Recommendation System

Ceng History X

October 9, 2013

Team Members:

1) Asena Ok, 1746296, asenaok@gmail.com
2) Ayşe Aybüke Taşdirek, 1746353, aybuketasdirek@gmail.com
3) Birant Altınel, 1745744, birantaltinel@gmail.com
4) Hacer Nihal Tarkan, 1744291, tarkan.nihal@gmail.com

1. Problem Definition and Background Information

The product that we want to create is Recommendation System. Recommender systems are systems that make recommendations for users on the basis of data the users have entered into the system. The more data a user has provided, the more accurate such systems can be. In addition, data submitted by individual users helps to improve the system overall, by generating information that can be used to make recommendations for other users. Recommender systems are commonly seen on sites like movie and television review sites and those with large inventories of retail items that would be functionally impossible to browse by looking at every item. [19]

Everywhere around the world where people use computers, smart phones and smart TVs and so on, there are many e-commerce or related websites, software, applications that a recommendation system can be very useful and should be available at. Other than countless e-commerce websites, mobile platforms for smart phones and tablets are also perfect environments since they have application stores, music stores and variety of other software that displays and sells digital content. Another idea would be satellite television providers such as Digiturk and D-Smart in Turkey. These providers include services for users to view and purchase movies, series and other visual content that would improve with a well integrated recommendation system.

The corresponding environments that this project will be integrated to are supposed to and probably will have a large number and variety of users in which many of them need such system to find whatever it is they need. The users may or may not be aware of their need for a recommendation feature on the software or website they are using, but such features can increase efficiency and save time to users, while they are looking in a place where there is a large number and variety of content which causes them to waste a lot of time to find what they need. From another point of view, there might be users that don’t exactly know what they are looking for and such situations can also make a useful solution out of this project.

People who use social network sites and the people who do shopping on the web will be affected by this system. This is a huge number if we think that only Facebook has 250 million users.

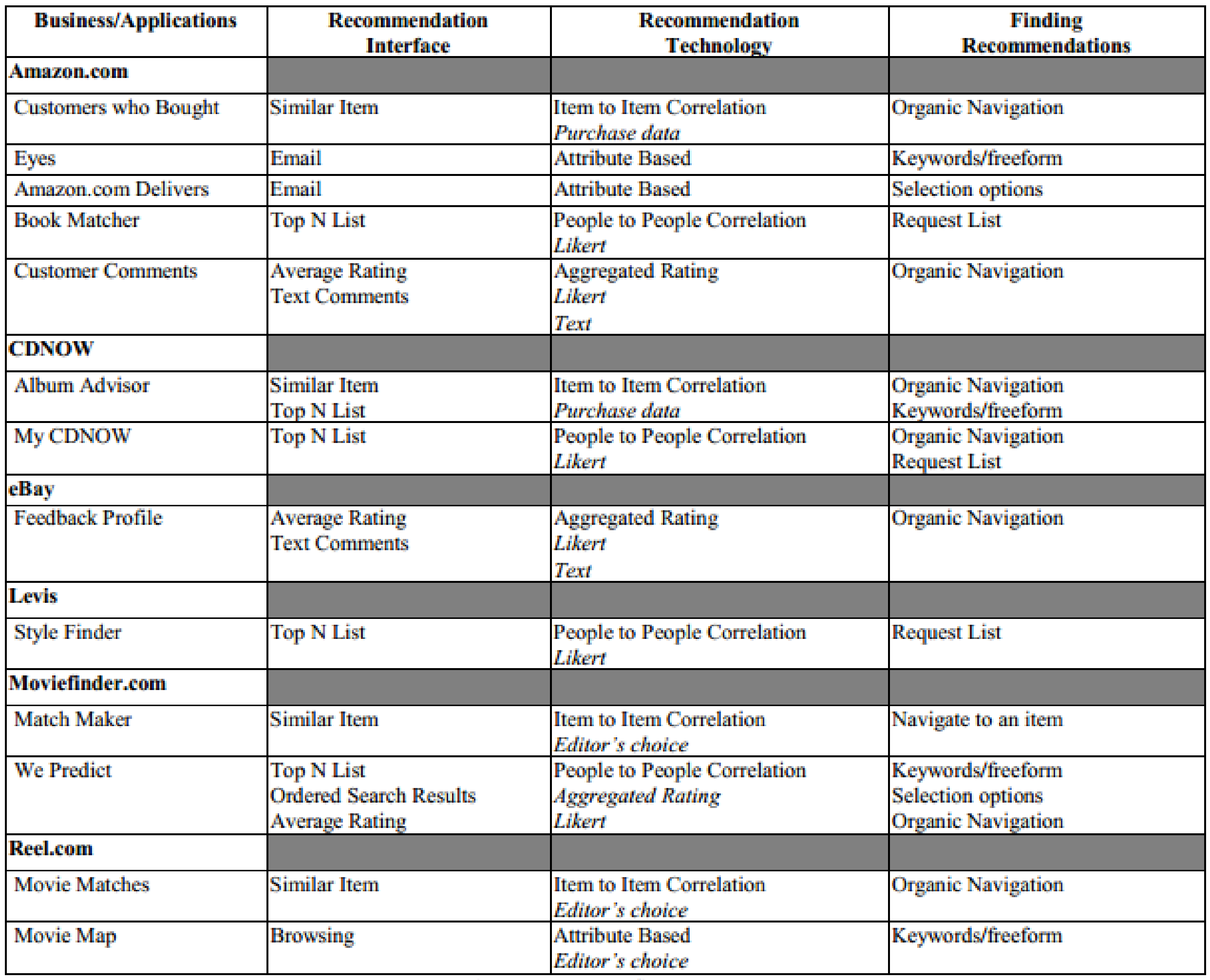
There are many variations as a solution on different websites and systems (Amazon, Netflix etc.), but they are usually specific solutions created only for the corresponding product. Many e-commerce systems have their own recommendation systems. However these only provide basic level recommendations. There are still many fields and platforms in the need of good recommendation systems. On the developing level this problem involves data manipulation. So database designers are trying to solve this problem. Because, the actual problem is not finding the items to recommend, also giving the relevant result in acceptable time is very important.

The first system using social filtering created by Malone, Grant, Turbak, Brobst, and Cohen (1987) more than 20 years ago. Today recommender systems are an accepted technology used by market leaders in several industries. Recommender systems use different filterings. These filterings are classified generally in two approaches: collaborative filtering and content-based filtering. Recommender algorithms are being studied. Several projects were initiated to implement recommender algorithms as you can see in Table 1 [20].

Software	Description	Language	URL
Apache Mahout	Machine learning library includes collaborative filtering	Java	http://mahout.apache.org/
Cofi	Collaborative filtering library	Java	http://www.nongnu.org/cofi/
Crab	Components to create recommender systems	Python	https://github.com/muricoca/crab
easyrec	Recommender for Web pages	Java	http://easyrec.org/
LensKit	Collaborative filtering algorithms from GroupLens Research	Java	http://lenskit.grouplens.org/
MyMediaLite	Recommender system algorithms.	C#/Mono	http://mloss.org/software/view/282/
SVDFeature	Toolkit for feature-based matrix factorization	C++	http://mloss.org/software/view/333/
Vogoo PHP LIB	Collaborative filtering engine for personalizing web sites	PHP	http://sourceforge.net/projects/vogoo/

Table 1: Recommender System Software

Yes, there is. For example, Amazon.com, CDNOW, eBay, Levis, Moviefinder.com, Reel.com, Netflix, Pandora Radio, Last.fm, Youtube, Yahoo, Tripadvisor, and IMDb etc. There are some detailed information about some of these products in Table 2 [18].



Business/Applications	Recommendation Interface	Recommendation Technology	Finding Recommendations
Amazon.com			
Customers who Bought	Similar Item	Item to Item Correlation *Purchase data*	Organic Navigation
Eyes	Email	Attribute Based	Keywords/freeform
Amazon.com Delivers	Email	Attribute Based	Selection options
Book Matcher	Top N List	People to People Correlation *Likert*	Request List
Customer Comments	Average Rating Text Comments	Aggregated Rating *Likert* *Text*	Organic Navigation
CDNOW			
Album Advisor	Similar Item Top N List	Item to Item Correlation *Purchase data*	Organic Navigation Keywords/freeform
My CDNOW	Top N List	People to People Correlation *Likert*	Organic Navigation Request List
eBay			
Feedback Profile	Average Rating Text Comments	Aggregated Rating *Likert* *Text*	Organic Navigation
Levis			
Style Finder	Top N List	People to People Correlation *Likert*	Request List
Moviefinder.com			
Match Maker	Similar Item	Item to Item Correlation *Editor's choice*	Navigate to an item
We Predict	Top N List Ordered Search Results Average Rating	People to People Correlation *Aggregated Rating* *Likert*	Keywords/freeform Selection options Organic Navigation
Reel.com			
Movie Matches	Similar Item	Item to Item Correlation *Editor's choice*	Organic Navigation
Movie Map	Browsing	Attribute Based *Editor's choice*	Keywords/freeform

Table 2: Recommender System Examples

2. Significance of the Problem and Motivation

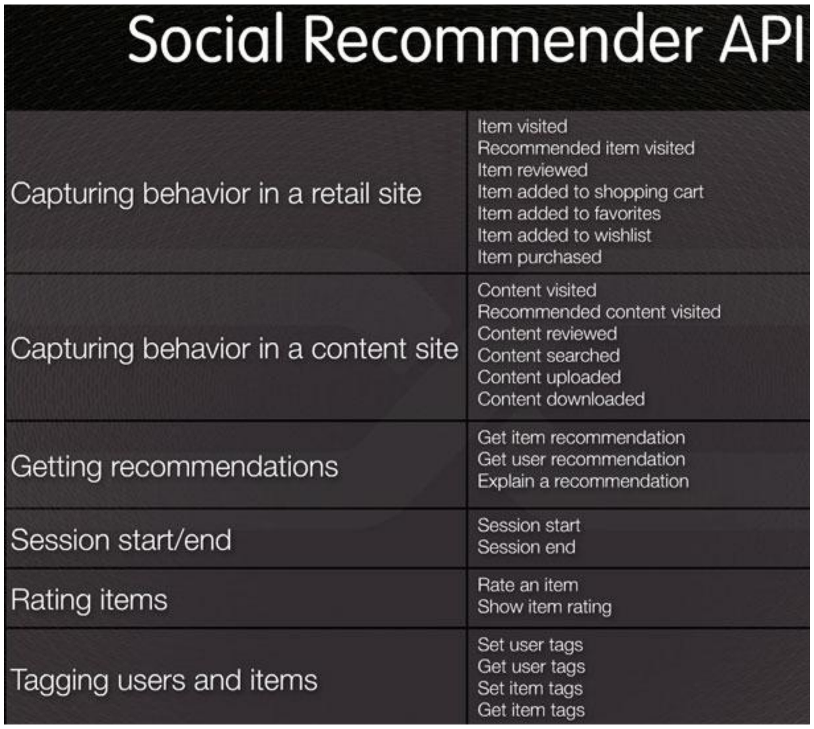
The main issues would be both huge, dense data sizes and sparse data sizes. There also may be issues about similarity calculations and data integration. To make the picture more clear, if we are to specify data integration process as an example, to integrate data from different sources requires combining heterogeneous data sources under a single query interface which raises an important issue.

The problem about huge and dense data is that it may lead to the need of more complex calculations with inefficient speed. Therefore we need to work heavily on different paradigms, algorithms and databases to create an efficient solution even for very large data sets. On the other hand, density of data is an advantage in terms of accuracy of the system. As stated above, if density can be run efficiently on the side of development, it would be improve the quality of the system.

Sparsity of data is also a major issue, because no matter how well the algorithms are, if there is no data to process, this system will be of no use. Therefore we need to create solutions to also deal with these situations. Cold start may cause sparsity. If there is not enough data for a new user in the system, it will affect the recommender negatively.

We considered and thought about all project ideas in detail, but decided on this one in conclusion. Considering we are a group of 4 people, and all of us have our own plans for future, we had to choose a project which will provide benefits to all group members. For example database management is a topic that we all interested in. In courses we have taken until now, unfortunately we could not get a chance to observe database management practically. Now here, to solve this problem we have to learn more techniques to manage the data that is kept as we learnt in the class, I mean relational database. Or we also have another chance to see or learn the other databases such as NOSQL, graph databases and use their query languages. This project includes mostly the points those we want to specialize in. Also, since some of us have industry-oriented plans and some of us have academic-oriented plans for the future, we have chosen the recommendation system project. It requires innovative thinking, making a lot of analysis, creating efficient solutions to any possible situation, possibly using Apache Mahout, maybe using map-reduce techniques on a cluster, dealing with NOSQL and neo4j databases, data integration, using hybrid models and so on, which provides many opportunities to all group members in the means of learning, experiencing, developing. We think this also is an appropriate project to enter national/international competitions etc.

This problem does not and will not have specific solutions that can be considered as satisfactory because of several major reasons. Perhaps the biggest issue facing recommender systems is that they need a lot of data to effectively make recommendations, which is only available for huge companies with a large chunk of consumer data. Another significant problem is changing trends, which result in changing data, and this situation eventually causes recommender systems to compute old information and reflect false results. In addition to changing trends, another important point would be about changing user preferences. It would be extremely misleading to assume that a user will always be in search of the content for themselves, and according to their own preferences. This assumption would result in failure in predicting the recommendations when, let’s say, the user is looking for items to purchase as a present to another individual. The other issue is that the thing people like is very ambiguous. For example, the type of movie that people either love or hate. These type of items are difficult to make recommendations on, because the user reaction to them tends to be diverse and unpredictable. Music is also full of these items. One another important reason for why we do not have satisfactory solutions yet is the stuff to be done is very complex. As it can be observed from the Table 3 below, it takes a lot of variables to do even the simplest recommendations.[6]



Social Recommender API

Capturing behavior in a retail site	Item visited Recommended item visited Item reviewed Item added to shopping cart Item added to favorites Item added to wishlist Item purchased
Capturing behavior in a content site	Content visited Recommended content visited Content reviewed Content searched Content uploaded Content downloaded
Getting recommendations	Get item recommendation Get user recommendation Explain a recommendation
Session start/end	Session start Session end
Rating items	Rate an item Show item rating
Tagging users and items	Set user tags Get user tags Set item tags Get item tags

Table 3: Social Recommender API

In the industry companies deal with big data. Also there are a lot of research lab in universities/industry working on improve already known recommender algorithm. It is a very challenging and wide research space. We have analyzed well known algorithms and decided to use the one that is most useful for our project. After achieve to develop a basic recommender, we will try to improve new ideas to contribute to the literature.

Most recommender systems found on the web are server-based and centralized. This suits the typical scenario where the relationship between the user and the recommender system is passive - a background process monitors their behavior and the resulting recommendations are embedded in the user interface with little or no opportunity for immediate interaction or refinement. [8]

The solution will be available to use in many different areas. The main advantage our product will provide to people is that it saves time and money for them. Our recommendation system can be used for the systems that use graphs to represent data.

On other hand, we will highly benefit from this project. While we will be improving our technical skills we also learn the whole processes of a project from scratch.

This project is perfect to use for both academic and commercial business. Firstly, we have plans for commercial issues. Deadline for pre-application of ‘Teknogirişim’ is very close, it is November 1. So, we will apply to Teknogirişim at first. Also we see this application as a change for understanding projects requirements, especially from customer and business sides, in detail. Secondly, we have also academic plans. At the moment, we are mostly concentrating on publishing paper or a journal in a scientific journal if we can reach interesting or useful results in our study. On the other hand, if we find an international conference that will take place in Turkey, maybe we can have a chance to participate with our study’s paper by sending them our abstract. In Turkey, we can also publish a paper in Turkish Journal of Engineering & Environmental Sciences, which is a journal that TUBITAK publishes.

3. Draft Project Plan

By creating a recommendation system for specific environments and platforms which predicts the potential “rating” of the user on different items based on a collected data and recommends user the items that they may be interested in. To create meaningful recommendations to users, 3 different approaches can be used in the background of recommendation engines[2] :

1) Collaborative Filtering (CF) : in this approach, a user is recommended items based on the user’s past behaviors, activities or ratings. Also recommender predicts what users will like based on their similarity to other users. Examples of collaborative filtering:

- Amazon.com: uses item-to-item collaborative filtering (people who buy x also buy y).
- Last.fm: based on tracking what users listen to and makes suggestions based on the users tastes.
- Social Networks: they are also using collaborative filtering by examining the network of connections between a user and their friends

Collaborative filtering algorithms have several advantages. First, algorithms with this approach are capable of taking an object/item quality or defect into account when suggesting objects/items, particularly in explicit customer rankings. Second, it is especially applicable and useful to use collaborative filtering algorithms when the analysis of domain’s content is very expensive or difficult without demanding any domain of knowledge. Although the collaborative filtering algorithms has several advantages and their quality level improve during the time, but startup in recommendation system is the most important problem in the situation that many objects and items are provided in the system but there are few customers and few or no rankings. This problem is named “*cold start*” and means that recommendation system cannot produce any suggestion or recommendations [1]. Many studies that involve different methods have been done to solve this problem. Second problem is named “*gray sheep*” and means that it is hard to do a recommendation system for people who do not belong to a part of an obvious group. Collaborative filtering is useful and work well for customer and user who are a part of a particular group with lots of similar neighbors. Scalability is another challenge for collaborative filtering. If the number of objects and customer increase, the traditional form of collaborative

filtering will have problems with scalability problem. The intricacy of collaborative filtering will be increased when the population of customers and the number of objects and items are big. Another challenge is synonymy. This problem comes from naming similar items differently.
CF methods can be further sub-divided into neighborhood-based and model-based techniques. [4]

a) Neighborhood-based (or Memory-based) Filtering: To make a prediction, memory-based collaborative filtering algorithms use the whole or a sample of the user-item database. Users are grouped as people with similar interests. In these algorithms, identifying the neighbors of a new user is needed. Neighbors are the users who have the maximum number of similar interests with new user. After identification of neighbors, a prediction of recommendation for user can be generated.[14]

The neighborhood-based collaborative filtering algorithm is the most known memory-based collaborative filtering algorithm. This algorithm uses the following steps:

I. Calculating the similarity or weight between two users or two items is the first step and it is very important in memory based collaborative filtering algorithms. There are different methods to calculate the similarity. For example, in correlation-based similarity method the formula is given below.

$$w_{u,v} = \frac{\sum_{i \in I}(r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I}(r_{u,i} - \bar{r}_u)^2}\sqrt{\sum_{i \in I}(r_{v,i} - \bar{r}_v)^2}}$$

Here $w_{u,v}$ is the similarity between two users u and v. On the other hand, it can be $w_{i,j}$ for the similarity between two items i and j. This method is also called Pearson correlation. $i \in I$ summations are over the items that both users u and v have rated. $\bar{r}_u$ is the average rating of commonly rated items of user u. Item based similarity calculation is similar to user based one. [14]

II. Producing a prediction for the active user. In the neighborhood-based collaborative filtering algorithm, weighed aggregate of the subset of nearest neighbors of the active user is used to produce a prediction.

$$P_{a,i} = \bar{r}_a + \frac{\sum_{u \in U}(r_{u,i} - \bar{r}_u).w_{a,u}}{\sum_{u \in U}|w_{a,u}|}$$

Here, $\bar{r}_a$ and $\bar{r}_u$ are the average ratings of user a and user u respectively. $w_{a,u}$ is the similarity between user a and user u. $u \in U$ summations are over the users who have rated on item i. [14]

III. Generating top-N recommendations is step is to recommend N top-ranked items. There are user-based and item-based top-N recommendation algorithms. [14]

There are many different algorithms to improve neighborhood-based CF performance:

a. **Item-based Collaborative Filtering:** Forecast a user's preference towards an item by looking at his/her preferences towards similar items.

b. **Significance Weighting:** If active user has few co-rated items with other users, these neighbors may be bad predictors. One approach is to multiply the similarity weight by a *significance weighting* factor, which devalues the correlations based on few co-rated items [9].

c. **Default Voting:** It turns out that correlation, as a similarity measurement, does not work very well on sparse data sets. That is, when two users have few items in common, their weights tend to be over-emphasized. Default voting simply adds a number of imaginary items that both have rated in common in order to smooth the votes[10].

d. **Inverse User Frequency:** There is an intuition that commonly enjoyed items are less important to weight than rarer items. That is, if everyone liked Star Wars, it doesn't help as much in determining weight as two people enjoying a rare independent film together. The inverse frequency can be defined as[10] :

$f_j = \log(n/n_j)$, where *nj* is the number of users who have rated item *j* and *n* is the total number of users.

e. **Case Amplification :** This is a simple one - we simply amplify each weight by an exponent so that higher weights get higher and lower ones get lower. This algorithm emphasizes high weights and punishes low weights[9]:

$$w'_{i,j} = w_{i,j} \cdot |w_{i,j}|^{\rho-1},$$

where *p* is the case amplification power, *p>=1* , and a typical choice of *p* is 2.5

b) Model-based Collaborative Filtering: These algorithms provide item recommendation by first developing a model of user ratings. Model-based CF take a probabilistic approach based on this model. The model (such as machine learning, data mining algorithms) can allow the system to learn to recognize complex patterns based on the training data, and then make intelligent predictions for the collaborative filtering tasks for test data or real-world data, based on the learned models. In other words, unlike memory-based CF, it assumes that the similarity between users and items is simultaneously induced by some hidden lower-dimensional structure in the data. [14]

You can see the strengths and weaknesses of this approach below.

Advantages

- It handles the sparsity better than memory based approach.
- This approach also helps with scalability with large data sets.
- Model-based filtering improves the prediction performance.
- Model-based filtering gives an intuitive rationale for the recommendations.

Disadvantages

- Model building is expensive with this approach.
- There is necessity to make a decision between prediction performance and scalability.
- It is possible to lose useful information due to reduction models with this approach.
- A number of models have difficulty explaining the predictions.

There are many algorithms used by model-based CF, some of them are given below:

i. **Bayesian Belief Net CF Algorithms**: Bayesian networks are graphical models. The nodes represent variables and edges represent direct connections between them. These direct connections are often causal connections. In addition, the quantitative strength of the connections between variables in bayesian networks model is allowing probabilistic beliefs about them to be updated automatically when new information becomes available.[12]

ii. **Clustering CF Algorithms:** Clustering techniques work by identifying groups of users with similar preferences. Once the clusters are created, predictions for an individual can be made by averaging the opinions of the other users in that cluster. Some clustering techniques represent each user with partial participation in several clusters. The prediction is then an average across the clusters, weighted by the degree of participation.

iii. **Regression-Based CF Algorithms:** Regression-based approach to collaborative filtering searches for similarities between items, builds a collection of experts in the form of simple linear models, and combines them efficiently to provide preference predictions for an active user.[5]

iv. **MDP-Based CF Algorithms:** Markov decision processes provides a mathematical framework for modeling decision making in situations where outcomes are partly random and partly under the control of a decision maker. MDPs introduce two benefits: they take into account the long-term effects of each recommendation, and they take into account the expected value of each recommendation.[7]

v. **Latent Semantic CF Models:** It bears some similarity with clustering methods.

2) Content-based Recommending: This system based on recommending items that are similar to those that a user liked in the past. This approach generally makes better personalized recommendations by knowing more about a user, such as personal interests, or about an item, such as director of a movie. [2]

Pandora Radio is a popular example of a content-based recommender system that plays music with similar characteristics to that of a song provided by the user as an initial seed. There are also a large number of content-based recommender systems aimed at providing movie recommendations, a few such examples include Rotten Tomatoes, Internet Movie Database, Jinni, Rovi Corporation.

There are advantages and disadvantages of this filtering. We can look at these strengths and weaknesses by listing them as below[15]:

<u>Advantages</u>

- In collaborative filtering, there is necessity of other users' rating to find the similarity between the users and then give the suggestion. Instead, content-based method only have to analyze the items and user profile for recommendation.
- Collaborative method gives a recommendation because some unknown users have the same taste like you, but content-based method can tell you they recommend you the items based on what features.
- Collaborative filtering have cold start problem, but in content-based filtering new items can be suggested before being rated by a substantial number of users.

<u>Disadvantages</u>

- If the content is in lack of enough information to distinguish the items precisely, the recommendation cannot be precisely done.
- Content-based method provides a limited innovation, because it has to match up the features of profile and items. That is, over specialization makes recommendation difficult.
- When there's not enough information to build a solid profile for a user, the recommendation could not be provided correctly. It is generally a problem for new users.

Content-based recommendation systems analyze item descriptions to identify items that are of particular interest to the user. Therefore, ‘item representation’ and ‘user profiles’ are becoming very important concepts for content-based recommendation systems. [16]

Item Representation: Items that can be recommended to the user are represented by a set of features. In most content-based filtering systems, item descriptions are textual features extracted from Web pages, emails, news articles or product descriptions. Textual features create a number of complications when learning a user profile, due to the natural language ambiguity such as[11]:

➔ Polysemy :the presence of multiple meanings for one word (relevant information can be missed if the pro-file does not contain the exact keywords in the documents)

➔ Synonymy : multiple words with the same meaning (wrong documents could be deemed relevant)

Semantic Analysis is a very powerful and innovative approach to solve these problems. The key idea is the adoption of knowledge bases for annotating items and representing profiles in order to obtain a semantic interpretation of the user information.[15]

User Profiles : Machine learning(ML) techniques, generally used in the task of inducing content-based profiles, are well-suited for text categorization. Most important ML techniques are: Decision Trees and Rule Induction, Nearest Neighbor Methods, Relevance Feedback and Rocchio’s Algorithm, Linear Classifiers, and Probabilistic Methods and Naïve Bayes.

3) Hybrid Approaches: Hybrid recommendation systems are mix of single recommendation systems as sub-components. This hybrid approach was introduced to cope with a problem of conventional recommendation systems. Netflix is a good example of hybrid systems. [2]

As the project's type is not specified yet, the system's prototype is not so strict. Also, we will be referring to already existing systems when the platform is decided. Likewise the basic functions of a recommendation tool, described below picture, it will include 3 main parts: data preparation, log mining and actual program. In data preparation data cleaning and identification staff will be handled. Then, transaction file is composed of algorithms and rules behind the system. Finally the main part, the recommender will gather the information through servers and then displays the result recommendations to the client. [21]

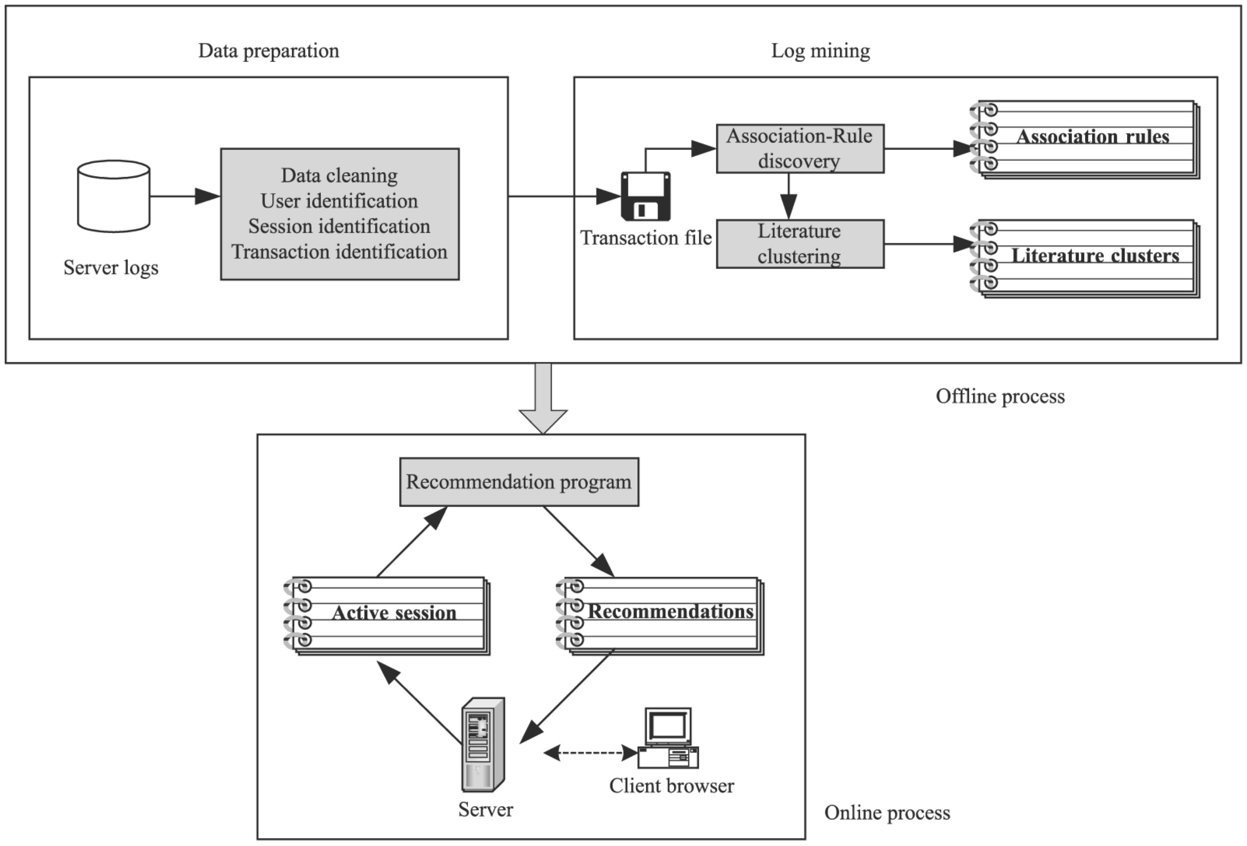


Figure 1: Architecture of a literature recommendation system

Actually the end-product and its usage will be determined on the website/service that we will integrate out project into. Since different environments will require different user interfaces, our project may result with different interfaces using same algorithms and techniques in the background. Briefly, we can say that we will use the end-product to create customized recommendation systems(mainly interface customization) for different services which either have ineffective and useless recommendation systems or have no recommendation system at all.

- Briefly describe (or prepare a table about) the distribution of the major tasks among project members.

Database management is a topic that we all interested in. Therefore, at this side of the project we will work together. For the rest of the project, we are planning to divide labor into 2 working area: client side and server side. We have not distributed working areas about them yet.

- Use Case Diagrams for users and stakeholders

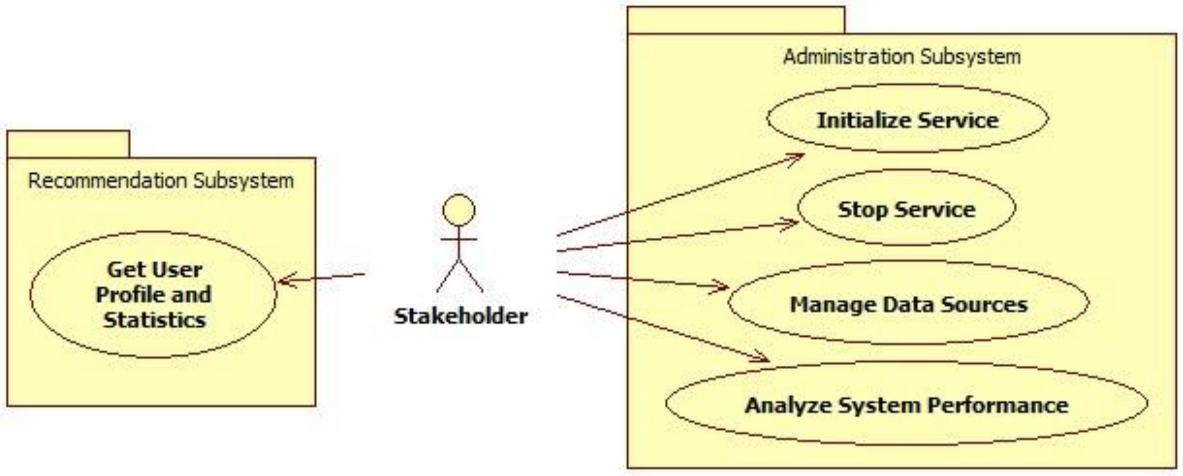


Figure 2: User Use Case

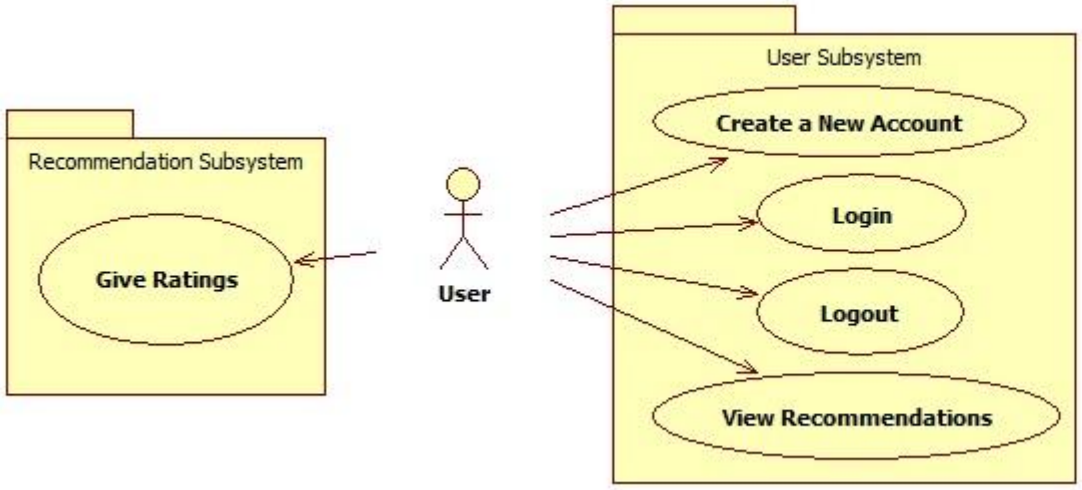


Figure 3: Stakeholder Use Case

User of the system:

General explanation for use case diagrams above:

- **Create a new account:** When the user creates a new account on the website/application that the recommendation system runs on, also implicitly an account for the user inside the recommendation system will be created to store information about the user profile and statistics to provide better recommendations.
- **Login:** When the user logs in to the website; Music Recommendation System will be informed and a recommendation session for the user will start and generate recommendations.
- **Logout:** When the user logs out from the website; Music Recommendation System will be informed and the associated recommendation session for the user will be closed.
- **View recommendations:** Whenever new recommendations are generated, they will be shown to the user through the recommendation GUI which is embedded into the GUI of the main system. The user will be able to view recommendations, click on them to listen the songs
- **Give ratings:** The user will give ratings to the songs that are generated as recommendations on the GUI of our system. These ratings will be stored to provide better recommendations in the future.
- **Initialize service:** This functionality allows the administrator of the system to initialize the recommendation service.
- **Stop service:** This functionality allows the administrator of the system to stop the recommendation service.
- **Manage data sources:** The administrator of the service will be able to manage the data sources, by modifying the existing database. The admin will be able to add new data into the system, as well as alter the existing data.
- **Analyze System Performance:** The administrator will analyze the system performance which is defined as accuracy of recommendations. The analysis will be based on the statistics of recommendations shown to the user, and the information whether the user listened any of the recommendations and what rating they provided.
- **Get User Profile and Statistics:** The administrator will be able to use this functionality to retrieve the information regarding to users, along with the recommendation history of this user.

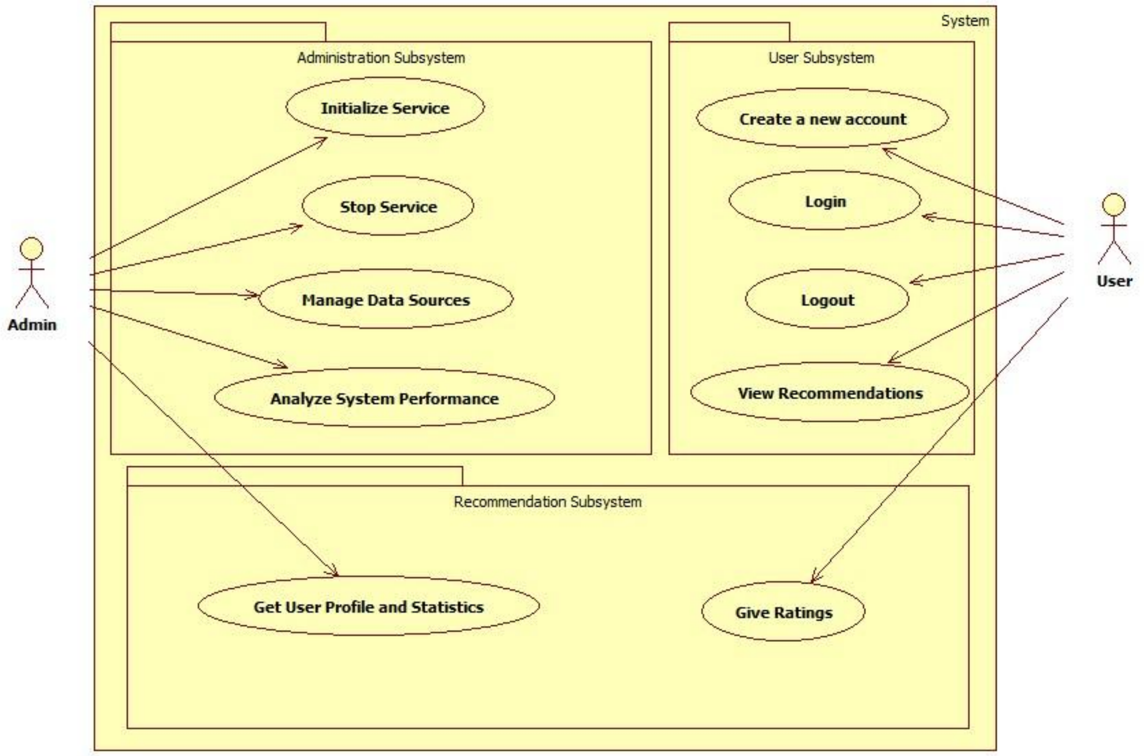


Figure 4: General Use Cases

- Component Diagram: The project is formed by 4 main submodules: Database, Recommender, Evaluation and End User GUI. Detailed explanation about the relations between these modules will be given in Software Design Document.

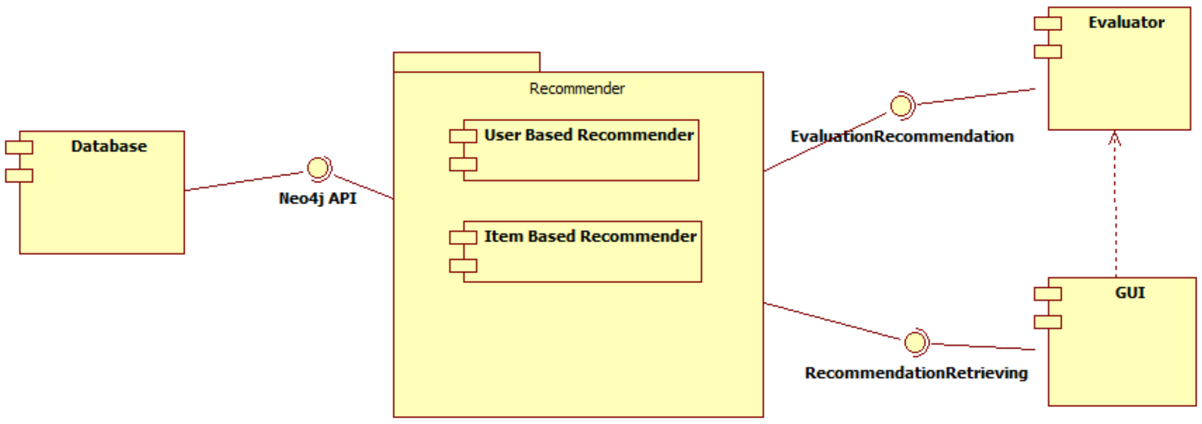


Figure 5: Component Diagram

- Deployment diagram: Deployment of these modules is explained in the below diagram:

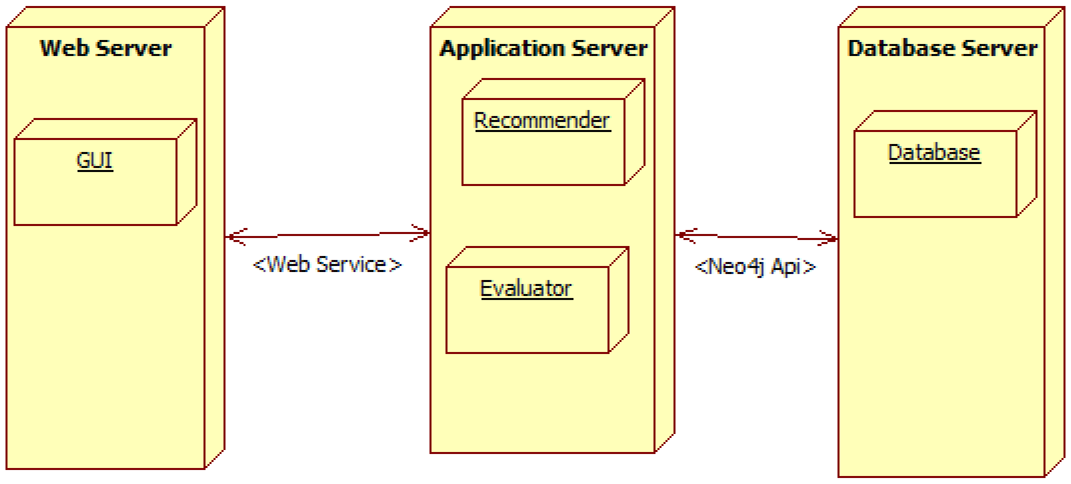


Figure 6: Deployment Diagram

Database Server: This component holds all the data collected for the recommendation system. The data it holds is accessed and processed to perform recommendations. All the database management related software we produce will be running on this component.

Website/Service Server: This is the main server of the application that we will integrate our recommendation system into. The user will access the recommendation system through this component by logging into the system and using it. Depending on the product, the user will get recommendations automatically or upon request.

3. Support

We will be supported by "AGMLab" in this project. Below are the contact information of the AGMLab ODTU Teknokent office:

ODTÜ Teknokent, Silikon Building Floor:1 , No: 25 06531 - ODTÜ | Ankara / TÜRKIYE

Tel :0 312 210 13 40, **Faks :** 0 312 210 13 41

The support we will be getting from AGMLab will be on 3 different subjects, which can be categorized as counseling , providing test data and possibly providing a cluster. They will be teaching us and giving information on some paradigms, methods and tools that are being used to create algorithms and manage some specific databases such as NOSQL which are required to create our recommendation system. They might also be helping us on making decisions about some technical decisions we will have to make on different stages of our project. The main support we are going to get is access to a large data set to test our software on. Even though we do not know the exact form of the data, it will mostly contain information and possibly some statistics about a music database and the actions of the users on this database. Besides including about a million music entities in the database, it will also include information about user's actions such as the songs they have listened and purchased and so on. We have been told that during further stages on our development we may have an access to a computer cluster to parallelize our computations since processing such a large database to create useful information probably will take a lot of time in some cases.

AGMLab might want to use either our entire project or some parts of it in their own projects, in case they decide so. However we will have freedom to enter various competitions or expand the boundaries of the project if we happen to find different data, information and help from other sources.

4. References

[1]http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.92.1111&rep=rep1&type=pdf

[2]http://en.wikipedia.org/wiki/Recommender_system

[3]http://findoutyourfavorite.blogspot.com/2012/04/content-based-filtering.html

[4]http://maisqual.squoring.com/wiki/index.php/Recommender_Systems#Structure_of_Learning_System

[5]http://ml.stat.purdue.edu/vdb/joke/Papers/cf_regression_vucetic05.pdf

[6]http://readwrite.com/2009/01/28/5_problems_of_recommender_systems#awesm=~okb0Pkha7ZM45o

[7]http://research.microsoft.com/en-us/um/people/heckerman/sbh02uai.pdf

[8]http://www.bbc.co.uk/blogs/researchanddevelopment/2012/05/client-side-recommendations.shtml

[9]https://www.coursera.org/course/recsys

[10]http://www.cs.carleton.edu/cs_comps/0607/recommend/recommender/memorybased.html

[11]http://www.cs.ust.hk/~qyang/Docs/2005/Sigir05Xue.pdf

[12]http://www.csse.monash.edu.au/bai/book/BAI_Chapter2.pdf

[13]https://www.google.com.tr/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0CDIQFj

[14]http://www.hindawi.com/journals/aai/2009/421425/

[15]http://www.ics.uci.edu/~welling/teaching/CS77Bwinter12/handbook/ContentBasedRS.pdf

[16]http://www.ijceronline.com/papers/Vol3_issue5/Part.2/H0352047052.pdf

[17]http://www.vikas.sindhwani.org/recommender.pdf

[18]http://www.win.tue.nl/~laroyo/2L340/resources/recommender-systems-e-commerce.pdf

[19]http://www.wisegeek.com/what-are-recommender-systems.htm

[20]http://cran.r-project.org/web/packages/recommenderlab/vignettes/recommenderlab.pdf

[21] http://www.emeraldinsight.com/journals.htm?articleid=862217&show=html