method, the system will use an unsupervised learning method to partition the space and then classify the users using a similarity metric to a segment or a cluster. To avoid having large clusters, systems use different methods to generate more small practical clusters. This process starts with an initial set of clusters that contains only one user, and then repeatedly assigns users to these clusters based on a similarity metric. Limiting features may become necessary to reduce clustering complexity. The system will create vectors for each segment and match the user to the vector. Nonetheless, the user may be classified as belonging to more than one cluster with a measure of similarity strength (Linden et al., 2003).

In the website I built I used the Euclidean distance metric to calculate the distance between users. For the clustering process I used K-means with 2 clusters due to the limited products and users I have. However, with Movie Lens dataset I have clustered the users in six groups. Where a, u are users, i is a common item and Vai is the rate given by the user a to the item i, the Euclidean distance can be calculated using the formula:

$$dist(a,u) = \sqrt{\frac{\sum_{\{i \in S_a \cap S_u\}} (v_{ai} - v_{ui})^2}{|\{i \in S_a \cap S_u\}|}}$$

(Candillier et al., 2007)

In the K-means clustering approach, the centroids will be represented in our case as a vector of items with their ratings. Thus, the rating will be the summation of ratings of the item by the users in the cluster divided by the number of users in the cluster:

$$\mu_{ki} = \frac{\sum_{\{u \in C_k | i \in S_u\}} v_{ui}}{|\{u \in C_k | i \in S_u\}|}$$

(Candillier et al., 2007)

Although clusters are more scalable and are favored for online use, the clustering process has to be done offline. Thus, we lose the advantage of dynamic clustering in real-time which affects the quality of recommendations. Also, since users are assigned to groups and are not compared to most similar users, the recommendations quality is reduced. The solution for this is to generate more refined clusters which will become computationally expensive, similar to the other traditional collaborative filtering approaches such as User-based and Item-based (Linden et al., 2003).

SEARCH BASED METHODS

The search based method relies on finding similar popular items to the one that the user has bought or rated highly. This method works by constructing a search query that looks for products with the same characteristics that the user has rated highly. For example, if the user rated high all Comedy movies by Tyler Perry, the system will recommend any popular movie by Taylor Berry. The performance of this approach depends on how many purchases that users made. If there are few transactions, the performance will be good. However, if the users have many purchases, the performance decreases.

The suggested solution is to use a subset of transactions for recommendations which will definitely affect the recommendation quality. Finding popular items that share one or more characteristics is not a customized system. Although, the core of recommender systems is to customize the web for every user, this will not be covered by this method (Linden et al., 2003).

AMAZON’S ITEM-TO-ITEM COLLABORATIVE FILTERING

Instead, Amazon uses Item-to-item Collaborative Filtering. This method, according to Linden (2003) is scalable and produces high quality recommendations. Also, they provide a feature where you can see what is recommended to you and edit your filter of recommendations by product line and subject. In addition they can access those products as well as their previous purchases to rate them.

Linden describes Amazon’s method Item-to-item by taking the item that a user has rated highly then finding similar items. Then, the system will construct a matrix of similar items that users tend to rate highly. After that, the system will use this matrix to recommend new items for users. This process is done offline and the matrix is built by iterating through items and building a similarity table. The pseudo code is given as follows: (Linden et al., 2003)

```
For each item in product catalog, $I_1$
  For each customer $C$ who purchased $I_1$
    For each item $I_2$ purchased by
       customer $C$
      Record that a customer purchased $I_1$
       and $I_2$
  For each item $I_2$
    Compute the similarity between $I_1$ and $I_2$
```

(Linden et al., 2003)

The cosine method is used to calculate the similarity between items and it is given as follow:

$$sim(i,j) = \frac{\sum_{\{u \in U | i \in S_u \& j \in S_u\}} (v_{ui} - \overline{v_u})(v_{uj} - \overline{v_u})}{\sqrt{\sum_{\{u \in U | i \in S_u \& j \in S_u\}} (v_{ui} - \overline{v_u})^2 \sum_{\{u \in U | i \in S_u \& j \in S_u\}} (v_{uj} - \overline{v_u})^2}}$$

(Candillier et al., 2007)

In this case, we will end up with a vector of each item and a list of similar items. The process of constructing this table, as mentioned earlier, is done offline and is computation intensive. The cost is estimated by O (NM), where N represents the number of items and M is the number of customers. When the user logs into the system, the algorithm works by examining this table, aggregating similar items, and then finds the popular one to recommend. However, this process depends on how many items that the user has rated or purchased (Linden et al., 2003).

Amazon justifies doing the matrix of similar items offline with the following reasons. First, the Amazon dataset is large; around 29 million users and a few million items, which makes all traditional Collaborative Filtering techniques, fall short of accommodating such scalability. The online processing of these methods makes them impractical to be used on large datasets. Again, the suggested solution of sampling or partitioning will lower the recommender quality which Amazon avoids (Linden et al., 2003).

Since the clustering is done offline and only a subset of features are compared, the quality of the resulting data is poor, making the clustering method unattractive. The suggested solution of increasing clusters makes the process computation intensive, and that is what Amazon avoids. Additionally, search based methods can build an offline index with targeting subjects. However, it lacks scalability for users with many purchases or ratings (Linden et al., 2003).

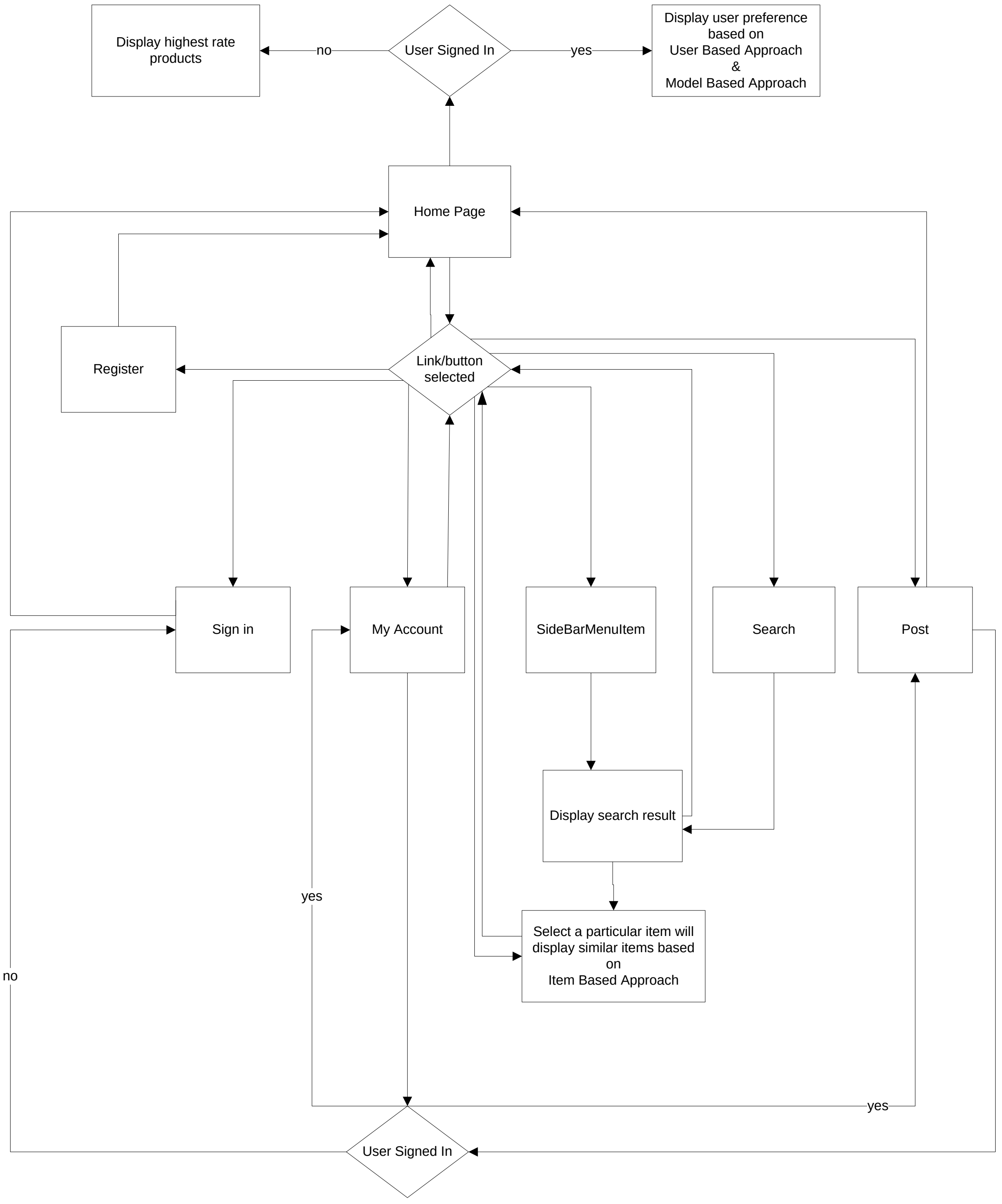
The scalability and good performance of this Item-to-item method comes from the fact that it creates a highly similar item table offline. Thus, the algorithm performance will not be affected by the number of customers. According to the Linden, Smith, and York, the quality of recommended items were high with limited user ratings or purchases in the Item-to-item method as opposed to traditional collaborative filtering methods (Linden et al., 2003). However, the number of items that have been purchased or rated by the user will affect the performance.

Amazon and other e-commerce businesses like to customize the shopping experience for every customer. A successful targeting of users' preferences is highly demanded and will potentially increase a profit. Having a dynamic response to generate new recommendations based on users' data and changing data is one goal that Amazon achieves by this method. Additionally, the scalability of this method, its ability to cover huge amounts of customers and items, and the accurate prediction of new products with even limited data are other goals that are accomplished by Item-to-item Collaborative Filtering (Linden et al., 2003).

PROGRAM REQUIREMENTS

The project consists of a website that makes the user able to browse products. Users can register to the website, post products and rate them. In addition, other products posted by other users can be rated by the user if the user is registered. Also, once the user logs into the system, the moving boxes will show the product ordered by highly predicted rating for this user.

FLOW CHART OF SYSTEM STRUCTURE


Display highest rate products
no
User Signed In
yes
Display user preference based on User Based Approach & Model Based Approach
Home Page
Register
Link/button selected
Sign in
My Account
SideBarMenuItem
Search
Post
Display search result
Select a particular item will display similar items based on Item Based Approach
yes
no
yes
User Signed In

HOME

The main page of the website has login and registration links. The login will redirect the user to a login form with formal username and password inputs. The registration form includes username, password, password confirmation, email, country, region, city, zip code, and gender. The user information is critical to build the clusters. The ability to cluster users based on country, region, city, zip code, or gender will give more accurate desired results.

The navigation bar below includes: Home, My Account, and Post. Home will redirect the user to the home page, where main categories are listed on the side bar to the left, along with a search bar in the middle. Additionally, if the user is registered, products that are predicted to be highly rated by this user will be listed below. The main categories will include subcategories, which are further broken down to brands. Users are able to click on any of these and review products that are listed under each. Further, the user can click on any product description to rate that item, if he or she is registered.

MY ACCOUNT

In my account form, the user profile will be displayed for editing. This only happens if the user has logged into the system. If the user clicked on my account before signing in, the user will be redirected to the login form. The form will include the old password as well as a new password field with a conformation if the user wants to change the password. In addition, email, country, region, city, and zip code can be edited and saved to the database.

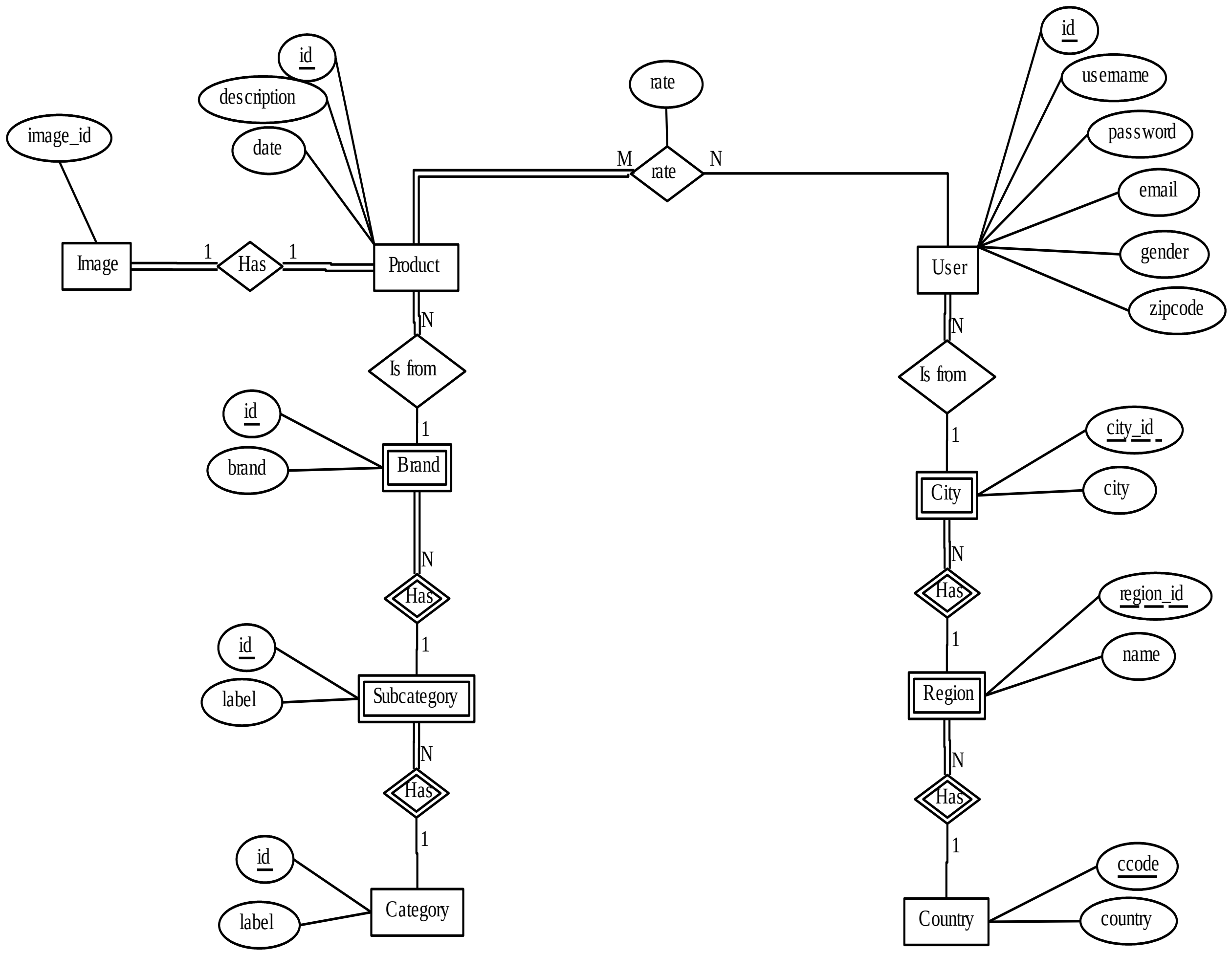
POST

The post link will redirect the user to the post form if he or she logged into the system. If he or she did not log in, the user will be directed to the login form. The post form includes the category of the product, subcategory, brand, and description, which are required. Also, an upload input to upload an image for the item is included. Additionally, a final required field is included to rate the item from 1-5. The category, subcategory, and brand are required to be able to classify items. Description is a way of giving a title or uniquely identifying the item. Rate is how the user liked the item.

DATABASE STRUCTURE:

The database was designed and implemented in MySQL. The database in the website contains 10 tables.

ER DIAGRAM



RELATIONAL DIAGRAM

User (id, username, password, email, gender, zipcode, ccode, region_id, city_id)

Product (id, description, date, category_id, subcategory_id, brand_id)

Country (ccode, country)

Region (id, ccode,name)

City (id, ccode, region_id, city)

Category (id, label)

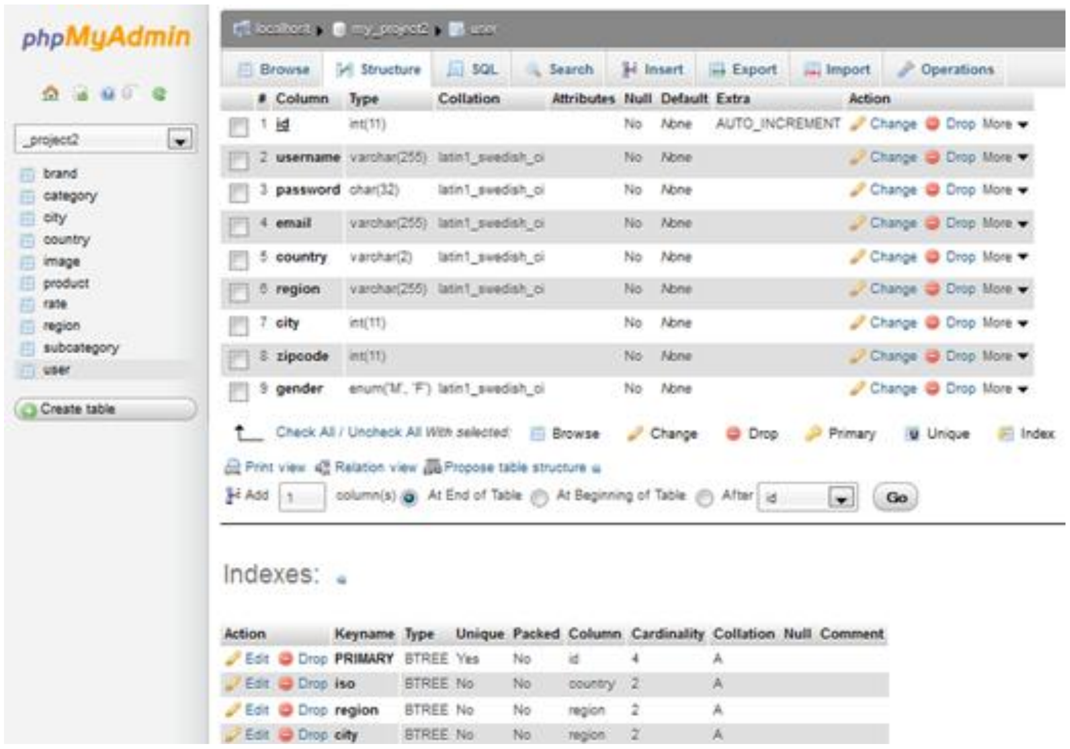
Subcategory (id, category_id,label)

Brand (id, category_id , subcategory_id , brand)

Image (product_id, image_id)

Rate (user_id, product_id, rate)

USER



#	Column	Type	Collation	Attributes	Null	Default	Extra	Action
1	id	int(11)			No	None	AUTO_INCREMENT	Change Drop More
2	username	varchar(255)	latin1_swedish_ci		No	None		Change Drop More
3	password	char(32)	latin1_swedish_ci		No	None		Change Drop More
4	email	varchar(255)	latin1_swedish_ci		No	None		Change Drop More
5	country	varchar(2)	latin1_swedish_ci		No	None		Change Drop More
6	region	varchar(255)	latin1_swedish_ci		No	None		Change Drop More
7	city	int(11)			No	None		Change Drop More
8	zipcode	int(11)			No	None		Change Drop More
9	gender	enum('M', 'F')	latin1_swedish_ci		No	None		Change Drop More

Check All / Uncheck All With selected: Browse Change Drop Primary Unique Index

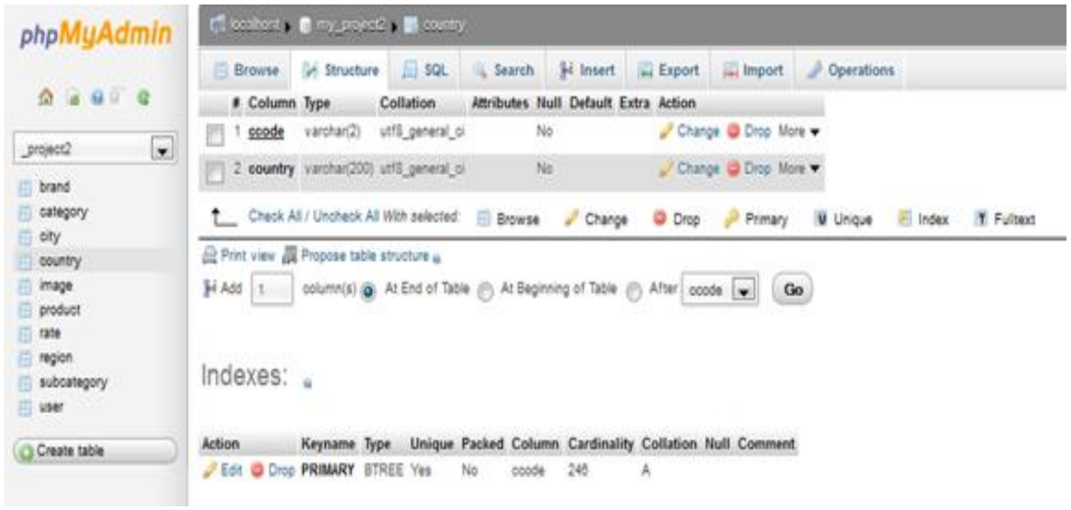
Print view Relation view Propose table structure

Add 1 column(s) [✓ At End of Table] At Beginning of Table After id Go

Indexes:

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Drop	PRIMARY	BTREE	Yes	No	id	4	A		
Edit Drop	iso	BTREE	No	No	country	2	A		
Edit Drop	region	BTREE	No	No	region	2	A		
Edit Drop	city	BTREE	No	No	region	2	A		

COUNTRY



#	Column	Type	Collation	Attributes	Null	Default	Extra	Action
1	ccode	varchar(2)	utf8_general_ci		No			Change Drop More
2	country	varchar(200)	utf8_general_ci		No			Change Drop More

Check All / Uncheck All *With selected:* Browse Change Drop Primary Unique Index Fulltext

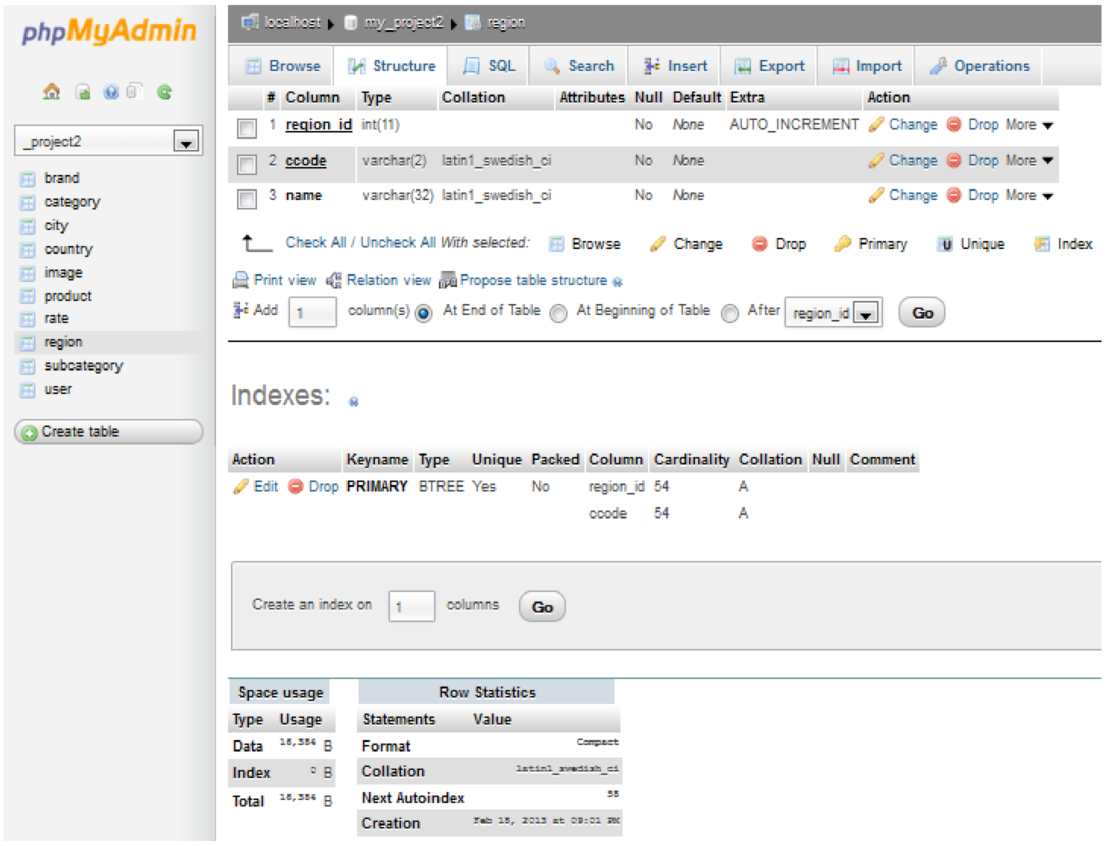
Print view Propose table structure

Add 1 column(s) [✓ At End of Table] At Beginning of Table After ccode Go

Indexes:

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Drop	PRIMARY	BTREE	Yes	No	ccode	248	A		

REGION



#	Column	Type	Collation	Attributes	Null	Default	Extra	Action
1	region_id	int(11)			No	*None*	AUTO_INCREMENT	Change Drop More
2	ccode	varchar(2)	latin1_swedish_ci		No	*None*		Change Drop More
3	name	varchar(32)	latin1_swedish_ci		No	*None*		Change Drop More

Check All / Uncheck All *With selected:* Browse Change Drop Primary Unique Index

Print view Relation view Propose table structure

Add 1 column(s) [✓ At End of Table] At Beginning of Table After region_id Go

Indexes:

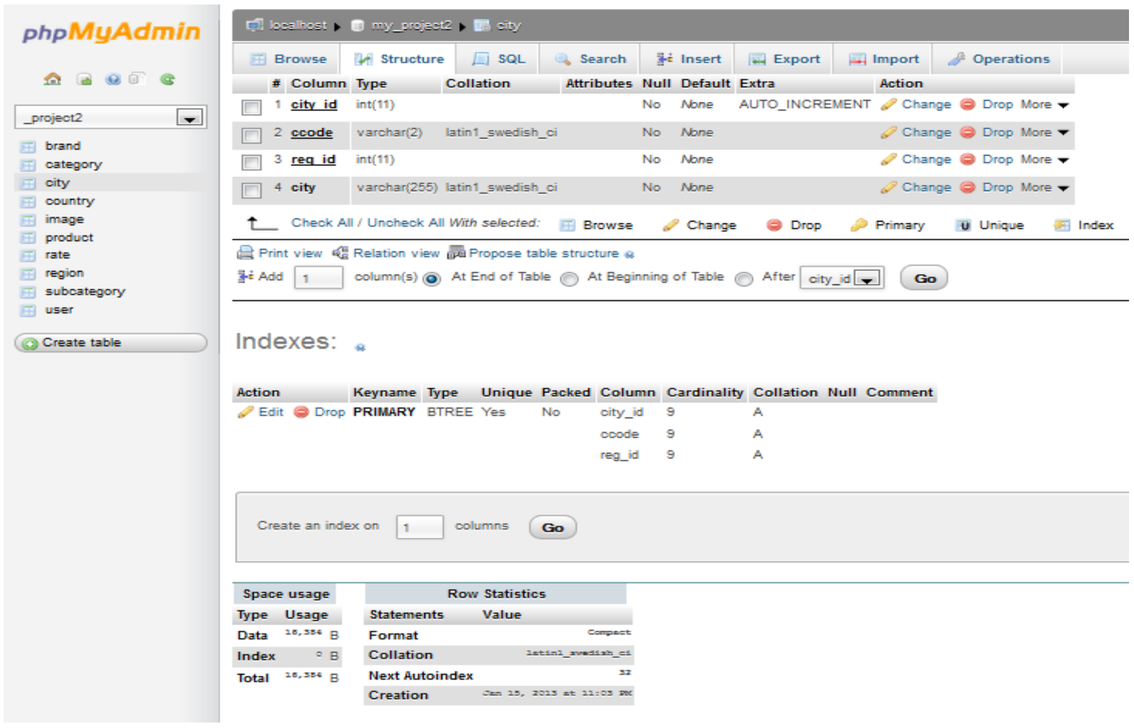
Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Drop	PRIMARY	BTREE	Yes	No	region_id	54	A		
					ccode	54	A		

Create an index on 1 columns Go

Space usage		
Type	Usage	
Data	16,384	B
Index	0	B
Total	16,384	B

Row Statistics	
Statements	Value
Format	Compact
Collation	latin1_swedish_ci
Next Autoindex	55
Creation	Feb 15, 2013 at 09:01 PM

CITY



	#	Column	Type	Collation	Attributes	Null	Default	Extra	Action
☐	1	city_id	int(11)			No	None	AUTO_INCREMENT	Change Drop More
☐	2	ccode	varchar(2)	latin1_swedish_ci		No	None		Change Drop More
☐	3	reg_id	int(11)			No	None		Change Drop More
☐	4	city	varchar(255)	latin1_swedish_ci		No	None		Change Drop More

Check All / Uncheck All *With selected:* Browse Change Drop Primary Unique Index

Print view Relation view Propose table structure

Add 1 column(s) [✓ At End of Table] At Beginning of Table After city_id Go

Indexes:

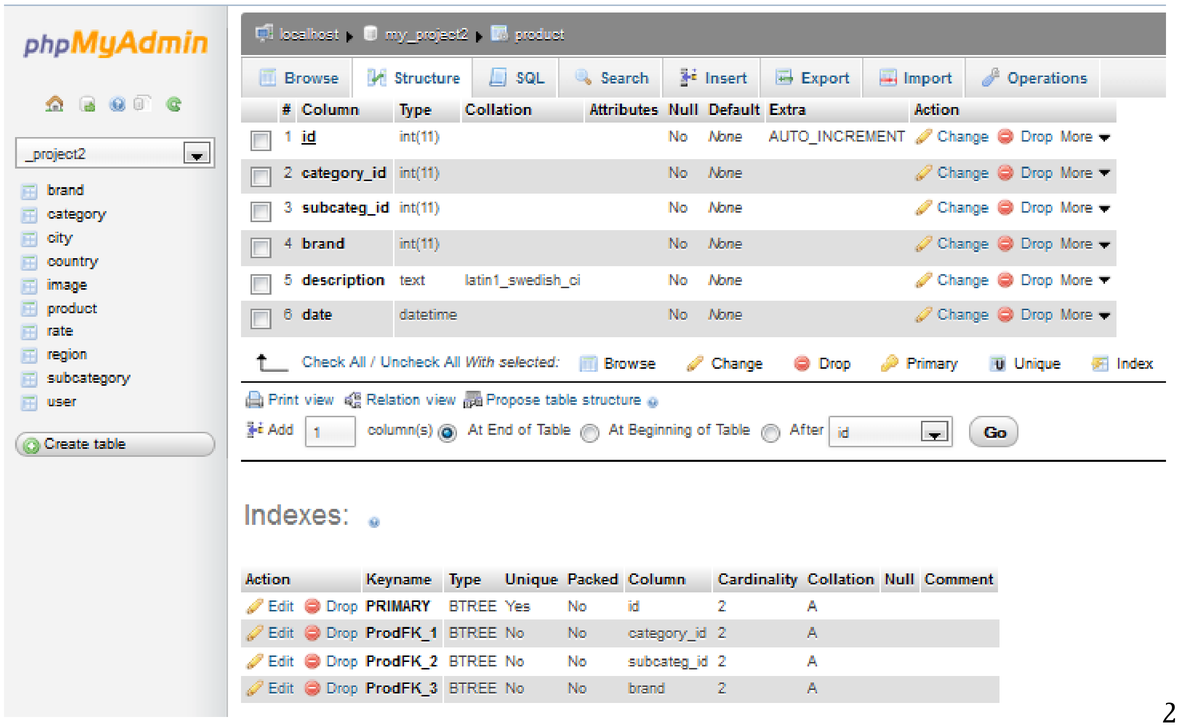
Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Drop	PRIMARY	BTREE	Yes	No	city_id	9	A		
					ccode	9	A		
					reg_id	9	A		

Create an index on 1 columns Go

Space usage		
Type	Usage	
Data	16,384	B
Index	0	B
Total	16,384	B

Row Statistics	
Statements	Value
Format	Compact
Collation	latin1_swedish_ci
Next Autoindex	32
Creation	Jan 15, 2013 at 11:03 PM

PRODUCT



	#	Column	Type	Collation	Attributes	Null	Default	Extra	Action
☐	1	id	int(11)			No	None	AUTO_INCREMENT	Change Drop More
☐	2	category_id	int(11)			No	None		Change Drop More
☐	3	subcateg_id	int(11)			No	None		Change Drop More
☐	4	brand	int(11)			No	None		Change Drop More
☐	5	description	text	latin1_swedish_ci		No	None		Change Drop More
☐	6	date	datetime			No	None		Change Drop More

Check All / Uncheck All *With selected:* Browse Change Drop Primary Unique Index

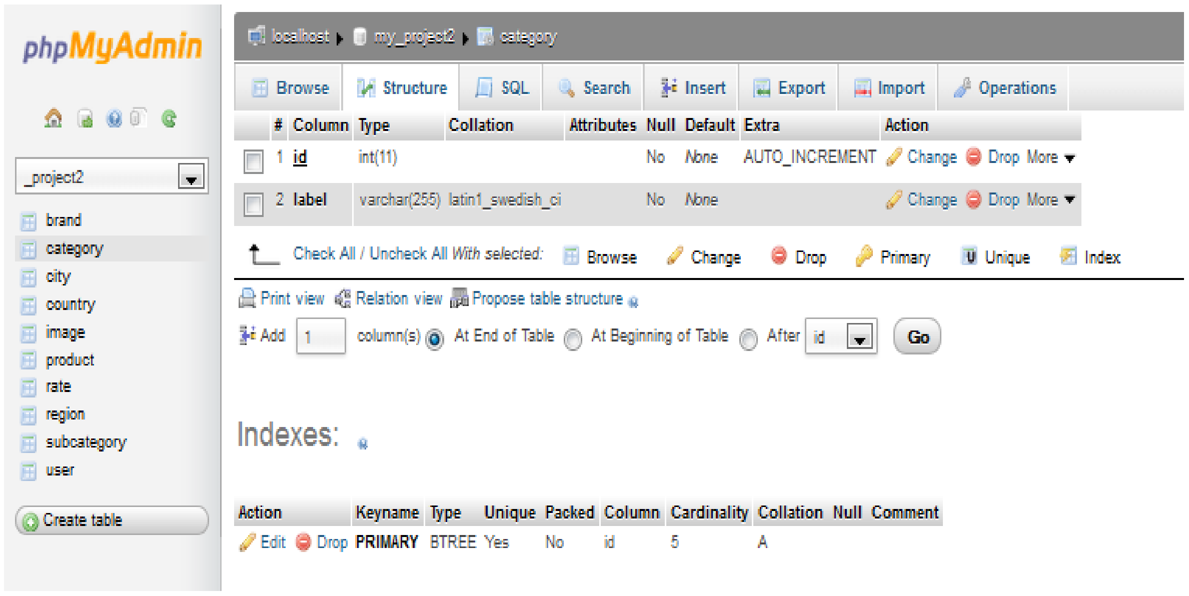
Print view Relation view Propose table structure

Add 1 column(s) [✓ At End of Table] At Beginning of Table After id Go

Indexes:

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Drop	PRIMARY	BTREE	Yes	No	id	2	A		
Edit Drop	ProdFK_1	BTREE	No	No	category_id	2	A		
Edit Drop	ProdFK_2	BTREE	No	No	subcateg_id	2	A		
Edit Drop	ProdFK_3	BTREE	No	No	brand	2	A		

CATEGORY

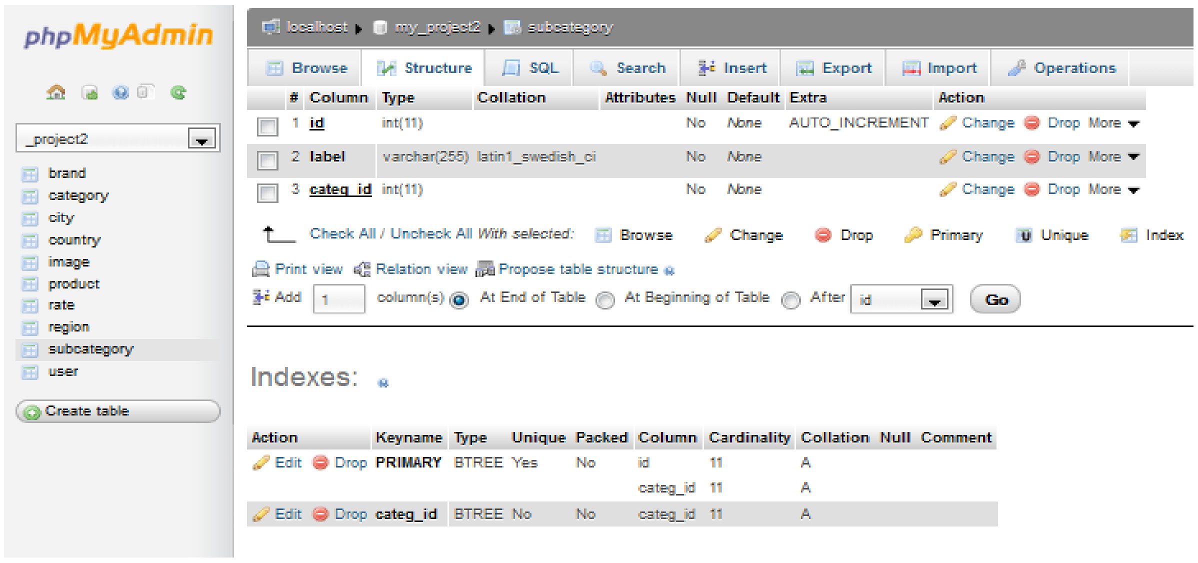


#	Column	Type	Collation	Attributes	Null	Default	Extra	Action
1	id	int(11)			No	None	AUTO_INCREMENT	Change Drop More
2	label	varchar(255)	latin1_swedish_ci		No	None		Change Drop More

Indexes:

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Drop	PRIMARY	BTREE	Yes	No	id	5	A		

SUBCATEGORY

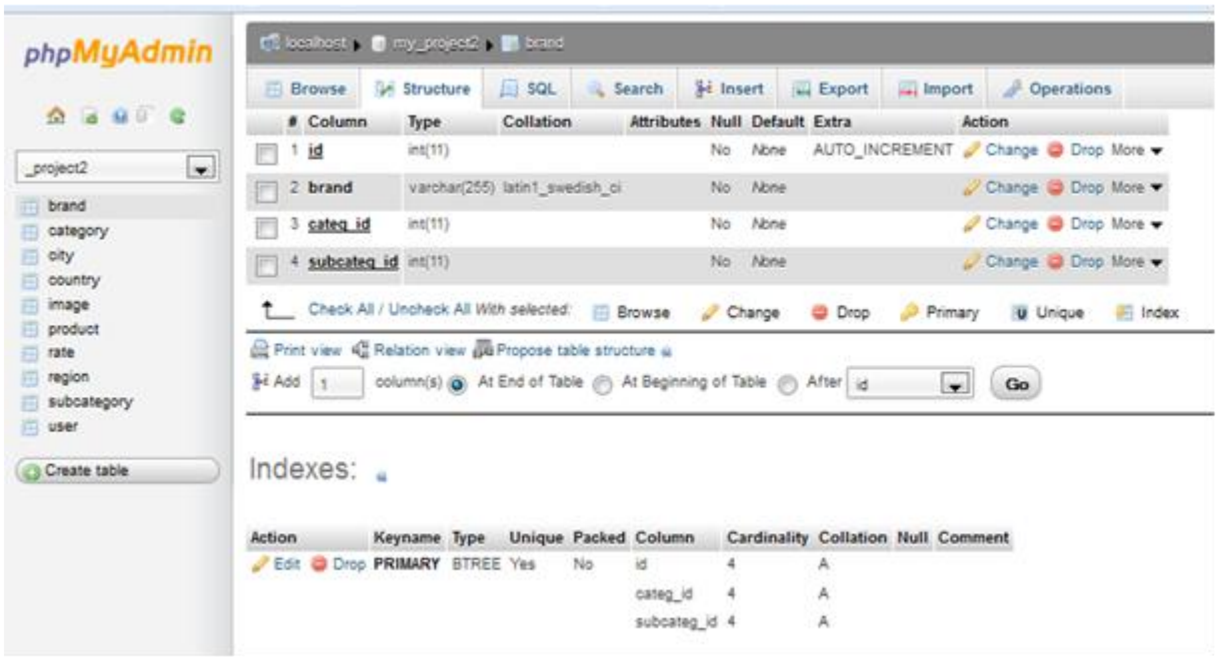


#	Column	Type	Collation	Attributes	Null	Default	Extra	Action
1	id	int(11)			No	None	AUTO_INCREMENT	Change Drop More
2	label	varchar(255)	latin1_swedish_ci		No	None		Change Drop More
3	categ_id	int(11)			No	None		Change Drop More

Indexes:

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Drop	PRIMARY	BTREE	Yes	No	id	11	A		
					categ_id	11	A		
Edit Drop	categ_id	BTREE	No	No	categ_id	11	A		

BRAND



#	Column	Type	Collation	Attributes	Null	Default	Extra	Action
1	id	int(11)			No	None	AUTO_INCREMENT	Change Drop More
2	brand	varchar(255)	latin1_swedish_ci		No	None		Change Drop More
3	categ_id	int(11)			No	None		Change Drop More
4	subcateg_id	int(11)			No	None		Change Drop More

Check All / Uncheck All *With selected:* Browse Change Drop Primary Unique Index

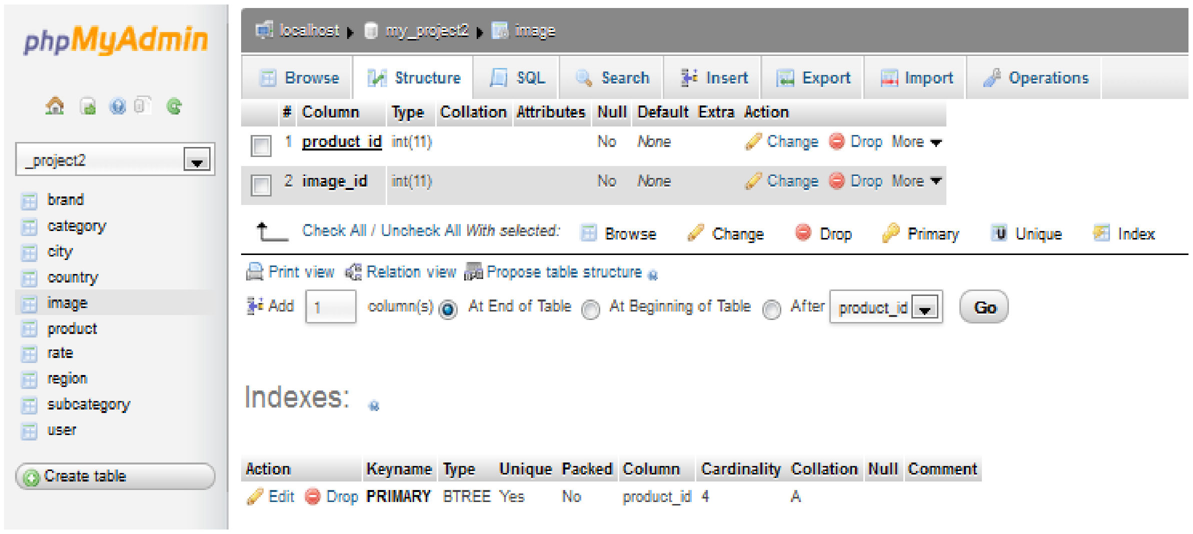
Print view Relation view Propose table structure

Add 1 column(s) [✓ At End of Table] At Beginning of Table After id Go

Indexes:

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Drop	PRIMARY	BTREE	Yes	No	id	4	A		
					categ_id	4	A		
					subcateg_id	4	A		

IMAGE



#	Column	Type	Collation	Attributes	Null	Default	Extra	Action
1	product_id	int(11)			No	None		Change Drop More
2	image_id	int(11)			No	None		Change Drop More

Check All / Uncheck All *With selected:* Browse Change Drop Primary Unique Index

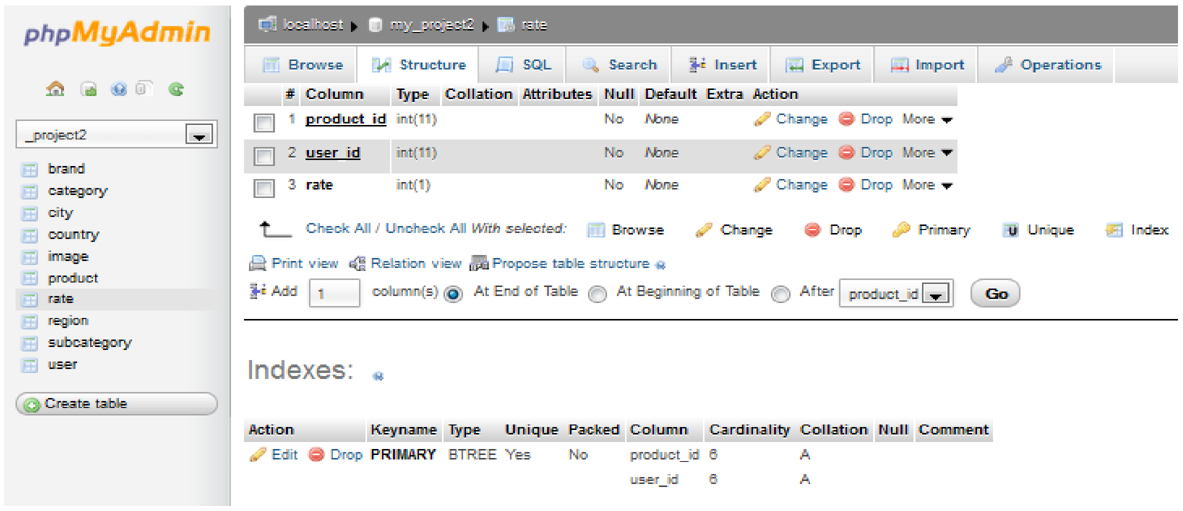
Print view Relation view Propose table structure

Add 1 column(s) [✓ At End of Table] At Beginning of Table After product_id Go

Indexes:

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Drop	PRIMARY	BTREE	Yes	No	product_id	4	A		

RATE



brand
category
city
country
image
product
rate
region
subcategory
user

#	Column	Type	Collation	Attributes	Null	Default	Extra	Action
1	product_id	int(11)			No	None		Change Drop More
2	user_id	int(11)			No	None		Change Drop More
3	rate	int(1)			No	None		Change Drop More

Check All / Uncheck All *With selected:* Browse Change Drop Primary Unique Index

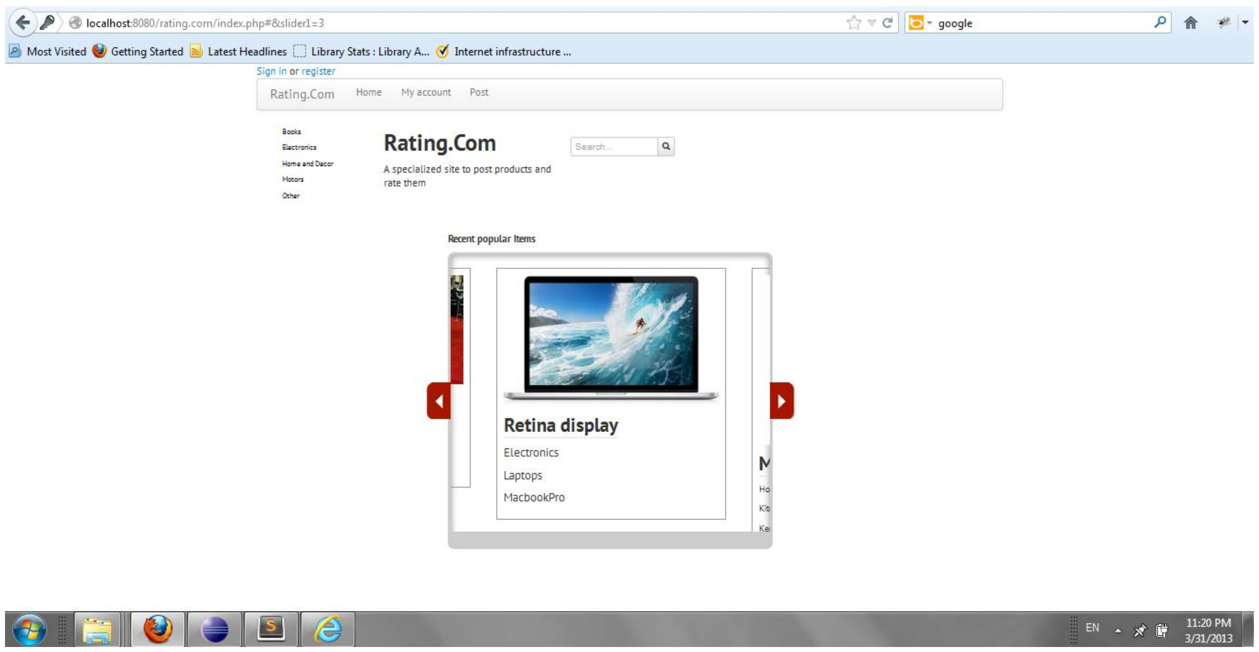
Print view Relation view Propose table structure

Add 1 column(s) [✓ At End of Table] At Beginning of Table After product_id Go

Indexes:

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Drop	PRIMARY	BTREE	Yes	No	product_id	6	A		
					user_id	6	A		

USER INTERFACE



Rating.Com

A specialized site to post products and rate them

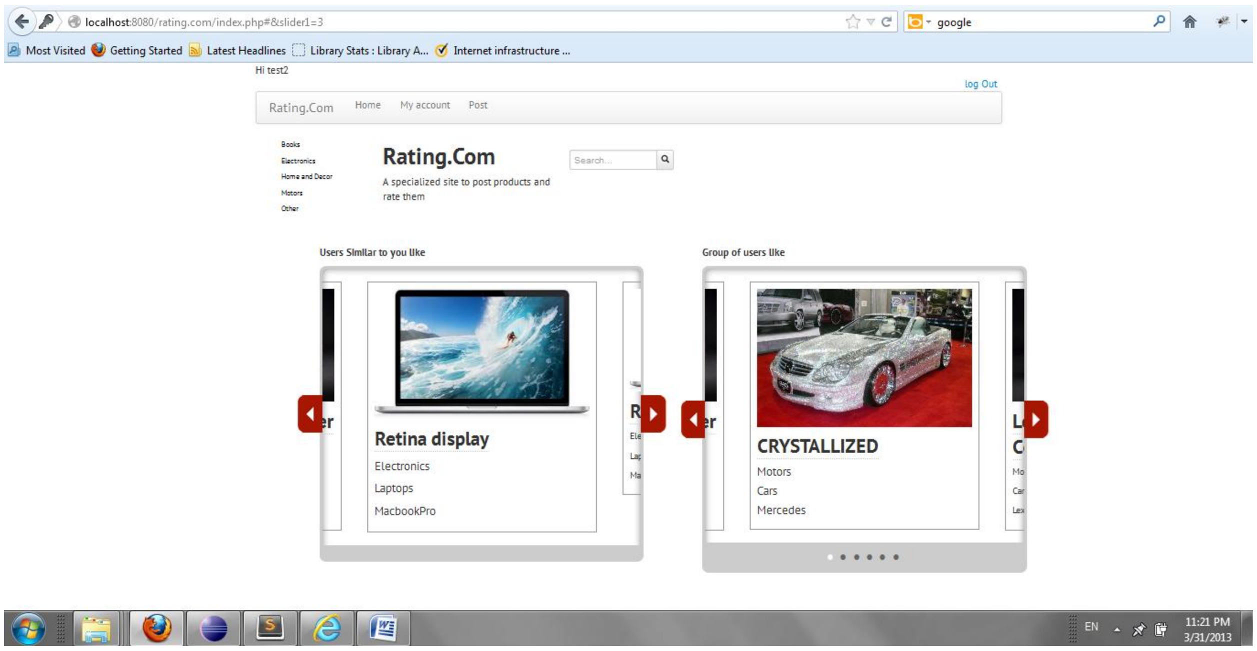
Recent popular Items

Retina display

Electronics

Laptops

MacbookPro



Rating.Com

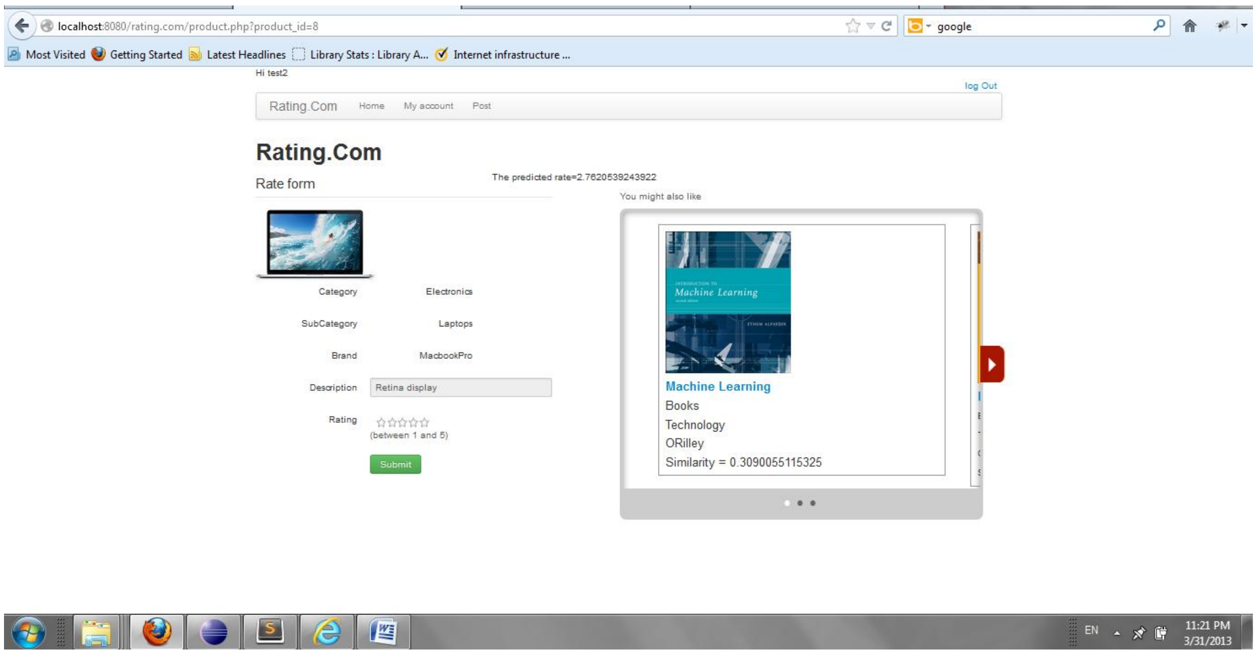
A specialized site to post products and rate them

Users Similar to you like

Retina display
Electronics
Laptops
MacbookPro

Group of users like

CRYSTALLIZED
Motors
Cars
Mercedes



Rating.Com

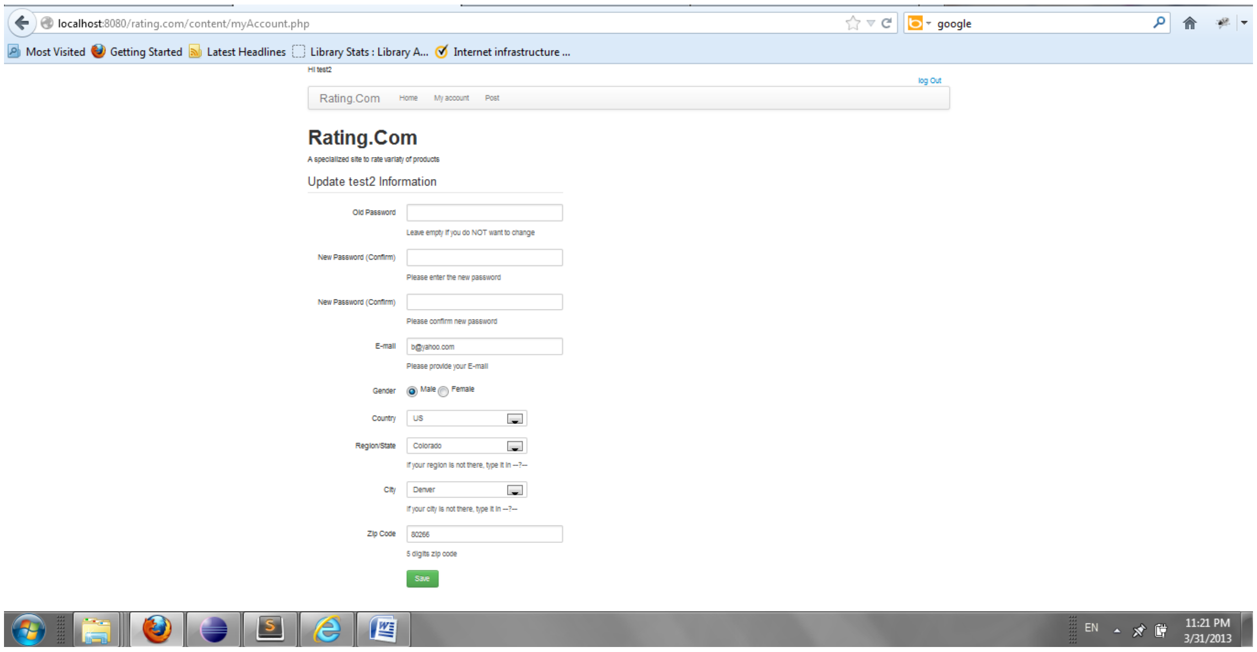
Rate form

The predicted rate=2.7620539243922

Category	Electronics
SubCategory	Laptops
Brand	MacbookPro
Description	Retina display
Rating	☆☆☆☆☆ (between 1 and 5)

You might also like

Machine Learning
Books
Technology
ORilley
Similarity = 0.3090055115325


localhost:8080/rating.com/content/myAccount.php
google
Most Visited
Getting Started
Latest Headlines
Library Stats : Library A...
Internet infrastructure ...
EN
11:21 PM
3/31/2013

Hi test2

log Out

Rating.Com Home My account Post

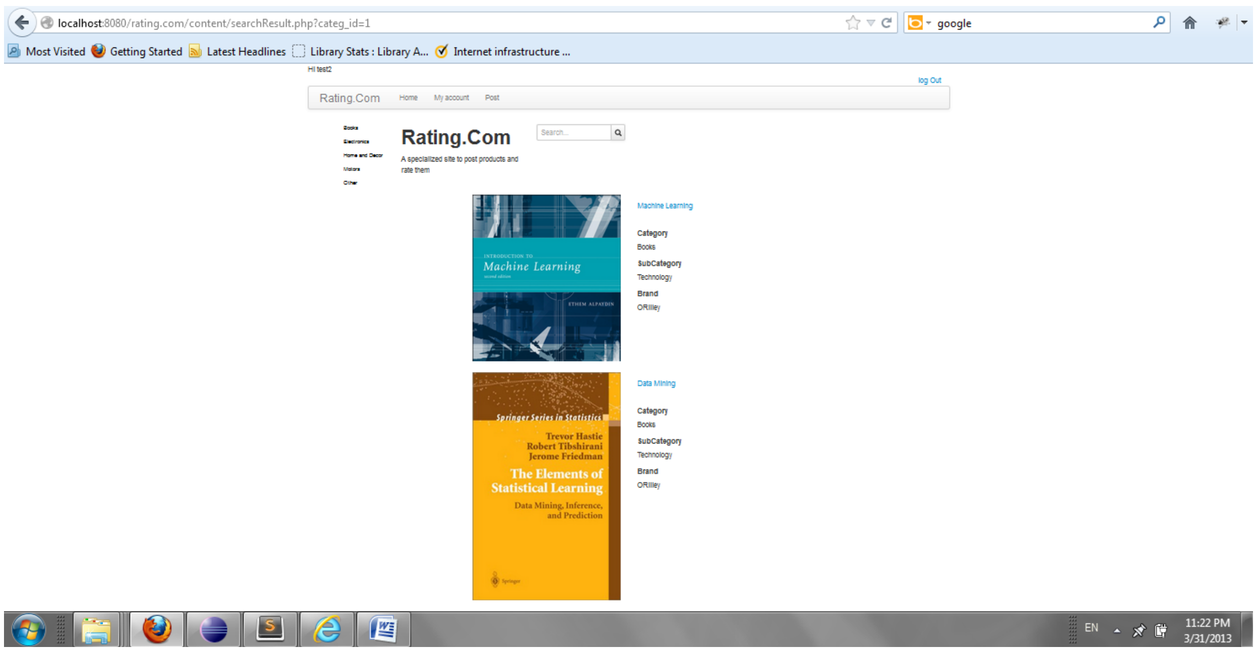
Rating.Com

A specialized site to rate variaty of products

Update test2 Information

Old Password	
	Leave empty if you do NOT want to change
New Password (Confirm)	
	Please enter the new password
New Password (Confirm)	
	Please confirm new password
E-mail	b@yahoo.com
	Please provide your E-mail
Gender	[✓ Male] Female
Country	US
Region/State	Colorado
	If your region is not there, type it in --?--
City	Denver
	If your city is not there, type it in --?--
Zip Code	80266
	5 digits zip code

Save


localhost:8080/rating.com/content/searchResult.php?categ_id=1
google
Most Visited
Getting Started
Latest Headlines
Library Stats : Library A...
Internet infrastructure ...
Search...
INTRODUCTION TO
Machine Learning
second edition
ETHEM ALPAYDIN
Springer Series in Statistics
Trevor Hastie
Robert Tibshirani
Jerome Friedman
The Elements of Statistical Learning
Data Mining, Inference, and Prediction
Springer
EN
11:22 PM
3/31/2013

Hi test2

log Out

Rating.Com Home My account Post

Books
Electronics
Home and Decor
Motors
Other

Rating.Com

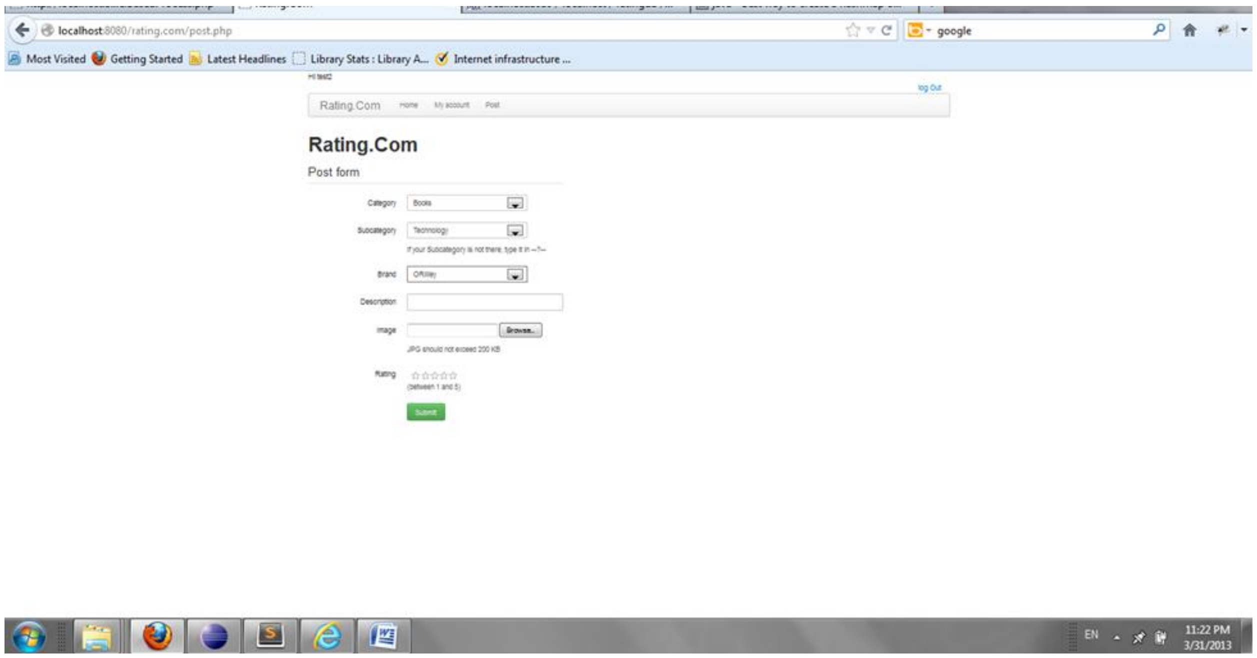
A specialized site to post products and rate them

Machine Learning

Category
Books
subCategory
Technology
Brand
ORilley

Data Mining

Category
Books
subCategory
Technology
Brand
ORilley


Hi test2

log Out

Rating.Com Home My account Post

Rating.Com

Post form

Category	Books
Subcategory	Technology
	If your Subcategory is not there, type it in --?--
Brand	OReilley
Description	
Image	Browse..
	JPG should not exceed 200 KB
Rating	☆☆☆☆☆ (between 1 and 5)

Submit

IMPLEMENTATION

HOW IT WORKS

The website provides a simple interface that collects ratings for certain products. Then, using Collaborative filtering algorithms, we suggest other products based on those ratings and allow the user to rate them. The Home page will display moving boxes with products that have been added recently and highly rated. If the user has logged into the system, the home page will display two moving boxes, one with products that are predicted to be highly rated using User-based Collaborative Filtering, another box will show products that are predicted to be highly rated based on Model-based Collaborative filtering.

Also, the website has search functionality where users can search for products. If a particular product is selected, the system will display a box with similar products based on Item-based Collaborative Filtering. If the user has signed in, then the system will display a predicted rating using the Item-based Collaborative Filtering next to the item that has been selected.

TECHNOLOGIES/TOOLS WERE USED

Bootstrap framework for CSS and JavaScript was used for designing the front end. Also, the interface was written in HTML. PHP was the language used for coding the server side. The database was implemented in MySQL. For local design and implementation I used WAMP. I had basic knowledge of web development like HTML and PHP that I have used in class CIS 658: Web Architectures. For evaluation and testing I used Java and eclipse.

As I was working with the project I have learned CSS with a great help from Bootstrap framework. Since the website has limited data we used an external dataset “Movie Lens” to test and evaluate the different Collaborative Filtering algorithms. Movie Lens data set with 100K (http://www.grouplens.org/node/73) was used for testing and evaluating the three algorithms that I have used in the website. User-based, Item-based, and Model-based Collaborative Filtering have been evaluated regarding error rate and execution time.

Also, I have learned how to apply the Machine learning algorithm and how to use some alternatives to speed up the processing time. For example, to calculate the error rate in the previous Collaborative Filtering algorithms I had to process the files in stages using Java due to the amount of data. In User-based Collaborative Filtering, first I calculated the average for each user and wrote that to a file. The next stage I calculated the weight for each user and again wrote it to a file. The last stage I was reading the information from both files to calculate the predicted rate and error rate.

CHALLENGES

ITEM FEATURES

We had to limit the features of items due to two reasons. One, there is a tradeoff between performance and quality of collaborative filtering. If we are going to use large amounts of information to predict a new preference, we will have to search large groups of neighbors, which will slow down system performance. However, with criteria that matches users (long user row information), quality increases. Two, we wanted to focus on the algorithm more than designing the interface. The project aims to apply machine learning on a problem and prove a concept.

As a consequence of limited features, we faced duplicate entry problems. Since we needed few features to apply the algorithm, the ability to detect duplicates was limited. In this project, we look for similar items regarding category, subcategory, brand, and similar phrase of description. For instance, I used the 'LIKE' in MySQL to detect similar descriptions. Adding more features, like every product's details, e.g., details about books, will drift the system from its main goal.

IMAGE STORAGE

When we decided to offer the option of adding images to the product, we had two options: store the image in the database or just its identification, and let the server handle the fetching and storing of images. Both have advantages and disadvantages. The first option has the advantage of the ease of packing the whole system in the database while reducing database performance. The second option, which is the one that we followed, provides better performance while the server should handle retrieving and storing images.

NEW USER AND NEW ITEM PROBLEM

The Item-to-item approach in Amazon uses the same algorithm that we used in Item-based Collaborative Filtering except that it executes the process offline and stores the result to be compared online. One of the challenges that commercial websites face is the new user or new item problem. I didn't face that problem with an existing item that has no rating in my website because every item posted has to be rated. Yet, for a new user I just suggest the list of highest rated items since I have no profile for this user to compare with others. In the same line, Candillier explains how they used the mean or majority of ratings for user or item to overcome this problem (Candillier et al., 2007).

The problem with traditional collaborative filtering was to find accurate recommendations with limited user data. Amazon states that their item-to-item approach proves superior in this issue, where recommendations were accurate with limited or amounts of data. However, the trend today is to base recommendation not only on website resource but to use social media to refine user preference. Thus, companies now try to use the API of Facebook, Twitter, and LinkedIn to create user profiles.

RESULTS, EVALUATION, AND REFLECTION

To evaluate the algorithms, I used the Movie Lens data set and wrote the algorithms using Java and Eclipse IDE. In particular, I applied User-based, Item-based, and Model-based Collaborative Filtering and calculated the error rate for each one using Mean Absolute Error Rate (MAE) and Root Mean Squared Error using the following formulas:

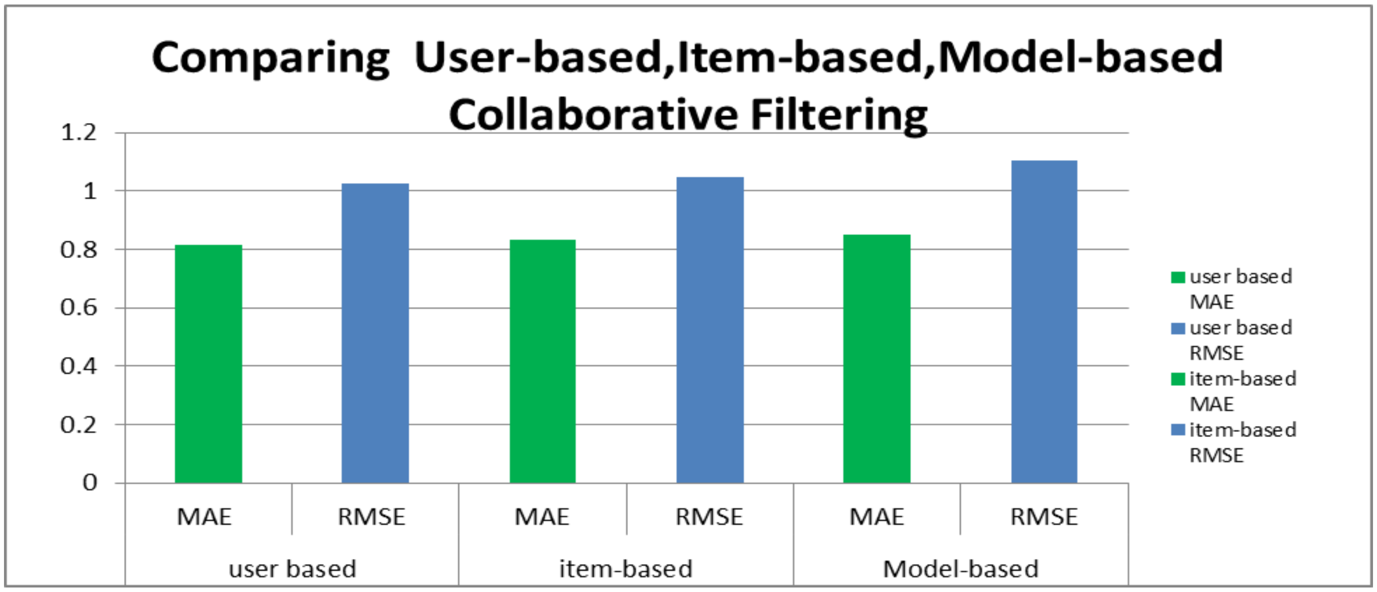
$$MAE = \frac{1}{|T|} \sum_{(u,i,r)\in T} |p_{ui} - r|$$

$$RMSE = \sqrt{\frac{1}{|T|} \sum_{(u,i,r)\in T} (p_{ui} - r)^2}$$

(Candillier et al., 2007)

Where T represent the number of test cases, pui the predicted rate by the algorithm, and r is the actual rate.

User-based		**Item-based**		**Model-based**	
MAE	RMSE	MAE	RMSE	MAE	RMSE
0.816429099	1.022968896	0.834149128	1.04480249	0.85019701	1.10512221



It’s obvious that the accuracy of the User-based approach is the greatest. Additionally, the complexity in computation and processing time for Item-based and Model-based are higher than User-based.

CONCLUSIONS AND FUTURE WORK

The huge amount of data and the computational power needed to bring accurate results should be considered in applying an online or even offline recommender system. The algorithms I used were popular once, however, there are other recommender engines like Hunch which is used in eBay, one of the biggest leaders in online commerce. eBay was looking to compete with Amazon by acquiring Hunch engine due to its house built Item-based recommender system that does not compete very well. Chis Dixon, the co-founders of Hunch and entrepreneur in the field of data mining and machine learning, refers to the value of using a history of data that can be linked to social commerce (Kim, 2013).

For example, Hunch uses Facebook accounts and twitter to refine what users like and build a graph that links people to objects. My6sense, a newsreader app, uses the same idea of Hunch where different API’s like Twitter and other social media are integrated for recommendation purposes. These types of engines monitor the links you clicked, time spent reading, and whether the link was passed or shared. In addition, Forage is a Hunch API that recommends YouTube videos based on user’s tweets (Let’s get personal, 2013).

Another example of how Hunch might be used in predicting gifts is to see what your friends’ like and interests and then predict the ultimate gift. In fact, Gifts.com, one of the popular websites that suggest gifts by a series of questions and answers, has partnered with Hunch to combine expertise with technology (Cafferty, 2010). Cafferty describes the new mechanisms in the following lines:

- Instant gift recommendations for the shopper's Facebook friends based on their likes, interests and profile information.
- Dynamic updates to the list of gift recommendations as the shopper fine tunes the suggestions by answering "yes" or "no" to each gift.
- The ability to increase the "confidence level" of Hunch's taste profile algorithm by answering conventional or more entertaining questions about the recipient like, "Alien Abductions: Does your recipient think they are real or fake?"

(Cafferty, 2010)

The limitation considered for Amazon and other customized recommender systems is that they are not evolving. In other words, the web is rich with information that can be used to build more personalized profiles. Amazon’s method relies on its history and does not exploit other resources like Twitter where users may share information or video about certain products. In addition, recommender engines now try not to rely on current matches and look for older products that may match user’s preference. To explain, My6sense app looks for articles from weeks ago and suggests them to the user (Let’s get personal, 2013).

The details of Hunch implementation were not discussed in detail. The taste graph is built as a user clicks or linked to people or objects. However, how the system builds the graph or parses the collected information to predict a new preference is not explained. The overview of how the system works mainly uses AI and some of machine learning to create a taste graph or taste profile.

The different characteristics of domains may also restrict the type of recommender system viability. A recommender system for the music industry differs in how neighbors are compared and clustered than other domains like books or movies. Other Collaborative Filtering approaches that we did not expand on here include Content-based filtering, Behavioral targeting technique, and Matchbox, which is a Bayesian recommender engine that uses collaborative and feature-based methods for predictions.

People change and their habits also change. This is the biggest challenge that recommender systems face. Another issue arises if the user refuses to connect his or her account to any of social media. Would Hunch will be useful in this situation? Another concern is whether a reliable stand-alone method, which can be scaled to use social media, exists. If this tool exists, security issues will be incurred and need to be dealt with.

BIBLIOGRAPHY

Cafferty, L. (2010, November 22). Gifts.com & Hunch Partner to Build the Ultimate Gift Recommendation Engine; New Feature Provides Personalized Recommendations for Facebook Friends . In *PR Newswire*. Retrieved March 23, 2013, from http://www.prnewswire.com/news-releases/giftscom--hunch-partner-to-build-the-ultimate-gift-recommendation-engine-new-feature-provides-personalized-recommendations-for-facebook-friends-109929159.html

Candillier, L., Meyer, F., & Boull'e Marc. (2007). Comparing state-of-the-art collaborative filtering systems. Proceedings of the 5th International Conference on Machine Learning and Data Mining in Pattern Recognition, Leipzig, Germany. 548-562. doi: 10.1007/978-3-540-73499-4_41

Kim, Ryan. Why eBay is buying recommendation engine Hunch . Retrieved March 21, 2013, from http://gigaom.com/2011/11/21/why-ebay-is-buying-recommendation-engine-hunch/

Linden, G.; Smith, B.; York, J.; , "Amazon.com recommendations: item-to-item collaborative filtering," Internet Computing, IEEE , vol.7, no.1, pp. 76- 80, Jan/Feb 2003
doi: 10.1109/MIC.2003.1167344
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1167344&isnumber=26323

Let's get personal: How recommendation engines are making consumption more dynamic. Retrieved March 21, 2013, from http://agiledudes.com/author/ron/

Tawakol, O. (2012, September 7). More data beats better algorithms — Or does it?. In *All Things D*. Retrieved January 15, 2013, from http://allthingsd.com/20120907/more-data-beats-better-algorithms-or-does-it/

Mahout (2011). *What is Apache Mahout*. Retrieved April 6, 2013, from http://mahout.apache.org/

Dataset:

mludwig /Rightsholder. (2011). MovieLens Data Sets .Location GroupLens Research Project at the University of Minnesota. URL: http://www.grouplens.org/node/73

GLOSSARY

Terms:

Item-based Collaborative Filtering

An algorithm to predict an item based on similarity between items.

User-based Collaborative Filtering

An algorithm to predict an item based on similarity between users.

Hybrid Collaborative Filtering

An algorithm to predict an item based on similarity between users and items.

Cluster

Unsupervised method to classify objects.

K-means

An algorithm for clustering which uses Euclidean distance or another distance measurement to group objects.

WAMP (Windows Apache MySQL PHP)

Software that contains Apache, MySQL, and PHP.

API (Application Programming Interface)

An interface that allows software components to communicate with each other.

MAE (Mean Absolute Error Rate)

A measurement to determine the accuracy of estimation compared to the actual value.

RMSE (Root Mean Squared Error)

A measurement to determine the accuracy of estimation compared to the actual value..