SOFTWARE DESIGN, IMPLEMENTATION, APPLICATION, AND REFINEMENT OF A BAYESIAN APPROACH FOR THE ASSESSMENT OF CONTENT AND USER QUALITIES

A THESIS

SUBMITTED TO THE DEPARTMENT OF COMPUTER ENGINEERING

AND THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Melihcan Türk

September, 2011

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Prof. Dr. Halil Altay Güvenir(Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Dr.-Ing. Markus Schaal

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Dr. David Davenport

Approved for the Graduate School of Engineering and Science:

Prof. Dr. Levent Onural
Director of the Graduate School

ABSTRACT

SOFTWARE DESIGN, IMPLEMENTATION, APPLICATION, AND REFINEMENT OF A BAYESIAN APPROACH FOR THE ASSESSMENT OF CONTENT AND USER QUALITIES

Melihcan Türk
M.S. in Computer Engineering
Supervisor: Prof. Dr. Halil Altay Güvenir
September, 2011

The internet provides unlimited access to vast amounts of information. Technical innovations and internet coverage allow more and more people to supply contents for the web. As a result, there is a great deal of material which is either inaccurate or out-of-date, making it increasingly difficult to find relevant and up-to-date content. In order to solve this problem, recommender systems based on collaborative filtering have been introduced. These systems cluster users based on their past preferences, and suggest relevant contents according to user similarities. Trust-based recommender systems consider the trust level of users in addition to their past preferences, since some users may not be trustworthy in certain categories even though they are trustworthy in others. Content quality levels are important in order to present the most current and relevant contents to users. The study presented here is based on a model which combines the concepts of content quality and user trust. According to this model, the quality level of contents cannot be properly determined without considering the quality levels of evaluators. The model uses a Bayesian approach, which allows the simultaneous *co-evaluation* of evaluators and contents. The Bayesian approach also allows the calculation of the updated quality values over time. In this thesis, the model is further refined and configurable software is implemented in order to assess the qualities of users and contents on the web. Experiments were performed on a movie data set and the results showed that the Bayesian co-evaluation approach performed more effectively than a classical approach which does not consider user qualities. The approach also succeeded in classifying users according to their expertise level.

Keywords: Information quality, web 2.0, collaborative systems, recommender systems, collaborative filtering, Bayesian networks, co-evaluation, trust-based systems.

ÖZET

BAYES YAKLAŞIMI KULLANILARAK İÇERİK VE KULLANICI NİTELİKLERİNİN DEĞERLENDİRİLMESİ İÇİN YAZILIM TASARIMI, KODLAMASI, UYGULAMASI VE GELİŞTİRİLMESİ

Melihcan Türk
Bilgisayar Mühendisliği, Yüksek Lisans
Tez Yöneticisi: Prof. Dr. Halil Altay Güvenir
Eylül, 2011

İnternet, her türden bilgiye ulaşmak için önemli bir kaynaktır. Teknolojik yenilikler, çok sayıda insanın internette bilgi paylaşabilmesini sağlamıştır. Bu durum, internetteki gereksiz bilgi miktarının artmasına yol açmış ve insanların ihtiyaç duydukları bilgiye ulaşması zorlaşmıştır. İnsanların gereksinim duydukları bilgiye kolayca ulaşabilmeleri için *Katılımcı Filtrelemesini Kullanan Tavsiye Sistemleri* ortaya çıkmıştır. Bu sistemler, kullanıcıların geçmiş tercihlerini göz önünde bulundurarak onları gruplandırır ve aralarındaki benzerlikleri dikkate alarak ilgili olan içerikleri onlara önerir. *Güvenirlik Tabanlı Sistemler* kullanıcı benzerliklerinin yanı sıra onların dürüstlük seviyelerini de hesaba katar. Çünkü bir kullanıcının birkaç konuda güvenilir olması o kişinin her konuda güvenilir olacağı anlamına gelmez. İnternetteki içeriklerin kalitesi de kullanıcılara uygun ve güncel içerikleri sunabilmek için önemlidir. Bu tezde, içerik kalitesi ve kullanıcı güvenirliği kavramlarını bir arada kullanan bir çalışmayı baz aldık. Bu çalışmaya göre, içeriği değerlendiren kişilerin kaliteleri hesaba katılmadan, o içeriğin kalitesi belirlenemez. Çalışmada, içerik ve kullanıcı kalitelerinin bir arada belirlenmesine imkan veren bir Bayes yötemi kullanılmıştır. Bayes yöntemi, bu kalite değerlerinin zaman içerisindeki değişimlerinin hesaplanmasına da olanak sağlar. Bu tez için, yukarıda bahsedilen modeli geliştirip, kullanıcı ve içerik kalitelerini değerlendirmek için kullanılacak bir yazılım geliştirdik. Bir film veri seti üzerinde yaptığımız deneyler, modelimizin kullanıcı kalitelerini göz önünde bulundurmayan klasik yöntemden daha iyi sonuç verdiğini göstermiştir. Yöntemimiz, kullanıcıları uzmanlık seviyelerine göre sınıflandırmada da başarılı olmuştur.

Anahtar sözcükler: Bilgi kalitesi, katılımcı sistemler, tavsiye sistemleri, katılımcı filtrelemesi, Bayes ağları, kullanıcı güvenirliği.

Acknowledgement

I am grateful to my supervisor Prof. Dr. Halil Altay Güvenir for his suggestions and criticisms about my study. I am also grateful to Dr.-Ing. Markus Schaal who has allowed me to use his previous studies and supplied essential material for my study. I would like to thank Dr. David Davenport for his valuable comments and suggestions.

I would like to address my thanks to The Scientific and Technological Research Council of Turkey (TÜBİTAK) for their financial support within National Scholarship Programme for MSc Students.

I would like to thank my parents and sister who encouraged and supported me during my studies.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The invention of the World Wide Web [15] is a milestone in information sharing, allowing many to reach greater amounts of information in any field. At the beginning, it was designed as read-only, but later, the readers themselves were able to create new materials or new links on the web.

The second wave of World Wide Web technology, Web 2.0 [21], is built on the concepts of information sharing, user-centered design and collaboration, resulting in the establishment of Wikis (e.g. Wikipedia [12]), blogs (e.g. Blogger [2]), and social networks (e.g. Google+ [7], Facebook [5], and Twitter [11]). These web sites allow users to participate on the web as content creators. These developments caused the amount of information available on the web to increase rapidly. As a result, a wide variety of information can now be accessed using the internet.

1.1 Statement of the Problem

The World Wide Web technology provides unlimited access to vast quantities of information. However, the information on the web is created without a control mechanism. As a consequence, there is a great deal of material which is either inaccurate or out-of-date, making it increasingly difficult to find relevant and

up-to-date content.

In order to solve this problem, some filtering mechanisms have been designed to cope with the mass of information available. Recommender systems were introduced in order to suggest important content items to the web users. Collaborative filtering is one of the methods used in recommender systems. The method considers user feedback on content items in order to group users according to their interests, so that items are recommended if similar users also liked or purchased that item.

However, the collaborative filtering approach has a number of weaknesses in the process of making recommendations. For example, some of the users may not be completely trustworthy in some categories, while they may be reliable in others. Trust-based systems were proposed in order to eliminate these weaknesses; such systems not only consider user similarities, but also the trust level of users.

The concept of information quality on the web is also used to address the issues mentioned at the beginning of the section. Information quality is a type of measurement unit used to define the relevancy of information to the needs of people. The concept *information quality* is also used for describing the quality level of the content items on the internet. The quality of each content item is determined by either the users of each item or the information system professionals. Then, only those content items that are relevant and of value to the users are presented.

The bridge between the information quality and trust is defined in an innovative study by Schaal [25]. This study proposes a model for assessing the quality of both contents and users on the web. Since the content and user qualities change according to time, they are updated throughout time.

1.2 Significance of the Study

In this thesis, we suggest improvements in Schaal's model, mentioned at the end of the previous section. Since the methodology can be applied to different systems, we designed and implemented configurable software. We also performed experiments to test the model and refined the methodology.

In the model, the feedbacks submitted by the users for the content items are considered. A feedback identifies the user and carries information about the content, the value, and the submission time. The system processes the feedbacks as they arrive and updates the user and content qualities over time.

1.3 Thesis Outline

The thesis is organized as follows: The related systems are reviewed in Chapter 2. In Chapter 3, the model of co-evaluation of content and user qualities by Bayesian reasoning is presented. The experimental design is explained and the results are discussed in Chapter 4. The aim of the experiments is to demonstrate the feasibility of the approach and measure the ability of the system to identify high quality users. In Section 4.1 the experimental parameters are presented and explained. The results of the experiments are shown in Section 4.2. In Section 4.3, the performance of the system is evaluated. We conclude the thesis and provide future works in Chapter 5.

Chapter 2

Literature Review

The chapter discusses previous works proposed to solve the information relevance problem defined in Chapter 1. Recommender systems were introduced in order to suggest the relevant content items on the web with respect to preferences shown in the past. There are two types of recommender systems. The first type is content based recommender systems, which consider the similarities between the attributes of content items. The second is collaborative filtering, which considers similarities between the preferences of collaborators. Trust based systems were introduced because of the wide variety of preferences across the range of content items. At the end of the chapter, information quality and the innovative applications which combine the concepts of user trust and information quality are discussed.

2.1 Recommender Systems

Due to the vast amount of information on the internet, some mechanisms are needed to filter the unnecessary data. Recommender systems were introduced because of this requirement. The aim of these systems [23] is to help people to locate the necessary information in the shortest possible time. These systems suggest useful content to the users based on their interests. Amazon [1] and eBay

[3] are two of the most famous web-sites which use recommendation systems in making suggestion for their products.

There are two main types of recommendation systems. The first is content-based recommender systems [22], which use the features of the item and the interest of the users. For example, the web site Amazon makes further suggestions of relevant books to customers based on the attributes of the book bought, which may be related to the author or genre of the purchased book.

The other methodology is collaborative filtering [16], which simply uses the user feedbacks on items in order to make recommendations. The details of the methodology are described below.

2.2 Collaborative Filtering

Some web sites allow users to submit feedback about the content items on the site. Collaborative filtering considers this feedback in the recommendation process. The approach is built upon the idea that people often make decisions after considering the suggestions of friends who have similar tastes. In order to determine the users whose tastes are similar, the system groups users according to similarities in feedback given. If one user in a group buys an item, it is also recommended to the others who have not yet purchased the item. In Amazon, collaborative filtering approach is also used. They recommend the relevant books with the statement "the users who bought this book also bought that book".

In the collaborative filtering methodology, information regarding contents is not required because the system relies on the item ratings of users. Recommendations are made taking into account the similarities between user ratings.

2.3 Trust Based Systems

Collaborative filtering recommendation strategy assumes that if the ratings submitted by a group of users are similar, their interests will also be similar for other items. However, the similarities between individuals' tastes differ according to the fields of interest. As a result, not all people will be reliable for all categories, although all will be reliable for some categories.

According to the approach proposed by O'Donovan, the reliability of collaborators should also be considered in the recommendation process [20]. This approach considers that user similarities in themselves are not a sufficient basis for making accurate suggestions, but that users should also have similar preferences and they should be considered trustworthy in order to make reliable recommendations.

According to Abdul-Rahman et al. [13], the concept of trust is defined in two categories: *Context-specific interpersonal trust* describes the trust of a user for a specific situation. In this kind of trust systems, if a user trusts another user under a certain condition, it does not necessarily mean he/she trusts the same user under a different condition. The second type of trust is *system/impersonal trust*, which defines the trust level of a user for the whole system.

Epinions [4] is one of the web-sites which include the trust level of evaluators in their system. The web-site is used to sell goods, and collaborators are allowed to write reviews about items. The system evaluates the reviewers and classifies them according to their trust level. In the suggestion process, both the feedback and the reliability of the reviewer are considered.

2.4 Information Quality

Information quality is a measurement mechanism used to define the quality of content items in the information systems. This mechanism was firstly introduced for applications such as databases and management information systems [17]. In

respect of this study, information quality is defined as the effectiveness of systems and subsystems in relation to the operational goals.

Wang et al. discuss the negative social and economic impacts of poor data quality and put forward the idea that data consumers have a wider perspective of the conceptualization of data quality as compared to information system professionals [27].

The measurement of the data quality on the web is also important because the internet holds great amounts of information [19]. The study by Moraga et al. represents the attributes which are considered relevant for the assessment of data quality in Web portals.

A recently accepted approach regarding the assessment of information quality considers the users point of view. Strong et al. suggest that the data quality cannot be determined independent of the data users [26].

Rieh [24] examines the problem of judgment of information quality and cognitive authority by considering web searching behavior. According to this study, those using the information have to evaluate its quality for themselves. The system creates a subjective environment for each user and the quality perception of an individual user has a little or no effect on the other users.

2.5 A Bayesian Approach for Small Information Trust Updates

As mentioned in Chapter 1, Schaal defined a reputation model for the source of the information in order to determine the quality of the contents on the web. The aim of this model is to assess the quality of the content items and the reputation of the item sources by considering the user feedbacks over time.

Two types of actors are defined in this approach, information providers and voters. The voters are the users who create feedback to evaluate the items.

According to the model, a content item can be evaluated after its activation time span on a binary scale as either *correct* or *wrong*. Voting on the item begins after the start of the activation time and the process concludes at the end of the activation time span. A voter has an option not to rate a particular content item.

In this system, the providers and voters have a value of trust between 0 and 1. The trust values changes throughout time. According to the initial theory, each trust value is 0.5 upon creation. A provider who has a high trust value provides *correct items*, and a provider who has a low trust value provides *wrong items* respectively. A voter whose trust value is high submits *correct votes* and a voter whose trust value is low submits *wrong votes*. Voters with high trust values have an enormous influence on the trust value of a provider. The trust value of a voter will decrease if he submits a vote against the actual trust value of a provider, similarly if a voter submits a vote in favor, his trust value will increase.

In this work, a Bayesian approach is used to assess the change in trust values of voters and providers according to time, because this is considered a well-designed model for the update of knowledge about voters and providers after each voting event.

Chapter 3

Co-Evaluation of Content and User Qualities

3.1 Model

Trust based recommender systems generally use implicit or explicit user feedbacks to evaluate only the quality of the contents on the web. Our model is based on the previous studies of Schaal which is mentioned in Section 2.4. The aim of the model is not only to assess the quality of the content items, but also the quality of the users. We define the quality of the users as their ability to make correct evaluations. Trustworthiness and expertise are some of the attributes which helps users to do right judgments.

In order to determine the content and user qualities, we consider the following elements in our system:

- a set of content items C
- a set of source channels S
- a set of evaluators E
- a set of feedbacks F

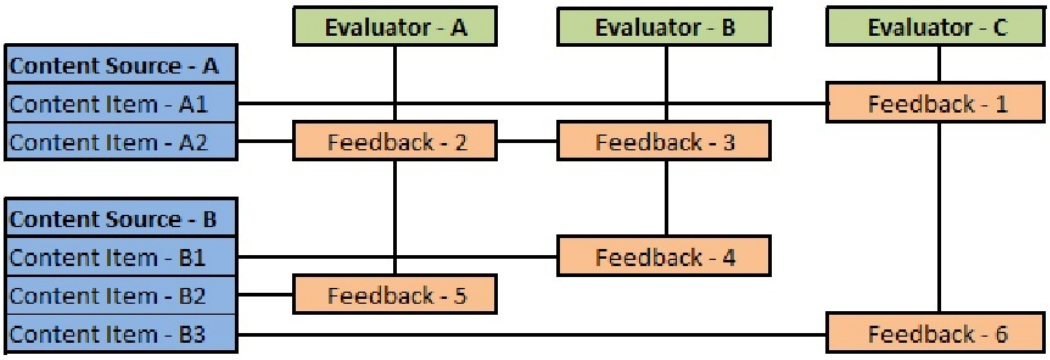


Figure 3.1: The Relations between the Elements of the System

In Figure 3.1, we see the relations between the elements of our system. In the system, each content item ($c_n \in C$) is a target which is created by a content provider. The versions of the articles in Wikipedia, blog entries in blog spaces, and the contents (videos, notes, images, etc.) added by the collaborators in social networks are some of the examples of the content items.

A source channel (content source) ($s_n \in S$) is the source of a content item. It can be defined as an element whose quality is correlated with the quality of an individual content item. In the system, the source channels are used in order to group the content items. For each content item c_n, there is only one source channel s_n. However, one source channel can be associated with multiple content items. A blog space is one of the examples of the source channels. In order to determine the quality of a blog space, the individual blog entries should be considered.

In our system, the role of an evaluator (e_n) is to provide feedbacks. A feedback hold the information about the content item (c_n), the evaluator (e_n), the value, and the submission time which are analyzed in Section 3.2 in detail.

3.2 User Feedback

The main input of our system is user feedbacks which contains the following information:

- Content item
- Evaluator
- Value
- Submission time

The model is built on the idea that a feedback holds the information about both a content item and an evaluator, so it can be used to assess the qualities of them. Since the information of the submission time is also available, the change in these qualities can also be observed in timeline. For each feedback, the system updates both the quality of the evaluator and the quality of the source of the content item.

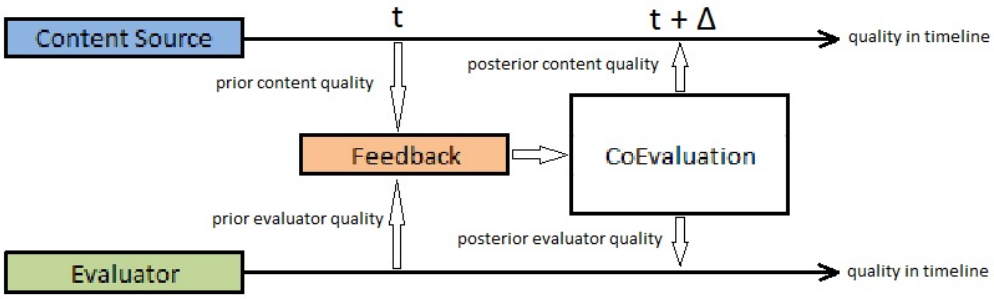


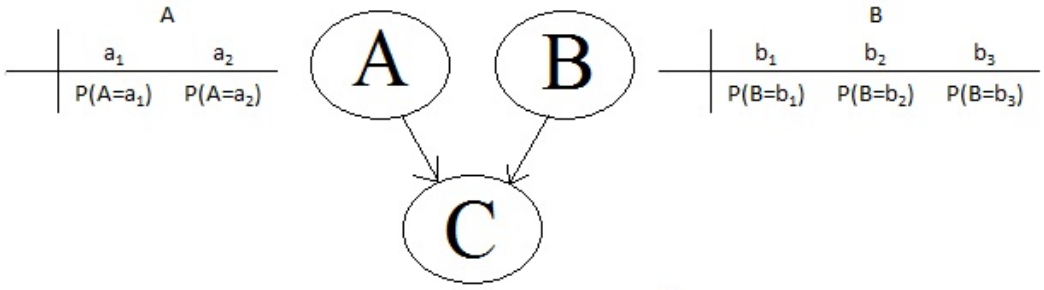
Figure 3.2: Co-Evaluation of Content and Evaluator Qualities

In Figure 3.2, we see the process of co-evaluation for a content source and an evaluator. Here, a *feedback* is submitted by an *evaluator* to one of the content items which belongs to a specific *content source* in time *t*. Our method uses the feedback *feedback* and the prior quality values of *content source* and *evaluator* in

order to calculate the posterior quality values of them. The posterior values are stored Δ time later than the time in which prior values are received. The value of Δ is defined as update frequency of our online processing methodology.

3.3 Bayesian Reasoning

In order to determine the qualities, we use a Bayesian approach. The approach is built based on the studies [14] of Thomas Bayes. By using these studies, Jensen and Nielsen introduced the concept of Bayesian networks and decision graphs [18] which are the graphical models of the approach. A Bayesian network is a directed acyclic graph which represents a set of variables and their conditional dependencies. Nodes of Bayesian networks represent the variables and edges represent conditional dependencies.



A	B	C c_1	C c_2	C c_3
a_1	b_1	$P(C=c_1 \mid A=a_1, B=b_1)$	$P(C=c_2 \mid A=a_1, B=b_1)$	$P(C=c_3 \mid A=a_1, B=b_1)$
a_1	b_2	$P(C=c_1 \mid A=a_1, B=b_2)$	$P(C=c_2 \mid A=a_1, B=b_2)$	$P(C=c_3 \mid A=a_1, B=b_2)$
a_1	b_3	$P(C=c_1 \mid A=a_1, B=b_3)$	$P(C=c_2 \mid A=a_1, B=b_3)$	$P(C=c_3 \mid A=a_1, B=b_3)$
a_2	b_1	$P(C=c_1 \mid A=a_2, B=b_1)$	$P(C=c_2 \mid A=a_2, B=b_1)$	$P(C=c_3 \mid A=a_2, B=b_1)$
a_2	b_2	$P(C=c_1 \mid A=a_2, B=b_2)$	$P(C=c_2 \mid A=a_2, B=b_2)$	$P(C=c_3 \mid A=a_2, B=b_2)$
a_2	b_3	$P(C=c_1 \mid A=a_2, B=b_3)$	$P(C=c_2 \mid A=a_2, B=b_3)$	$P(C=c_3 \mid A=a_2, B=b_3)$

Figure 3.3: An example of a Bayesian Network

In order to make the concepts clear, we give an example of a Bayesian network which is presented in figure 3.3. The nodes A, B, and C represent the variables and the arrows show that there are conditional dependencies between the variables

$A - C$ and $B - C$. The variables A and B are called the parent variables of C. The values of each variable and their probabilities are associated with the variable. According to the figure, the variable A has two values (a_1, a_2), and the variables B, and C have three values (b_1, b_2, b_3 and c_1, c_2, c_3 respectively) for each. The probabilities of these values (probability distribution) are given below the values of each variable. $P(Variable = value_n)$ represent the probability of $Variable$ being $value_n$. For the variable C, the probabilities take two inputs (one for each variable) which are used to represent the conditions. The conditions are the possible combinations of the values of the parent variables. Since we have two values for the variable A and three values for the variable B, we have six conditions for each value of the variable C.

Bayesian networks can be used to calculate the updated knowledge of the values of the variables by using the observation of the other variables. For example, if we have a knowledge about the value of the variable C, we can calculate the updated probability distributions of the other variables by considering the prior probability distributions of them and the conditional probability values of the variable C.

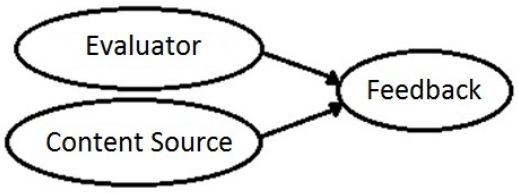


Figure 3.4: Structure of the Bayesian Network

The aim of our methodology is to assess the quality level of evaluators and content sources by considering the values of feedbacks. The basic structure of the Bayesian network which we use in our model is presented in Figure 3.4. The network contains three variables which represent quality of an evaluator, quality of a content source, and a feedback. The values of the variables for an evaluator and content can be varied according to the application. However, they should be orderable in the sense of quality. The values of a feedback variable are determined according to the data set. They should be same as the values of the feedbacks in the data set.

Since the value of a feedback is conditionally dependent on the quality level of the relevant evaluator and content, the relations are represented by the arrows between the corresponding variables.

In the system, the information about the value of a feedback (observed variable) is available, so it can be used to calculate the quality level of evaluators and content sources (target variables). Prior probability values of the target variables, conditional probability values of the feedback variable, and the observed value of the feedback are considered in this calculation. The prior probability values represent the distribution of evaluators and content sources according to the quality level. For example, we consider two values (high and low) for the evaluator quality variable. If the numbers of high and low quality evaluators are assumed to be equal, the prior probabilities are uniformly distributed to the values high and low (0.5 for each).

As feedbacks are processed, posterior probability values are calculated by using the prior probabilities. The posterior probabilities which are inferred for a target (evaluator or content) by using a Bayesian network are considered as prior probability values of the same target in different network which is created later.

The conditional probabilities of the feedback variable are determined according to the values of the target variables. For a high quality content source, the value of the feedback is most likely high. The value of the feedback is probably low for a low quality content source. Since the high quality evaluators are able to make accurate judgments, the variances of the conditional probability distributions are low. However, the variances are high for low quality evaluators.

3.4 Software Module

In this section, we explain the details about the software module of our system. Since our aim is to apply the methodology to various fields, design of a configurable software system is needed.

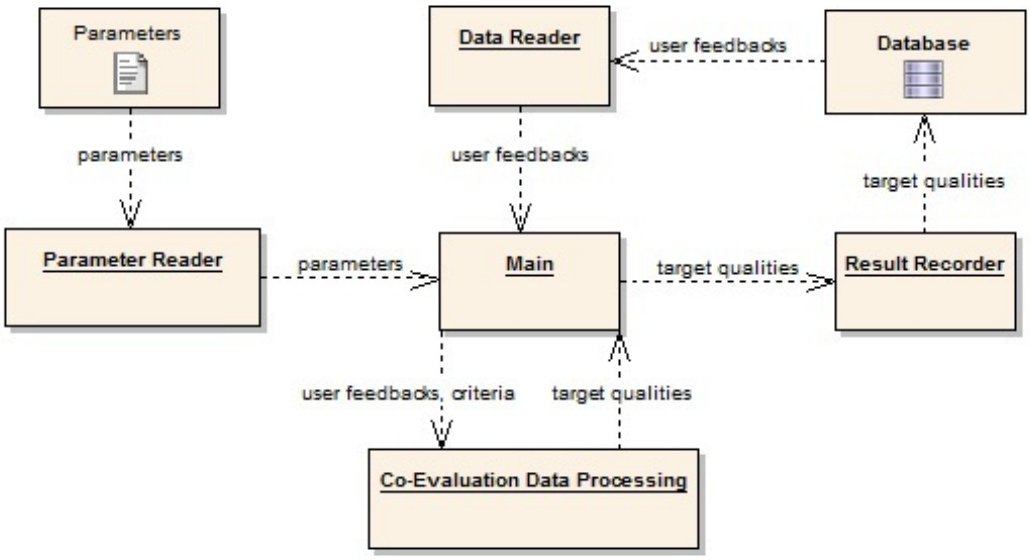


Figure 3.5: System Architecture

Our system gets the relevant data from database, processes them, and stores the calculated quality values to the corresponding table. Figure 3.5 presents the system architecture. The system consists of five main components:

- Parameter Reader
- Data Reader
- Main
- Co-Evaluation Data Processing
- Result Recorder

In order to make the software flexible, the parameters which are necessary to apply the methodology are stored in a text file. *Parameter Reader* is implemented to read the parameters which are used to connect to the relevant database (user, password, database), associate the table columns with the variables, add the variables (name, values) and define the conditional probabilities.

Data Reader is implemented in order to obtain relevant data from database. It is used to retrieve feedbacks and their corresponding evaluators (users) and content items.

Main is responsible for the communication between the components. Firstly, it gets the parameters and use them in order to connect database and retrieve user feedbacks. After that, the feedbacks and criteria are sent to *Co-Evaluation Data Processing* and the qualities of users and content sources (targets) are collected at the end of the process. Finally, results are sent to *Result Recorder*.

Result Recorder is used in order to store the qualities of target variables to the database. Each result has the information about variable, target (unique identifier of evaluator or content), time, value, and probability.

Co-Evaluation Data Processing is responsible for processing the data. It process user feedbacks in order to calculate target qualities.

As we mentioned before, the software can be applied to various fields. For each field, it should be possible to run the software by defining different sets of criteria. In our system, each performance of the software is defined as a *case*. In order to distinguish the outcome of each *case*, the results are stored with their corresponding *case id*.

Figure 3.6 presents the class diagram of the application programming interface (API) of *Co-Evaluation Data Processing*. The main component of the API is *CoEvaluationDataProcessing* class. The class is used as follows:

- A case is setup by the method *requestCaseHandle()*.
- Criteria are defined by the methods *registerCriterion(...)* and *defineModel(...)*.
- A feedback is added by *addFeedback(...)*.
- Feedbacks are processed by *process()*.
- Results are collected by *getResult(...)*.

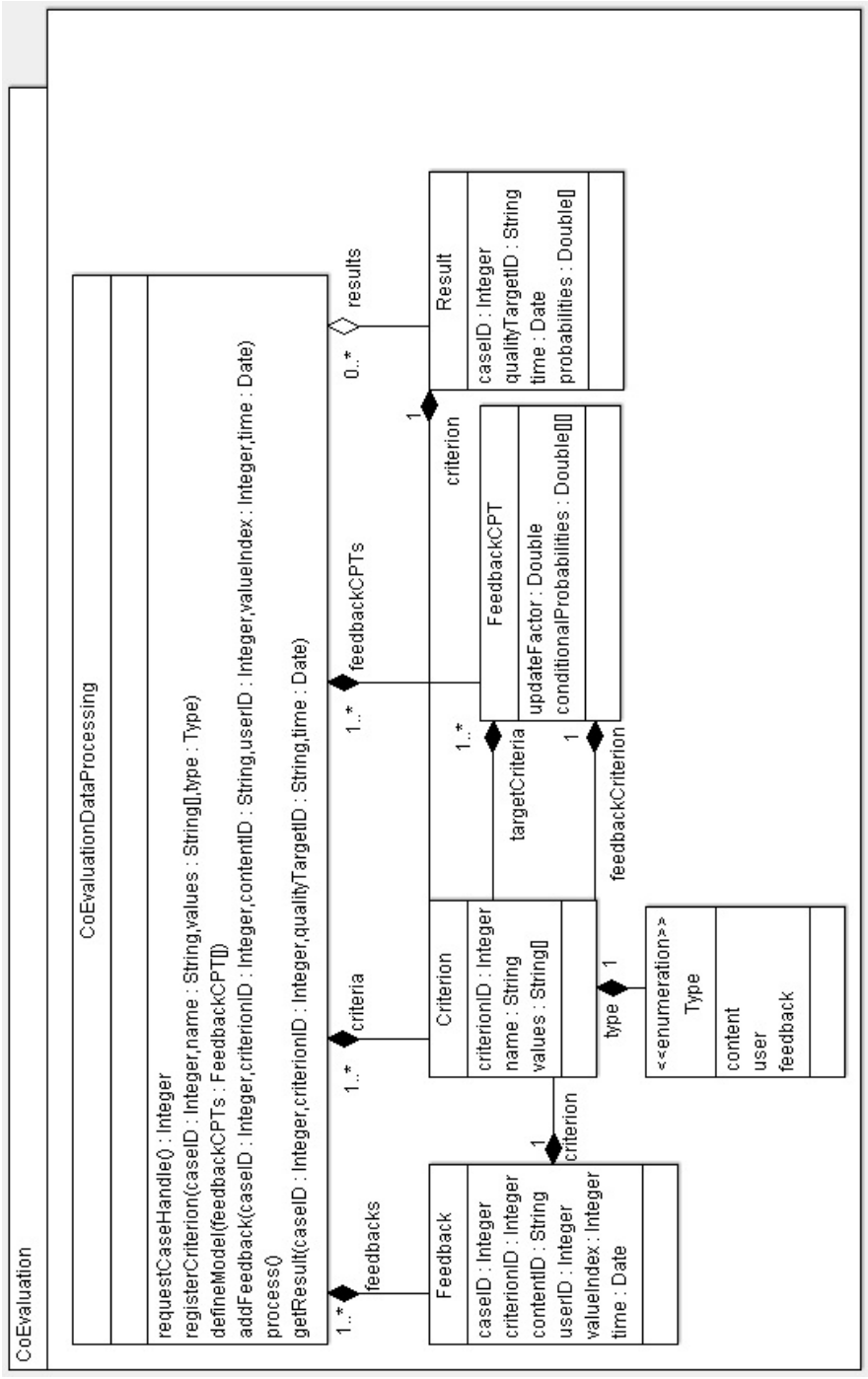


Figure 3.6: Class Diagram of Co-Evaluation API

The roles of the methods are explained below:

- **requestCaseHandle() : Integer** - Creates new case and returns the case ID
- **registerCriterion(caseID : Integer, name : String, values : String[], type : Type)** - Registers new criterion with specified name value and type to a given case
- **defineModel(feedbackCPTs : FeedbackCPT[])** - Defines a model which includes a set of CPTs (conditional probability tables) for given feedback and target criteria.
- **addFeedback(caseID : Integer, criterionID : Integer, contentID : String, userID : Integer, valueIndex : Integer, time : Date)** - Adds a feedback which has specified content, user, value and time to the given case and criterion
- **process()** - Processes all respective data submitted by addFeedback(...)
- **getResult(caseID : Integer, criterionID : Integer, qualityTargetID : String, time : Date)** - Returns result for a particular case, quality target, criterion, and time

The method *process()* uses feedbacks and calculates quality values of contents and users with respect to specified criteria and model. The details of the method is presented in Algorithm 1. The system iterates over all of the feedbacks and builds a Bayesian network for each content item. After previously defined amount of time passes, inference is done for parent variables by considering the value of the feedbacks, prior probability distributions of the target variables and conditional probabilities between the variables. Then the system stores the result for the current time and removes the relevant Bayesian network. The probabilities which are calculated in this step are used as prior probabilities for the next steps. At the end, we have probability distributions of both content sources and users for different time points.

Algorithm 1 Co-Evaluation Data Processing

Input: feedbacks $Feedbacks$, time $period$
create variable $Networks$
for each $feedback$ in $Feedbacks$ **do**
  $content = feedback_{content}$
  $user = feedback_{user}$
  $value = feedback_{value}$
  $time = feedback_{time}$
  $inserted = false$
  for each $network$ in $Networks$ **do**
    if $content$ is in $network$ **then**
      add variable v_{user} to $network$
      add variable $v_{feedback}$ to $network$
      assign probability values of the variables v_{user}, and $v_{feedback}$
      add observation $v_{feedback} = value$
      $inserted = true$
    end if
    if $time - network_{creationtime} > period$ **then**
      for each $variable$ in $network$ **do**
        if $variable_{type} =' content'$ or $variable_{type} =' user'$ **then**
          calculate posterior probabilities of $variable$
          record probabilities of $variable$
        end if
      end for
      remove $network$ from $Networks$
    end if
  end for
  if $inserted = false$ **then**
    create a Bayesian network $network$
    assign $network_{creationtime} = time$
    add variable $v_{content}$ to $network$
    add variable v_{user} to $network$
    add variable $v_{feedback}$ to $network$
    assign probability values of the variables $v_{content}$, v_{user}, and $v_{feedback}$
    add observation $v_{feedback} = value$
    add $network$ to $Networks$
  end if
end for

In Co-Evaluation API, there are also sub-components for defining criterion, feedback, conditional probability table, and result. The class *Criterion* is used to define feedback, user, and content variables and has the attributes *criterionID*, *name*, *values*, and *type*. The attribute *criterionID* specifies unique id of the criterion. The name of the variable is represented by *name* attribute. The values of the variable are stored in the attribute of *values*. The attribute *type* represents whether the variable is in type content, user, or feedback. The enumeration *Type* is also created because of this purpose.

The second class *Feedback* is used to represent the attributes of a feedback. A feedback keeps the information about content, user, value, and time which are identified by the attributes *contentID*, *userID*, *valueIndex*, and *time* respectively. The attribute *criterionID* is used in order to associate the feedback with relevant variable.

The information about a conditional probability table is kept by the class *FeedbackCPT*. The attributes *feedbackCriterion*, and *targetCriteria* represent feedback and target (user and content) variables for which the conditional probability table is generated. In our approach, we do not use posterior probability values, which are obtained as outputs of Bayesian networks, directly. These values affects the general quality by some rate which is defined by *updateFactor*. The calculation of general quality can be seen in Equation 3.1.

$$P_(Quality) = P_{prior}(Quality) \times (1 - updateFactor) + P_{posterior}(Quality) \times updateFactor \tag{3.1}$$

The last attribute *conditionalProbabilities* holds the conditional probabilities in a matrix in the following format:

$$\begin{bmatrix} P(fv_1|cv_1,uv_1) & P(fv_1|cv_1,uv_2) & ... & P(fv_1|cv_2,uv_1) & P(fv_1|cv_2,uv_2) & ... \\ P(fv_2|cv_1,uv_1) & P(fv_2|cv_1,uv_2) & ... & P(fv_2|cv_2,uv_1) & P(fv_2|cv_2,uv_2) & ... \\ ... & ... & ... & ... & ... & ... \end{bmatrix}$$

fv_n, cv_n, and uv_n denote the nth values of feedback, content, and user variables respectively. If the number of the values are f (for feedback variable), c (for content variable), and u (for user variable), the column and row numbers of the

matrix will be $c \times u$ and f respectively.

Result class is used to store the values of the user or content qualities respectively for different times. It holds the information about the case ID (*caseID*), the user or content variable (*criterion*), the description of content source or user (qualityTargetID), the record time (*time*), and the probability values (*probabilities*).

Chapter 4

Experiments

In this chapter, the details about experiments are explained. There are two main aims of the experiments:

- Assessing the content and user qualities
- Demonstrating the feasibility of the approach

In order to perform experiments, the MovieLens data set [10] is used. The data set consists of ratings submitted for movies. A rating also has the information about its evaluator, value, and submission time which are essential inputs of our methodology. The main advantage of the data set is that it includes 10M ratings which are submitted between the years 1996 and 2009. Since many ratings are given in wide range of time, we will have considerable number of results to observe the change in qualities over time.

We performed four types of experiments in order to assess the qualities of movies and evaluators.

- **Experiment 1: Co-Evaluation USER EXPERTISE/ DIRECTOR QUALITY:** In this experiment, we assume that the quality level of a movie is similar to the quality level of its director, so we define directors of

movies as their source channel. We also assume that the expertise of a user affects the ability of him / her to rate the quality of movies correctly. For this experiment, we use a Bayesian network whose variables represent the quality of a movie director and the expertise of a user.

- **Experiment 2: Extension GENRE-Related EXPERTISE/ DIRECTOR QUALITY:** As an extension of the first experiment, we define the expertise of a user as a function of a genre, because high experience of a user in a genre does not mean that the user is also an expert in another genre. For this experiment, the variables of a Bayesian network represent the quality of a director and the genre-expertise of a user.

- **Experiment 3: Co-Evaluation USER EXPERTISE/ MOVIE QUALITY:** In this experiment, we changed the first assumption of the first experiment. A movie data set may not be suitable for the idea of source channel, so we ignore directors. Instead of a director, each movie is its source channel in this experiment. The assumption about users is the same as the assumption in the first experiment, so the variables of a Bayesian network represent the quality of a movie and the expertise of a user.

- **Experiment 4: Extension GENRE-Related EXPERTISE/ MOVIE QUALITY:** This is the extension of the third experiment, and the genre-expertise of a user is used instead of the general expertise.

In order to demonstrate the feasibility of our methodology, we compare the performance of the methodology with a classical approach. Since IMDb [8] is considered to be most reliable system in evaluating movies, top movies in IMDb is chosen as a gold standard for high quality movies. Precisions of two approaches are measured in order to compare the performances of them.

4.1 Experimental Setup

4.1.1 Data Set

As it is mentioned at the beginning of the chapter, we choose the MovieLens data set which contains the ratings of the movies. The data set has the following data:

- 10M ratings
- submitted by 71567 users
- for 10681 movies

We consider two files *ratings.dat* and *movies.dat* in our experiments. The structures of the files are explained below.

4.1.1.1 Ratings

Each line of the file holds the information about a rating value of a movie submitted by a user and has the following format:

UserID::MovieID::Rating::Timestamp

Ratings are submitted on a 5-star scale, with half-star increments.

This file supplies user feedbacks which have the information about users, content items, values, and times for our system.

4.1.1.2 Movies

The file keeps the information about movie titles, and genres of a movie and has the following format:

MovieID::Title::Genres

In order to provide the information about source channels for our experiments, we associate the MovieID values with the IDs of the movies in IMDb. The association allows us to gather the information about directors, writers, and stars of the movies.

Genres column represents the genres from the set {Action, Adventure, Animation, Children's, Comedy, Crime, Documentary, Drama, Fantasy, Film-Noir, Horror, Musical, Mystery, Romance, Sci-Fi, Thriller, War, and Western}.

4.1.2 Experimental Parameters

In this part, we determine the values of some parameters for each experiment. These parameters are the variables of the Bayesian network, their values, prior probabilities of the values, conditional dependencies between the variables, update factor, lifetime of Bayesian networks, and source channel.

4.1.2.1 Variables

For each experiment, we define three variables for content-sources, users, and feedbacks.

As we mentioned in Section 3.3, the features of a feedback variable are determined according to the data set. The values of the feedback variable should be the same as the rating values in MovieLens.

- *Rating*: The variable represents the value of a rating. The values of this variable are 1_*star*, 1.5_*star*, 2_*star*, 2.5_*star*, 3_*star*, 3.5_*star*, 4_*star*, 4.5_*star*, and 5_*star* which are the same as the values in MovieLens data set.

We define the remaining two variables for each experiment below. Since the computational complexity of inference is high, we limit the number of the values to three for each variable.

- **Co-Evaluation USER EXPERTISE/ DIRECTOR QUALITY:**
  - $MovieDirectorQuality$**:** Since we try to find out whether the quality of a movie is similar with the quality of its director, the variable represents the director quality which is the source channel of the corresponding movies. The values of the variable are *low*, *average* and *high*.
  - $UserExperience$**:** The variable shows the experience level of a user. Its values are *novice*, *intermediate*, and *expert*.
- **Extension GENRE-Related EXPERTISE/ DIRECTOR QUALITY:**
  - $MovieDirectorQuality$**:** The variable and its values are same as the director quality variable and its values in the first experiment.
  - $User(Genre_n)Experience$**:** Multiple variables are defined in order to represent the expertise of a user for a specific genre. The names of the variables are determined according to the genre (e.g. UserComedyExperience, UserDramaExperience, etc.). The values of the variables are *novice*, *intermediate*, and *expert*.
- **Co-Evaluation USER EXPERTISE/ MOVIE QUALITY:**
  - $MovieQuality$**:** The variable represents quality of a movie because each movie is its source channel. The values of the variable are *low*, *average* and *high*.
  - $UserExperience$**:** As in the first experiment, this variable shows the experience level of a user. Its values are *novice*, *intermediate*, and *expert*.
- **Extension GENRE-Related EXPERTISE/ MOVIE QUALITY:**
  - $MovieQuality$**:** The variable and its values are the same as the movie quality variable and its values in the third experiment.

– $User(Genre_n)Experience$**:** As in the third experiment, there are multiple variables in order to define the expertise level of a user for a genre. The values of the variables are *novice*, *intermediate*, and *expert*.

4.1.2.2 Probability Table

The probability tables represent the conditional probabilities between the variables in a Bayesian network. According to our initial theory, one Bayesian network is created for one content item. In the first and third experiments, one Bayesian network is for each movie. However, in the genre extensions of the experiments, we try to calculate the genre expertise of the users. If a movie has multiple genres, we create a Bayesian network for each movie-genre pair.

Figure 4.1 shows the dependencies between the variables in each experiment. According to the figure, a rating value ($Rating$) is dependent on the quality of a movie ($MovieQuality$) or a director ($MovieDirectorQuality$) and general ($UserExperience$) or genre ($User(Genre_n)Experience$) experience of a user. $Genre_n$ represents one of the genres of the corresponding movie.

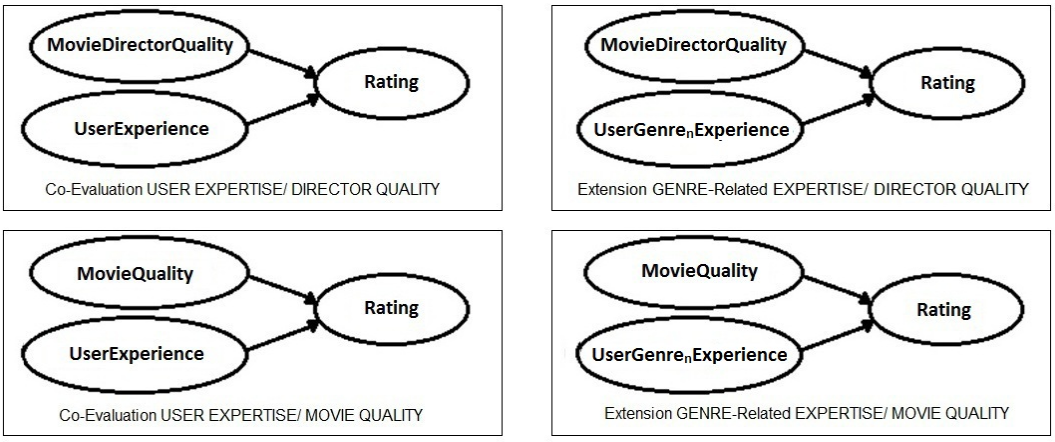


Figure 4.1: Structure of the Bayesian Networks in the experiments

The way of determining the conditional probabilities is explained in Section 3.3. According to that, expert users give ratings most accurately, the variance of the probability distribution is low for them. The ratings of novice users are

distributed in wide range which means the variance of the probability distribution is high.

We determined a set of conditional probabilities by using the way which is described above. In order to increase the performance of classifying users according to their expertise level, we create another set of conditional probability values by considering the behaviors of users in different conditions. Since the second set gives better results, we use it in the experiments.

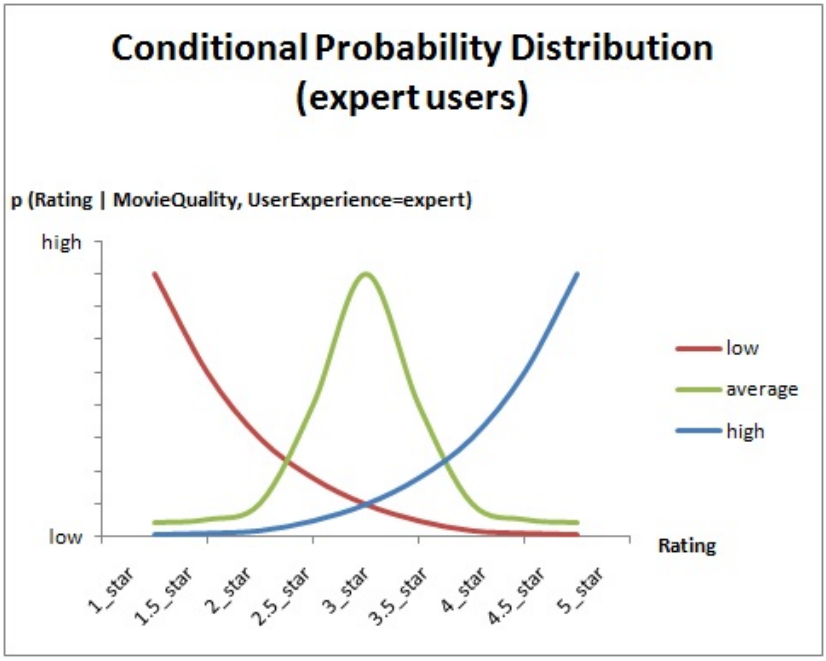


Figure 4.2: Conditional Probability Distribution Functions for Expert Users

In order to model the behaviors of users, we assume that there are two factors which affect the decision of a user about a movie.

- Understanding of users
- Expectation of users

Understanding of users is about whether the users are able to understand movies and the expectation is that how much the users expect from a movie. Our first estimation is that the expert users can generally understand movies, and they

have high expectations. So they tend to give high ratings for the high quality movies, average ratings for the average quality movies, and low ratings for the low quality movies. The intermediate users can also understand movies, but it is not as much as the understanding of an expert. Their expectations are also lower than the expectations of the expert users. So they tend to give high and average values for the high quality movies; low, average, and high ratings for the average quality movies; and low and average ratings for the low quality movies. Our final assumption is that the novice users can often understand the low and average quality movies, and they can rarely understand the high quality movies. Their expectations are generally very low. Because of the low expectations of the novice users, they tend to give low, average, and high ratings for the low quality movies; and average and high ratings for the average quality movies. The novice users tend to give low and high ratings to the high quality movies, because we assume that they generally cannot understand them and give low rating, or they somehow understand the movie and give high ratings. Here, we also assume that they do not give average ratings to the high quality movies.

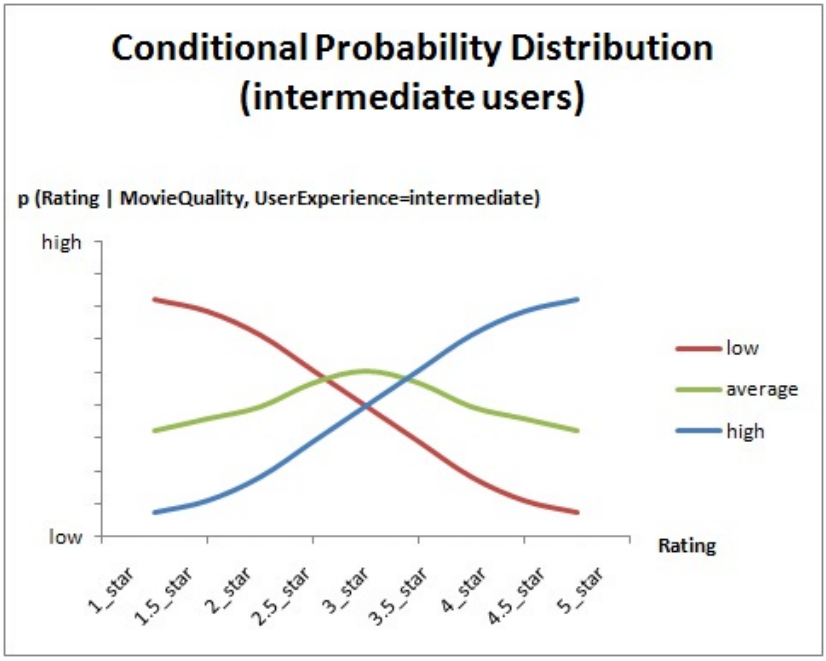


Figure 4.3: Conditional Probability Distribution Functions for Intermediate Users

In order to determine the conditional probability values, we created probability distribution functions by considering the criteria for different types

of users and movies. Figure 4.2 presents the conditional probability distribution function for expert users. The x-axis represents the values of the variable *Rating* and the legends represent the values of the variable *MovieQuality*. The y-axis of the chart represents the conditional probability value $p(Rating|MovieQuality, UserExperience = expert)$ which means the probability distribution of rating values for different quality level of movies and expert users. For example, red line represents the probability distribution of *Rating* when *MovieQuality* is *low* and *UserExperience* is expert. According to the distribution, the expert users tend to give low ratings for the low quality movies as we mentioned before. Figure 4.3 and Figure 4.4 present the conditional probability distributions for intermediate and novice users respectively.

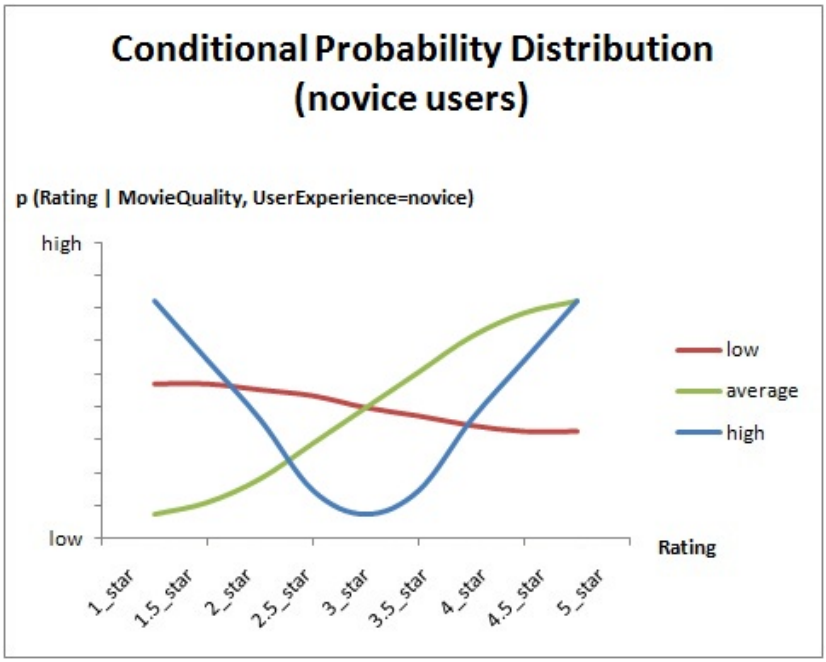


Figure 4.4: Conditional Probability Distribution Functions for Novice Users

By using the probability distribution functions, we determine conditional probability values for each condition. Table 4.1 represents the conditional probability values of a rating for different user and movie types. The first two columns of the table represent the conditions (values of the variables *MovieQuality* or *MovieDirectorQuality* and *UserExperience* or $UserGenre_nExperience$ respectively), and the other columns shows the probability distribution of *Rating* under these conditions.

MQ^1	UE^2	1 *star*	1.5 *star*	2 *star*	2.5 *star*	3 *star*	3.5 *star*	4 *star*	4.5 *star*	5 *star*
low	*nov*	0.130	0.130	0.120	0.120	0.110	0.110	0.100	0.090	0.090
low	*int*	0.200	0.180	0.160	0.140	0.120	0.100	0.050	0.030	0.020
low	*exp*	0.400	0.250	0.150	0.100	0.050	0.030	0.010	0.006	0.004
avg	*nov*	0.020	0.030	0.050	0.100	0.120	0.140	0.160	0.180	0.200
avg	*int*	0.100	0.110	0.110	0.120	0.120	0.120	0.110	0.110	0.100
avg	*exp*	0.020	0.030	0.050	0.200	0.400	0.200	0.050	0.030	0.020
high	*nov*	0.200	0.150	0.100	0.040	0.020	0.040	0.100	0.150	0.200
high	*int*	0.020	0.030	0.050	0.100	0.120	0.140	0.160	0.180	0.200
high	*exp*	0.004	0.006	0.010	0.030	0.050	0.100	0.150	0.250	0.400

[1] $Movie/DirectorQuality$
[2] $User/Genre_nExperience$

TABLE 4.1: CONDITIONAL PROBABILITY DISTRIBUTION VALUES

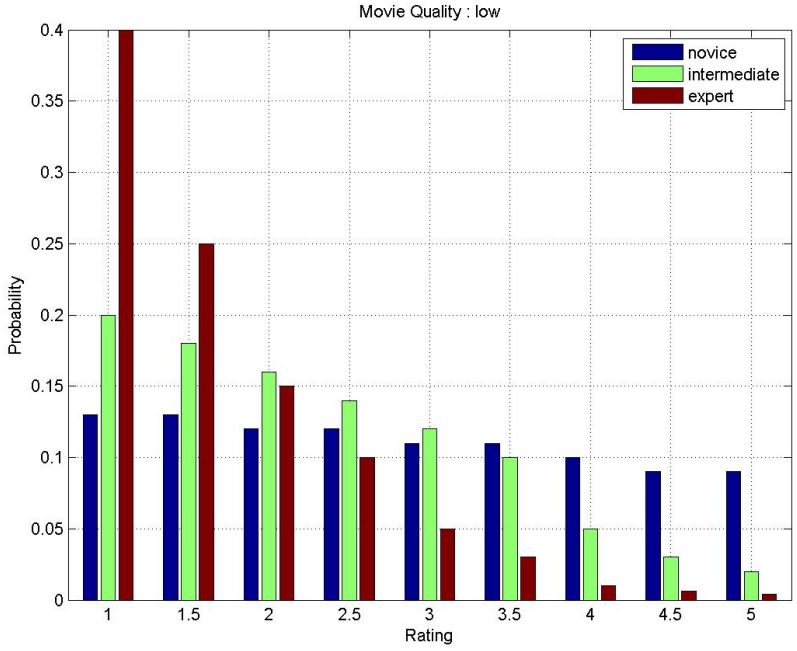


Figure 4.5: Conditional probability distribution (low quality movie/director)

Fig. 4.5, Fig. 4.6, and Fig. 4.7 represent the probability distribution charts of the ratings when the movie/director quality is low, average, and high respectively. The x-axis shows us the feedback values and the y-axis represents the probability of these values under the specified conditions. The chart legends *novice*, *intermediate*, and *expert* are the experience level of a user.

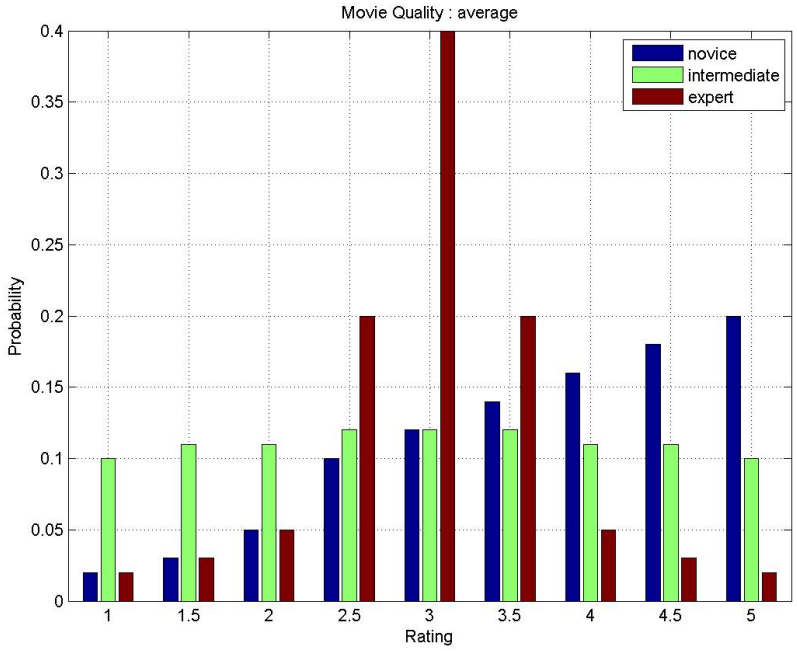


Figure 4.6: Conditional probability distribution (average quality movie/director)

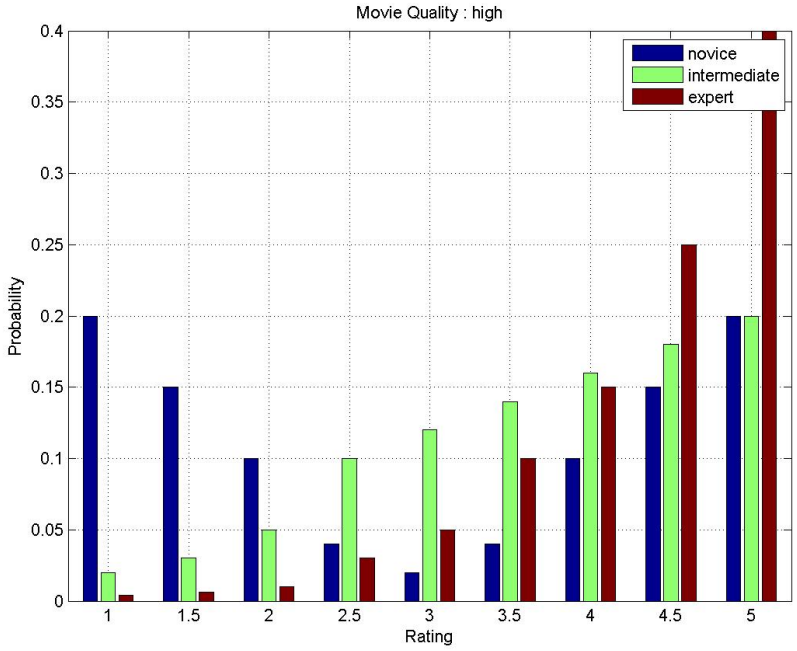


Figure 4.7: Conditional probability distribution (high quality movie/director)

4.1.2.3 Lifetime of Bayesian Networks

As we mentioned in Chapter 3, feedbacks are processed and one Bayesian network is created for each content item. After some amount of time passes, inference is done for parent variables and the relevant network is removed from the set. In this section, we determine the amount of time which is used as lifetime of Bayesian networks. Because of the computational limitation of Bayesian inference, we limit the maximum number of the target variables (user and content) in a network to ten.

In experiments, since our system creates one network for each movie, we analyze the number of the ratings submitted for a movie in a specific length of time. In Table 4.2, we see the information about how many ratings are submitted for how many movies in one month period.

Number of the Ratings	Number of the Movies
1 - 10	8477
11 - 20	937
21 - 30	396
31 - 40	191
41 - 50	130
51 - 60	103
61 - 70	62
71 - 80	38
81 - 90	39
91 - ...	25

Table 4.2: Number of Movies Grouped by Monthly Ratings

According to the table, less than or equal to ten ratings are submitted for 81% of movies in a month. So one month-length period is suitable for the experiments. For the remaining 19% of movies, we define a threshold value of ten variables.

4.1.3 Description of the Experiments

In the experiments, we split the data set and use randomly selected 8M of them to assess user and content qualities. Since the expert users have the ability to make correct judgments, their ratings should be more predictable if we know the correct qualities of movies. In order to evaluate the ability of our system to identify expert users, we use the remaining 2M ratings and try to predict the values of them. For each prediction, we calculate an *error* value by taking the difference between the actual and predicted values. If we can assess movie qualities correctly, low error values for expert users means that our system is also good at classifying users according to their expertise level.

4.1.3.1 Co-Evaluation USER EXPERTISE/ DIRECTOR QUALITY

In this experiment, we find the quality values of the variables $MovieDirectorQuality$ and $UserExperience$ for different times. In order to predict the value of a rating, we firstly check the time of the rating, and then take the probabilities of target variable values which are recorded most recently in the past time. If there is no such record, then we skip the rating. After we get the probabilities of the target variable values, we calculate the distribution of the rating by considering the conditional probability table which is defined before. Since we need a single value for prediction, we calculate the mean, median, and maximum likelihood of the distribution. An error value is recorded for each prediction type. Since we have to pick the value from the set {1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5}, we take the value which is closest to the calculated mean, median, or maximum likelihood respectively. The mean of a distribution is calculated by Equation 4.1. The values of a rating are {1, 1.5 ... 4.5, 5} and $P(Rating = value)$ represents the probability of $Rating$ being $value$.

$$mean = \sum_{value \in Rating} (P(Rating = value) \times value) \tag{4.1}$$

The median of a distribution is value which satisfies the condition in Equation 4.2.

$$P(X \leq median) \geq \frac{1}{2} \tag{4.2}$$

The maximum likelihood is the value whose probability value $P(Rating = value)$ is the maximum.

4.1.3.2 Extension GENRE-Related EXPERTISE/ DIRECTOR QUALITY

The main difference between this and the first experiment is that we calculate the expertise value of a user for each genre. In order to decrease the computational time, we decide to choose a subset of the genres which are represented in Section 4.1.1. First of all, we analyze the numbers of the movies which are associated with the genres. For each genre, we also calculate the number of the movies which are only related with that genre.

Genre	Number of All Movies	Number of Movies (only specified genre)
ACTION	**1485**	**79**
Adventure	1004	32
Animation	376	3
Children's	524	4
COMEDY	**3654**	**1035**
Crime	1104	25
DOCUMENTARY	**457**	**346**
DRAMA	**5277**	**1770**
Fantasy	531	1
Film-Noir	147	12
HORROR	**1000**	**260**
Musical	433	26
Mystery	502	6
Romance	1656	37
Sci-Fi	743	53
Thriller	1683	126
War	495	18
Western	270	91

Table 4.3: Number of Movies Grouped by Genres

According to Figure 4.3, the genres *comedy*, *drama*, *documentary*, and *horror*

have greatest numbers in third column. In order to lose less movies, we have to choose these four genres. The fifth genre which has the greatest value in third column is *thriller*. Since this genre is similar to the genre *horror* (the number of common movies for these genres is high), we skip it and take the genre *action* in order to distinguish the genres better.

In the quality assessment part, we create a Bayesian Network for each movie-genre pair. Since most of the movies have more than one genre, we generally have multiple networks for them. At the end, we have the quality values for directors and user-genre pairs. In the prediction part, if a movie has more than one genre, we do the prediction and error calculation steps - which are explained in the first experiment - for each genre.

4.1.3.3 Co-Evaluation USER EXPERTISE/ MOVIE QUALITY

This experiment is very similar to the first experiment. Here, we use the variable $MovieQuality$ instead of $MovieDirectorQuality$.

4.1.3.4 Extension GENRE-Related EXPERTISE/ MOVIE QUALITY

The description of this experiment is similar to the description of the second experiment. However, the quality of a movie is considered instead of the quality of a director.

4.2 Experimental Results

In this section, we present the results in order to evaluate the feasibility of our approach and the ability of the system to identify expert users. In the first part, we compare our system with a classical approach in order to measure the success of assessing movie qualities. If our system succeeds in determining movie

qualities, we evaluate the performance of the prediction according to expertise level of users. Since the experts are good at evaluating movies, more successful prediction for experts means that we are able to classify the users according to their expertise level.

4.2.1 Comparison with Classical Approach

In order to check whether our co-evaluation approach makes improvement on the classical approach, we evaluate the performance of them by using the data in IMDb. We consider the list of top 250 movies in IMDb [9] as gold standard for the high quality movies. After we associate the movies in IMDb and MovieLens, we have 228 top quality movies because some of the movies are not included in MovieLens.

In order to start evaluation, we ordered the movies according to the quality values which are obtained as an output of co-evaluation software. We configured the software with the same parameters which are used in experiment 3. After that, we take the first 228 movies and the movies which have the same quality values with the 228th. Finally we calculate the precision, recall, and F-measure values which are calculated by the equations below.

$$precision = \frac{|\{TopMovies\} \bigcap \{RetrievedMovies\}|}{|\{RetrievedMovies\}|} \tag{4.3}$$

$$recall = \frac{|\{TopMovies\} \bigcap \{RetrievedMovies\}|}{|\{TopMovies\}|} \tag{4.4}$$

$$F = 2\frac{precision \times recall}{precision + recall} \tag{4.5}$$

As a classical approach, we calculate the quality value of a movie by taking the average of the ratings which are submitted for the relevant movie. After that, we ordered the movies according to the average rating values. As we did in our approach, we take the best 228 movies and the movies which have the same average rating values with the 228th.

In MovieLens data set, some of the movies have very low number of ratings, so the reason of the difference between the performance of our approach and the classical approach can be because of both of the learning power of Bayesian networks (the movies should be voted by many users in order to be determined as high quality if Bayesian approach is used) and the concept of co-evaluation (trust of the user is also included in process). In order to observe the effect of co-evaluation, we filtered the movies and take those which have at least 1000 ratings and apply classical approach for them. Decrease or increase in the threshold value (1000) causes a decrease in precision.

	Classical	Classical with Filter	Co-Evaluation
Precision	0.4912	0.5482	0.6214
Recall	0.4912	0.5482	0.6623
F-measure	0.2456	0.2741	0.3206

Table 4.4: Precision, Recall and F-Measure Values per Approach

The precision, recall, and F-measure values of three approaches are presented in Table 4.4. According to these values, Bayesian co-evaluation approach overrides the classical approach although additional filtering is applied. These results prove that our system is good at assessing movie qualities.

4.2.2 Ability of the System to Identify Experts

In this section, we evaluate the ability of our system to classify expert users. In order to measure the performance of prediction, we group the users according to their expertise level and calculate average error values for each prediction type. The average error values are calculated by taking the averages of the prediction errors. At the beginning, we consider the correlation between the average errors and the user expertise at a given time. The expertise level of a user at a given time is calculated based on the values of *UserExperience* variable in time when the corresponding rating is given. We also calculate the expertise level of a user at the end in order to check whether our system identifies the experts better after processing all of the ratings. Since there is no major difference between those two

types of results, we just present the charts for the expertise at given time.

The value of *expertise* represents the general or genre experience according to the experiment. It is calculated by the following equation:

$$expertise = (P(intermediate) + 2 \times P(expert))/2 \tag{4.6}$$

In the formula, *P(intermediate)/P(expert)* represents the probability that a user is intermediate/expert. The *expertise* value lies in the range [0,1].

The format of the data and the charts for each experiment are given below. The data which is used to draw charts are presented in Appendix A.

4.2.2.1 Co-Evaluation USER EXPERTISE/ DIRECTOR QUALITY

- Validation

  Timestamp::UserID::MovieID::MeanError::MedianError::MlError

- User

  Timestamp::UserID::Novice::Intermediate::Expert

- Movie MovieID::DirectorID

Validation table holds the information about the success of the prediction of a rating submitted by a user (*UserID*) for a movie (*MovieID*). The columns *MeanError*, *MedianError*, and *MlError* represent the difference between the actual value and the estimated value which is found by mean, median, or maximum likelihood prediction respectively.

We create a user table in order to keep the information about the expertise level of users for different times. The table is used to calculate *expertise* values defined in Equation 4.6.

Movie table is created to associate a movie to its director because we consider directors as source channel of the movies in this experiment.

Figure 4.8 shows the relation between the average error values and user expertise. The y-axis represents the average error values and the x-axis represents the expertise level of users. The legends show the average error values for different prediction types. Here, expertise is calculated based on the values of $UserExperience$ variable in time when rating is given.

The data which is used to draw the chart is presented in Table A.1. The users are grouped according to their expertise values (0.00, 0.05, 0.10 ... 0.95, 1.00), and the numbers of the users per group are given in the fifth column of the table.

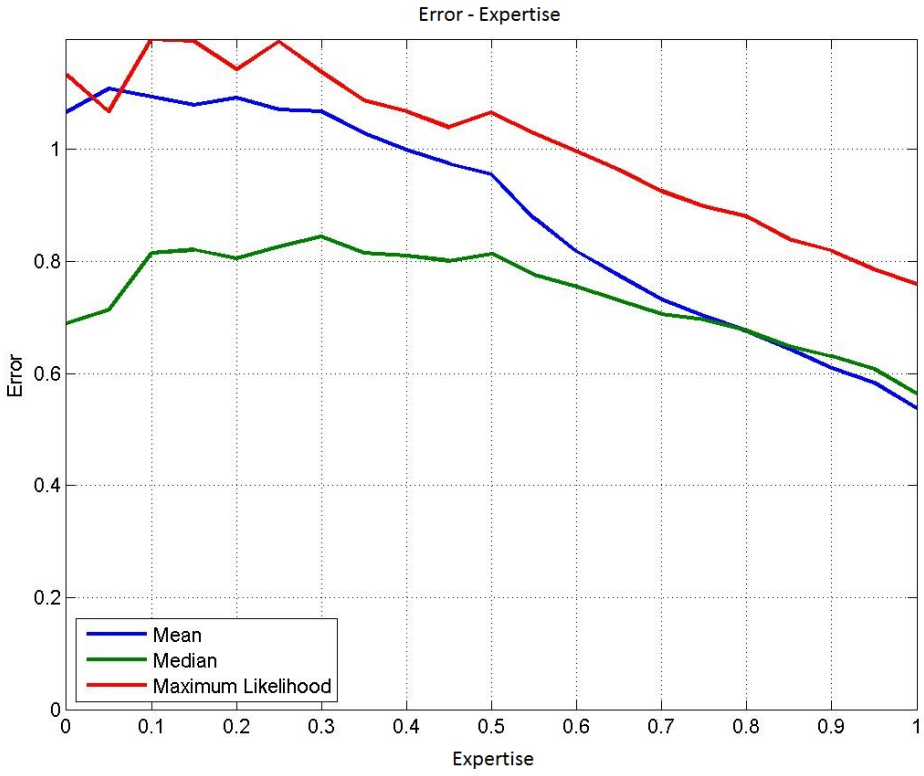


Figure 4.8: Change in Average Errors According to Expertise

According to the figure, we make better predictions for expert users in all type of prediction ways. In generally, median prediction performs better but after the expertise level 0.8, mean average error is the least. The graph also shows that maximum likelihood prediction performs worst for all of the expertise level.

4.2.2.2 Extension GENRE-Related EXPERTISE/ DIRECTOR QUALITY

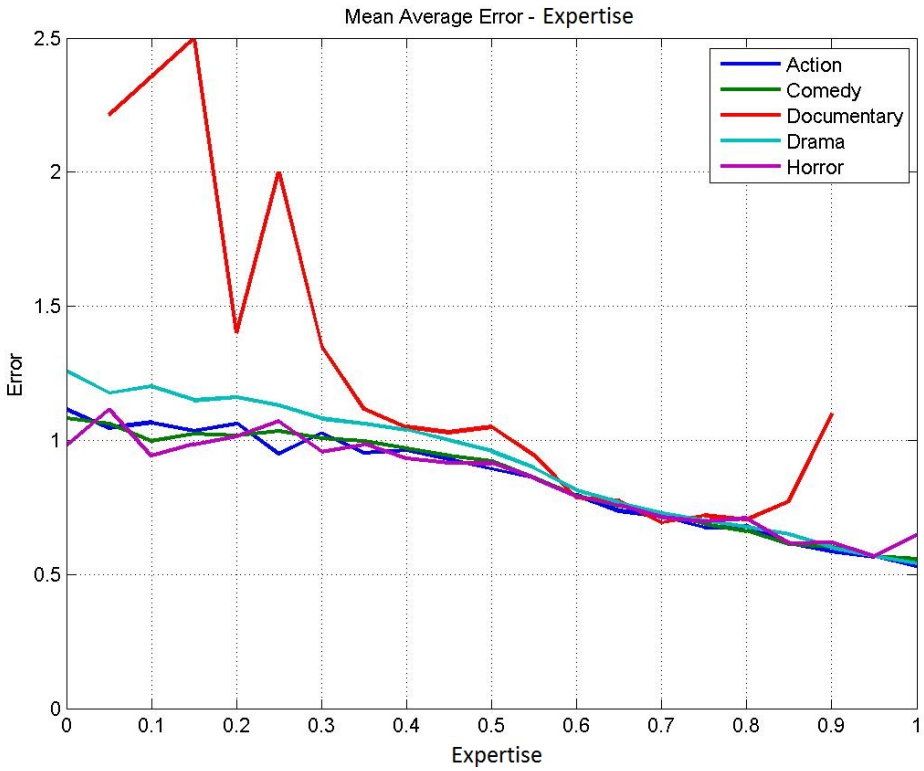


Figure 4.9: Change in Mean Average Error According to Expertise

- Validation

  Timestamp::UserID::Genre::MovieID::MeanError::MedianError::MlError

- User

  Timestamp::UserID::Genre::Novice::Intermediate::Expert

- Movie MovieID::DirectorID

For this experiment, validation table keeps the information about the success of the prediction which is done based on genre-experiences instead of general experiences. Each prediction is recorded according to *UserID*, *Genre*, and *MovieID*

information. The columns *MeanError*, *MedianError*, and *MlError* represent the error values as in the first experiment.

User table holds the information about the quality level of users according to genres for different times. The table is used to calculate *expertise* values defined in Equation 4.6 for each genre.

Movie table is created for the same purpose with the first experiment.

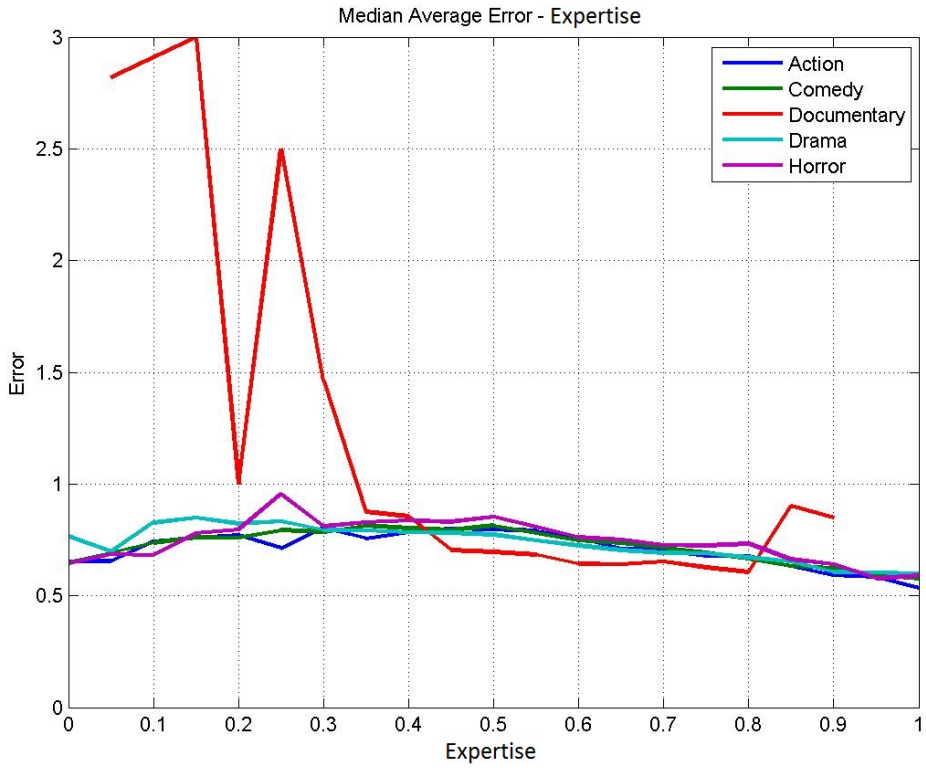


Figure 4.10: Change in Median Average Error According to Expertise

Figures 4.9, 4.10, and 4.11 show the relation between the mean / median / maximum likelihood average errors and expertise. Here, the order of successes of prediction types is the same as the order in the previous experiment. The performance of the prediction for the genres *action*, *comedy* and *horror* are almost similar. However, the success of the prediction for *drama* movies is lower for the maximum likelihood prediction and novice users. It means that a user should be more expert in order to make accurate decision about the *drama* movies. If

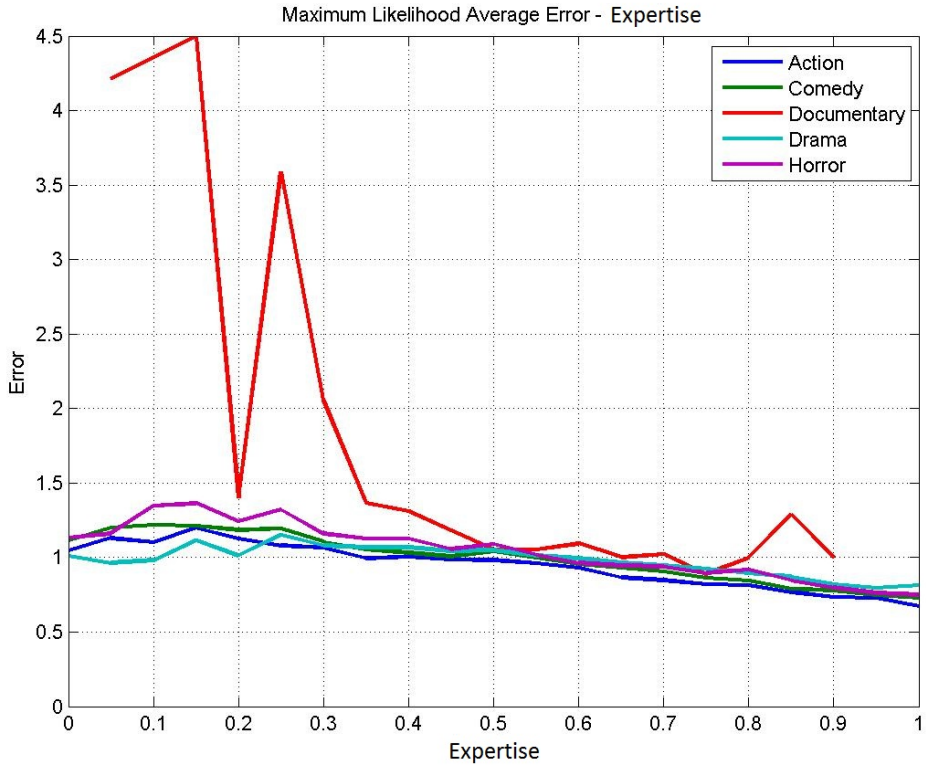


Figure 4.11: Change in Maximum Likelihood Average Error According to Expertise

we consider the *documentary* movies, we come to the similar conclusion with the *drama* movies. For the median prediction - whose accuracy is the highest - the performance is better for the *documentary* movies and the expert users. It means that the experts in this area have the ability to make more correct judgments. The reason of rises and falls in the curves for *documentary* is that the number of the ratings given to this type of movies is not adequately high.

4.2.2.3 Co-Evaluation USER EXPERTISE/ MOVIE QUALITY

- Validation

  Timestamp::UserID::MovieID::MeanError::MedianError::MlError

- User

  Timestamp::UserID::Novice::Intermediate::Expert

In this experiment, the format of the result data is almost similar to the format in the first experiment, but we do not have a movie table because movies are source channels instead of directors.

Figure 4.12 presents the change in average error values with respect to the expertise level of the users. The results are similar with the results in the first experiment.

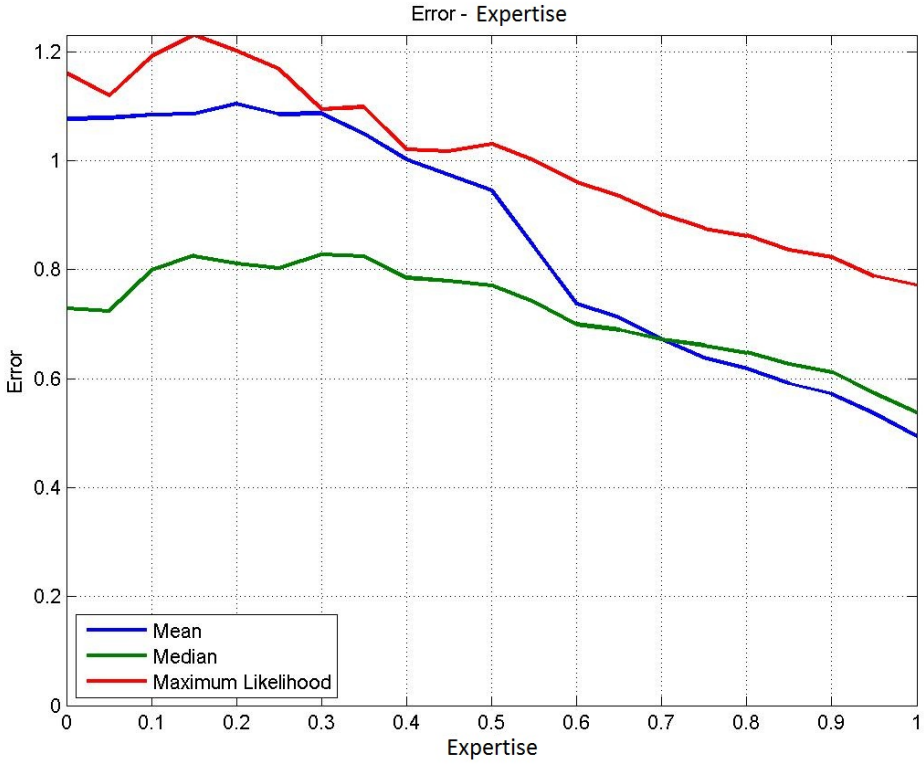


Figure 4.12: Change in Average Errors According to Expertise

4.2.2.4 Extension GENRE-Related EXPERTISE/ MOVIE QUALITY

- Validation

  Timestamp::UserID::Genre::MovieID::MeanError::MedianError::MlError

- User

Timestamp::UserID::Genre::Novice::Intermediate::Expert

The format of the data is like the format in the second experiment, but there is no movie table because of the same reason which is mentioned in the details of the third experiment.

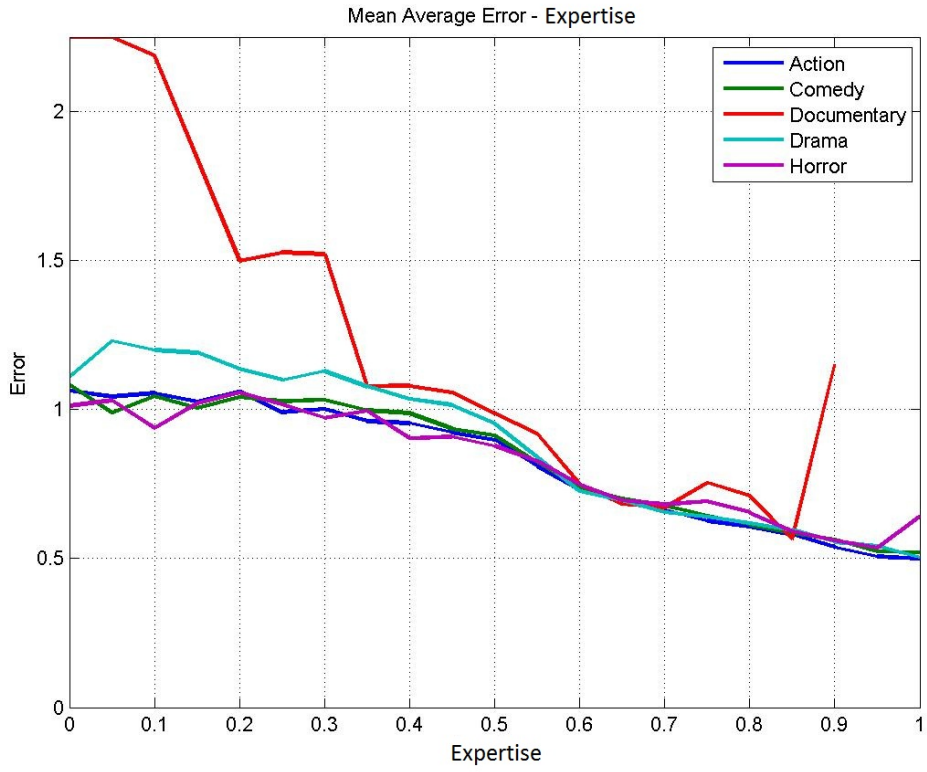


Figure 4.13: Change in Mean Average Error According to Expertise

Figures 4.13, 4.14, and 4.15 represent the change in average error values according to the expertise level of the users. The results are similar with the results in the third experiment.

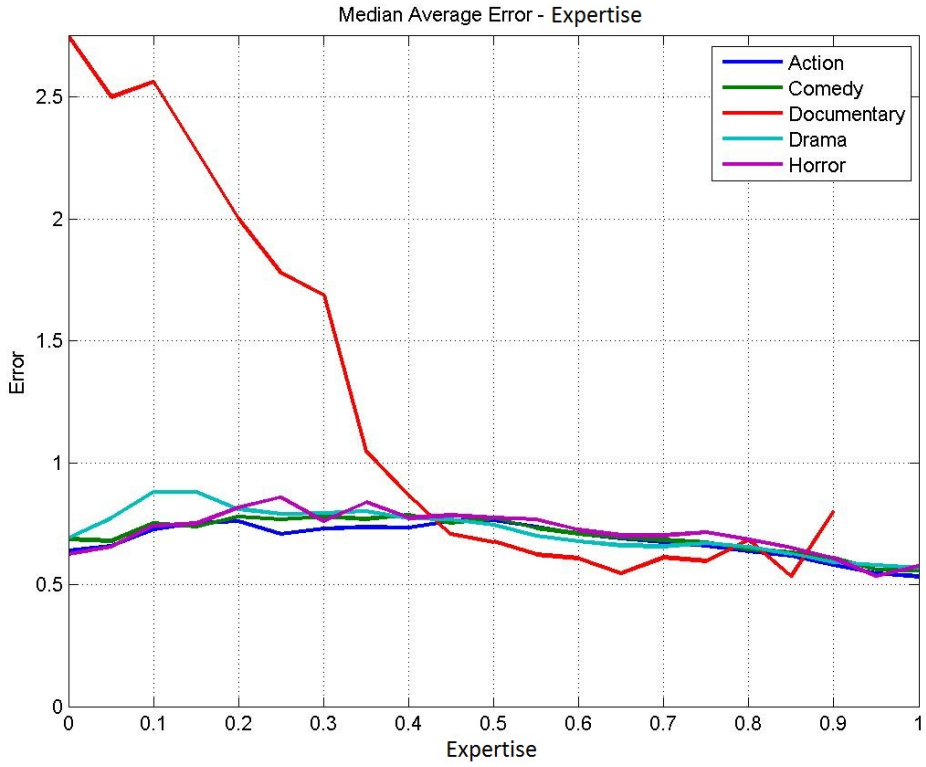


Figure 4.14: Change in Median Average Error According to Expertise

According to the results of the experiments, the prediction performance is significantly higher for expert users. These results show that our system has the ability to determine the user expertise level.

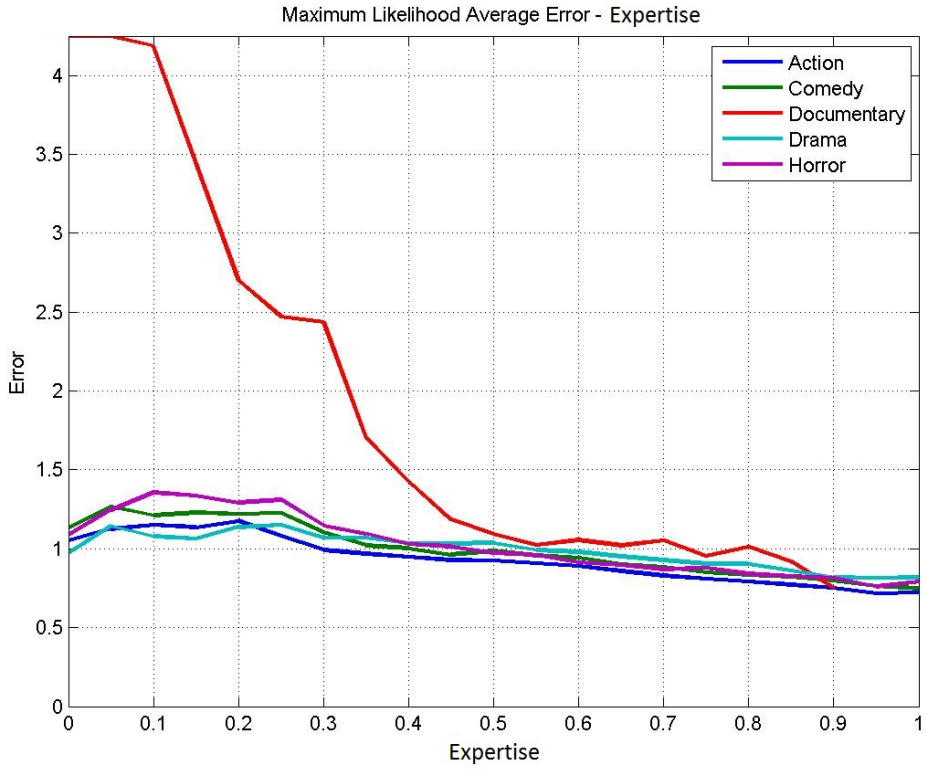


Figure 4.15: Change in Maximum Likelihood Average Error According to Expertise

4.2.3 Comparison between Experiments

In this section, we present the average error values for each experiment in order to compare the prediction performances. The values are given in Table 4.5.

According to the values, median is the best way to determine a single rating by using the probability distribution of rating values. Determining the quality of movies instead of the quality of directors makes significant improvement in the prediction. The genre extensions of the experiments do not cause significant improvement in the performance of the prediction.

Experiment	Mean Avg. Err.	Median Avg. Err.	ML.[1] Avg. Err.
1	0.8407	0.7500	0.9910
2	0.8455	0.7456	0.9775
3	0.7842	0.7105	0.9587
4	0.7938	0.7073	0.9482

[1]Maximum likelihood

Table 4.5: Average Error Values per Experiment

4.3 Evaluation

In this section, we evaluate the results which are given in Section 4.2.

In order to indicate the feasibility of the study, we compare our methodology with a classical approach which is used to find the quality of the movies. According to the result of that comparison, Bayesian co-evaluation approach causes a significant improvement in determining the quality of the movies.

Since our system has the ability to detect the high quality movies, we are able to measure the capability of it to classify the users according to their expertise level. Since the expert users evaluate movies more accurately, the users whose ratings are predicted precisely are defined as experts in our system. Since there is a positive correlation between the prediction performance and expertise level, our system is able to identify the expert users correctly.

According to the comparison between the experiments, the idea of using director information as source channel decreases the performance in the domain the evaluation of movies. In addition, genre extensions do not cause a significant improvement in predictions. Our final analyze is that the median prediction is the best way to estimate a single value from a distribution.

Chapter 5

Conclusions

As mentioned, the World Wide Web is the source of enormous amounts of information. As a result, it can be time-consuming to locate relevant information. In this thesis, we study the problem of determining the content qualities on the web. We provide the implementation of a novel approach *Bayesian Co-Evaluation*, which is built upon the idea of evaluation of contents and users together. The experimental results show that this methodology outperforms a classical approach in assessment of movie qualities.

5.1 Thesis Summary

Recommender systems are introduced in order to suggest the relevant information to the users on the internet. The collaborative filtering method is a well-known implementation of recommender systems, which considers the interests of users. In this kind of system, a recommendation is made based on the similarity of users. The concept of trust is defined because of the problem that a user may not be trustworthy in all categories. According to this approach, the trust level of users should be included in the suggestion process because it is not enough to consider user similarities alone. The co-evaluation approach, which contributes to the theory part of this thesis, brings a new perspective to the trust based

systems by including a Bayesian approach, and the concept of change in quality over time.

In the experiments, we compared the retrieval performance of our system with that of a classical approach in order to demonstrate the feasibility of the methodology. The results show that this system performs better than the classical approach in the assessment of the qualities of different movies .

In view of the success of the system, we performed additional experiments in order to measure the ability of our system to detect the expert users. We assessed the movie and user qualities over time and used those quality values in order to predict the values of the other ratings. Since this approach was successful in assessing movie qualities, the users with the greater ability to make correct judgments about the movies were those who submitted the ratings which we predicted well. In other words, the prediction performance of the system should be better for the expert users, and the results show that the system also has the ability to identify the expert users.

5.2 Contributions and Future Work

In this thesis, we designed and implemented software for the co-evaluation of user and content qualities through a Bayesian approach. We tested the system using the ratings submitted to the movies in the MovieLens data set. The results show that Bayesian co-evaluation is successful in assessing the quality of movies, and classifying users according to their expertise level.

The major contributions of this study are as follows:

- The theory of the Bayesian co-evaluation of the user and content qualities is improved
- The algorithm of the Bayesian co-evaluation process is defined, and the first version of the software is implemented.

- The feasibility and the performance of the methodology are tested using the MovieLens data set.

The research described in this thesis can be extended in the following directions.

- In order to increase the prediction performance, the MovieLens experiments can be performed with different parameters (different variables and models).
- Since our software is configurable, the methodology can be applied to the other static rating data sets.
- The aim of the software is online processing of the data created by the collaborators on the internet. The online processing can be applied to the contents in the systems such as Google+, Wikipedia, Facebook, etc...
- Google [6] aims to create a voting mechanism for all of the contents on the web through their project Google+. Our software can be used to assess the qualities of the contents if explicit feedback is available.

Bibliography

[1] Amazon. "http://www.amazon.com".

[2] Blogger. "http://www.blogger.com/".

[3] ebay. "http://www.ebay.com".

[4] Epinions.com. "http://www.epinions.com/".

[5] Facebook. "http://www.facebook.com/".

[6] Google. "http://www.google.com".

[7] The google+ project. "https://plus.google.com/".

[8] The internet movie database (imdb). "http://www.imdb.com/".

[9] The internet movie database (imdb) top 250. "http://www.imdb.com/chart/top".

[10] Movielens data sets. "http://www.grouplens.org/node/73/".

[11] Twitter. "http://twitter.com/".

[12] Wikipedia, the free encyclopedia. "http://www.wikipedia.org/".

[13] A. Abdul-Rahman and S. Hailes. A distributed trust model. In *Proceedings of the 1997 workshop on New security paradigms*, NSPW '97, pages 48–60, New York, NY, USA, 1997. ACM.

[14] T. Bayes and M. Price. An essay towards solving a problem in the doctrine of chances. *Philosophical Transactions of the Royal Society of London*, 53:370–418, 1763.

[15] T. Berners-Lee, R. Cailliau, J.-F. Groff, and B. Pollermann. World-wide web: The information universe. *Internet Research*, 2:52–58, 1992.

[16] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Commun. ACM*, 35:61–70, December 1992.

[17] K. Ivanov. *Quality-Control of Information: On the concept of accuracy of information in data-banks and in management information systems.* PhD thesis, The Royal Institute of Technology KTH, Stockholm, 1972.

[18] F. V. Jensen and T. D. Nielsen. *Bayesian Networks and Decision Graphs.* Information Science and Statistics series. Springer-Verlag, New York, 2 edition, June 2007.

[19] C. Moraga, M. A. Moraga, C. Calero, and A. Caro. Towards the discovery of data quality attributes for web portals. In *Proceedings of the 9th International Conference on Web Engineering*, ICWE '9, pages 251–259, Berlin, Heidelberg, 2009. Springer-Verlag.

[20] J. O'Donovan and B. Smyth. Trust in recommender systems. In *Proceedings of the 10th international conference on Intelligent user interfaces*, IUI '05, pages 167–174, New York, NY, USA, 2005. ACM.

[21] T. O'Reilly. What is web 2.0: Design patterns and business models for the next generation of software. *Communications and Strategies*, 1:17, 2007.

[22] M. Pazzani and D. Billsus. Content-based recommendation systems. In P. Brusilovsky, A. Kobsa, and W. Nejdl, editors, *The Adaptive Web*, volume 4321 of *Lecture Notes in Computer Science*, pages 325–341. Springer Berlin / Heidelberg, 2007.

[23] P. Resnick and H. R. Varian. Recommender systems. *Commun. ACM*, 40:56–58, March 1997.

[24] S. Y. Rieh. Judgment of information quality and cognitive authority in the web. *Journal of the American Society for Information Science and Technology*, 53(2):145–161, 2002.

[25] M. Schaal. A bayesian approach for small information trust updates. In *Proceedings of IeCCS*, 2006.

[26] D. M. Strong, Y. W. Lee, and R. Y. Wang. Data quality in context. *Commun. ACM*, 40:103–110, May 1997.

[27] R. Y. Wang and D. M. Strong. Beyond accuracy: What data quality means to data consumers. *Journal of Management Information Systems*, 12:5–34, 1996.

Appendix A

Experimental Results Data

Expertise	Mean Error	Median Error	ML Error	Number
0.0000	1.0672	0.6893	1.1333	2055
0.0500	1.1087	0.7142	1.0671	2502
0.1000	1.0947	0.8151	1.1974	3283
0.1500	1.0799	0.8213	1.1936	3655
0.2000	1.0924	0.8054	1.1428	4964
0.2500	1.0714	0.8268	1.1930	8030
0.3000	1.0677	0.8446	1.1382	11410
0.3500	1.0295	0.8152	1.0875	17228
0.4000	0.9995	0.8104	1.0683	24038
0.4500	0.9741	0.8014	1.0395	39915
0.5000	0.9552	0.8140	1.0657	56983
0.5500	0.8771	0.7760	1.0292	72058
0.6000	0.8179	0.7543	0.9962	70857
0.6500	0.7747	0.7305	0.9642	60225
0.7000	0.7321	0.7059	0.9253	46851
0.7500	0.7030	0.6958	0.8981	34238
0.8000	0.6758	0.6757	0.8801	25516
0.8500	0.6448	0.6487	0.8395	17477
0.9000	0.6099	0.6300	0.8190	12688
0.9500	0.5833	0.6083	0.7856	8963
1.0000	0.5375	0.5636	0.7602	6048

Table A.1: Experiment 1: Average Errors - Expertise

Expertise	Mean Error	Median Error	ML Error	Number	Genre
0.0000	1.1159	0.6522	1.0453	276	Action
0.0500	1.0462	0.6558	1.1304	552	Action
0.1000	1.0674	0.7438	1.1026	853	Action
0.1500	1.0370	0.7592	1.1985	947	Action
0.2000	1.0640	0.7713	1.1268	1305	Action
0.2500	0.9489	0.7124	1.0818	1907	Action
0.3000	1.0264	0.8039	1.0649	3082	Action
0.3500	0.9534	0.7571	0.9939	4107	Action
0.4000	0.9626	0.7853	1.0027	6662	Action
0.4500	0.9293	0.7992	0.9860	9461	Action
0.5000	0.8936	0.7977	0.9823	13569	Action
0.5500	0.8590	0.7902	0.9600	16137	Action
0.6000	0.7940	0.7578	0.9291	13812	Action
0.6500	0.7369	0.7106	0.8665	10412	Action
0.7000	0.7166	0.7050	0.8464	8184	Action
0.7500	0.6759	0.6782	0.8202	5904	Action
0.8000	0.6788	0.6768	0.8129	4273	Action
0.8500	0.6148	0.6301	0.7685	3201	Action
0.9000	0.5864	0.5912	0.7333	2171	Action
0.9500	0.5654	0.5827	0.7268	1742	Action
1.0000	0.5312	0.5341	0.6769	1201	Action
0.0000	1.0848	0.6481	1.1096	584	Comedy
0.0500	1.0611	0.6921	1.1986	924	Comedy
0.1000	0.9978	0.7361	1.2188	1131	Comedy
0.1500	1.0252	0.7612	1.2112	1610	Comedy
0.2000	1.0180	0.7589	1.1841	2026	Comedy
0.2500	1.0342	0.7936	1.1940	3060	Comedy
0.3000	1.0071	0.7851	1.1054	4533	Comedy
0.3500	0.9964	0.8138	1.0522	7314	Comedy
0.4000	0.9701	0.8017	1.0325	11288	Comedy
0.4500	0.9424	0.7933	1.0077	16886	Comedy
0.5000	0.9217	0.8135	1.0392	24619	Comedy
0.5500	0.8617	0.7787	0.9962	28610	Comedy
0.6000	0.7930	0.7494	0.9555	26265	Comedy
0.6500	0.7551	0.7338	0.9332	19880	Comedy
0.7000	0.7254	0.7150	0.9062	14324	Comedy
0.7500	0.6886	0.6914	0.8639	9908	Comedy
0.8000	0.6620	0.6655	0.8443	7292	Comedy
0.8500	0.6123	0.6331	0.7896	5058	Comedy
0.9000	0.6038	0.6249	0.7815	3748	Comedy
0.9500	0.5687	0.5922	0.7509	2228	Comedy
1.0000	0.5566	0.5779	0.7289	1944	Comedy
0.0500	2.2143	2.8214	4.2143	14	Documentary
0.1500	2.5000	3.0000	4.5000	1	Documentary
0.2000	1.4000	1.0000	1.4000	5	Documentary
0.2500	2.0000	2.5000	3.5882	17	Documentary
0.3000	1.3514	1.4595	2.0541	37	Documentary
0.3500	1.1154	0.8750	1.3654	52	Documentary
0.4000	1.0494	0.8547	1.3110	172	Documentary
0.4500	1.0297	0.7039	1.1855	488	Documentary
0.5000	1.0496	0.6950	1.0539	1159	Documentary
0.5500	0.9433	0.6834	1.0533	1737	Documentary
0.6000	0.7853	0.6432	1.0932	971	Documentary
0.6500	0.7733	0.6410	1.0000	525	Documentary
0.7000	0.6931	0.6525	1.0212	259	Documentary
0.7500	0.7203	0.6271	0.8898	118	Documentary
0.8000	0.7049	0.6066	1.0000	61	Documentary
0.8500	0.7742	0.9032	1.2903	31	Documentary
0.9000	1.1000	0.8500	1.0000	10	Documentary

Table A.2: Experiment 2: Average Errors - Expertise (Part-1)

Expertise	Mean Error	Median Error	ML Error	Number	Genre
0.0000	1.2589	0.7667	1.0145	448	Drama
0.0500	1.1768	0.6974	0.9629	727	Drama
0.1000	1.2011	0.8271	0.9814	833	Drama
0.1500	1.1493	0.8498	1.1160	1172	Drama
0.2000	1.1605	0.8227	1.0140	1720	Drama
0.2500	1.1299	0.8330	1.1503	2772	Drama
0.3000	1.0808	0.7947	1.0794	4580	Drama
0.3500	1.0621	0.7912	1.0690	8005	Drama
0.4000	1.0394	0.7833	1.0693	11947	Drama
0.4500	1.0004	0.7822	1.0443	18912	Drama
0.5000	0.9583	0.7739	1.0491	28939	Drama
0.5500	0.8974	0.7472	1.0130	35195	Drama
0.6000	0.8130	0.7242	0.9948	31310	Drama
0.6500	0.7677	0.7034	0.9655	25675	Drama
0.7000	0.7273	0.6908	0.9473	17955	Drama
0.7500	0.7014	0.6899	0.9249	12689	Drama
0.8000	0.6765	0.6720	0.8938	8777	Drama
0.8500	0.6498	0.6531	0.8731	6140	Drama
0.9000	0.6002	0.6066	0.8205	3889	Drama
0.9500	0.5693	0.6029	0.7950	2498	Drama
1.0000	0.5404	0.5969	0.8143	1548	Drama
0.0000	0.9778	0.6444	1.1333	45	Horror
0.0500	1.1154	0.6846	1.1615	65	Horror
0.1000	0.9412	0.6814	1.3480	102	Horror
0.1500	0.9846	0.7797	1.3634	227	Horror
0.2000	1.0144	0.7950	1.2428	278	Horror
0.2500	1.0700	0.9544	1.3205	493	Horror
0.3000	0.9578	0.8094	1.1588	913	Horror
0.3500	0.9850	0.8294	1.1278	1436	Horror
0.4000	0.9318	0.8370	1.1267	2162	Horror
0.4500	0.9156	0.8307	1.0573	3751	Horror
0.5000	0.9143	0.8522	1.0893	5207	Horror
0.5500	0.8590	0.8076	1.0193	5920	Horror
0.6000	0.7907	0.7607	0.9645	4304	Horror
0.6500	0.7564	0.7504	0.9466	2919	Horror
0.7000	0.7134	0.7270	0.9390	1804	Horror
0.7500	0.6967	0.7240	0.8953	1098	Horror
0.8000	0.7087	0.7323	0.9168	721	Horror
0.8500	0.6153	0.6638	0.8481	464	Horror
0.9000	0.6182	0.6406	0.7971	313	Horror
0.9500	0.5675	0.5775	0.7625	200	Horror
1.0000	0.6466	0.5905	0.7500	116	Horror

Table A.3: Experiment 2: Average Errors - Expertise (Part-2)

Expertise	Mean Error	Median Error	ML Error	Number
0.0000	1.0770	0.7286	1.1616	1590
0.0500	1.0788	0.7239	1.1192	1909
0.1000	1.0846	0.7997	1.1920	2596
0.1500	1.0857	0.8254	1.2306	3139
0.2000	1.1043	0.8112	1.2022	3715
0.2500	1.0855	0.8029	1.1686	6305
0.3000	1.0865	0.8282	1.0941	8720
0.3500	1.0489	0.8242	1.0989	13541
0.4000	1.0023	0.7849	1.0210	19982
0.4500	0.9739	0.7794	1.0174	30334
0.5000	0.9450	0.7711	1.0309	51599
0.5500	0.8413	0.7404	1.0006	67445
0.6000	0.7367	0.6996	0.9602	70674
0.6500	0.7116	0.6892	0.9352	63208
0.7000	0.6717	0.6715	0.9008	49367
0.7500	0.6385	0.6610	0.8755	39278
0.8000	0.6181	0.6475	0.8622	28667
0.8500	0.5906	0.6267	0.8361	18510
0.9000	0.5722	0.6126	0.8238	14163
0.9500	0.5361	0.5731	0.7881	8773
1.0000	0.4943	0.5388	0.7716	5502

Table A.4: Experiment 3: Average Errors - Expertise

Expertise	Mean Error	Median Error	ML Error	Number	Genre
0.0000	1.0635	0.6369	1.0496	252	Action
0.0500	1.0436	0.6589	1.1260	516	Action
0.1000	1.0546	0.7254	1.1521	723	Action
0.1500	1.0280	0.7523	1.1367	874	Action
0.2000	1.0601	0.7594	1.1768	1114	Action
0.2500	0.9906	0.7070	1.0803	1657	Action
0.3000	1.0012	0.7291	0.9916	2554	Action
0.3500	0.9598	0.7345	0.9695	3705	Action
0.4000	0.9546	0.7333	0.9493	5767	Action
0.4500	0.9230	0.7612	0.9276	8572	Action
0.5000	0.8996	0.7636	0.9237	12187	Action
0.5500	0.8094	0.7370	0.9089	14883	Action
0.6000	0.7298	0.7075	0.8895	13591	Action
0.6500	0.6898	0.6908	0.8584	11236	Action
0.7000	0.6623	0.6751	0.8282	8157	Action
0.7500	0.6264	0.6583	0.8106	6451	Action
0.8000	0.6069	0.6350	0.7917	4188	Action
0.8500	0.5805	0.6174	0.7709	3204	Action
0.9000	0.5378	0.5796	0.7503	2211	Action
0.9500	0.5064	0.5458	0.7170	1714	Action
1.0000	0.4995	0.5323	0.7226	1022	Action
0.0000	1.0830	0.6861	1.1317	524	Comedy
0.0500	0.9895	0.6788	1.2679	713	Comedy
0.1000	1.0437	0.7527	1.2120	1132	Comedy
0.1500	1.0030	0.7380	1.2278	1328	Comedy
0.2000	1.0407	0.7775	1.2214	1890	Comedy
0.2500	1.0284	0.7667	1.2258	2553	Comedy
0.3000	1.0320	0.7775	1.1042	3422	Comedy
0.3500	0.9969	0.7692	1.0199	6066	Comedy
0.4000	0.9883	0.7855	1.0019	9967	Comedy
0.4500	0.9359	0.7535	0.9624	14629	Comedy
0.5000	0.9133	0.7722	0.9870	22327	Comedy
0.5500	0.8223	0.7310	0.9564	27007	Comedy
0.6000	0.7366	0.7091	0.9432	26537	Comedy
0.6500	0.7031	0.6917	0.9014	20512	Comedy
0.7000	0.6761	0.6837	0.8845	15365	Comedy
0.7500	0.6444	0.6725	0.8483	11049	Comedy
0.8000	0.6134	0.6446	0.8327	7696	Comedy
0.8500	0.5866	0.6328	0.8220	5166	Comedy
0.9000	0.5629	0.6087	0.7963	3681	Comedy
0.9500	0.5235	0.5619	0.7621	2535	Comedy
1.0000	0.5203	0.5587	0.7483	1575	Comedy
0.0000	2.2500	2.7500	4.2500	4	Documentary
0.0500	2.2500	2.5000	4.2500	2	Documentary
0.1000	2.1875	2.5625	4.1875	8	Documentary
0.2000	1.5000	2.0000	2.7000	5	Documentary
0.2500	1.5278	1.7778	2.4722	18	Documentary
0.3000	1.5208	1.6875	2.4375	24	Documentary
0.3500	1.0781	1.0469	1.7031	32	Documentary
0.4000	1.0791	0.8669	1.4245	139	Documentary
0.4500	1.0572	0.7063	1.1867	332	Documentary
0.5000	0.9867	0.6755	1.0932	1014	Documentary
0.5500	0.9182	0.6236	1.0244	1639	Documentary
0.6000	0.7486	0.6071	1.0545	1046	Documentary
0.6500	0.6825	0.5450	1.0208	600	Documentary
0.7000	0.6706	0.6120	1.0518	299	Documentary
0.7500	0.7542	0.5958	0.9542	120	Documentary
0.8000	0.7089	0.6835	1.0127	79	Documentary
0.8500	0.5645	0.5323	0.9194	31	Documentary
0.9000	1.1500	0.8000	0.7500	10	Documentary

Table A.5: Experiment 4: Average Errors - Expertise (Part-1)

Expertise	Mean Error	Median Error	ML Error	Number	Genre
0.0000	1.1095	0.6921	0.9690	242	Drama
0.0500	1.2293	0.7720	1.1427	410	Drama
0.1000	1.1991	0.8802	1.0775	555	Drama
0.1500	1.1908	0.8804	1.0628	828	Drama
0.2000	1.1368	0.8095	1.1359	1155	Drama
0.2500	1.0980	0.7901	1.1521	1739	Drama
0.3000	1.1289	0.7930	1.0685	2956	Drama
0.3500	1.0771	0.8011	1.0699	5259	Drama
0.4000	1.0347	0.7723	1.0359	8800	Drama
0.4500	1.0154	0.7672	1.0315	14684	Drama
0.5000	0.9520	0.7455	1.0374	25095	Drama
0.5500	0.8394	0.7003	0.9930	33698	Drama
0.6000	0.7251	0.6771	0.9788	33552	Drama
0.6500	0.6935	0.6608	0.9515	27758	Drama
0.7000	0.6547	0.6568	0.9279	20507	Drama
0.7500	0.6395	0.6688	0.9057	15061	Drama
0.8000	0.6189	0.6539	0.9041	9676	Drama
0.8500	0.5950	0.6268	0.8653	6368	Drama
0.9000	0.5558	0.5903	0.8192	4253	Drama
0.9500	0.5412	0.5803	0.8154	2863	Drama
1.0000	0.5029	0.5679	0.8183	1208	Drama
0.0000	1.0125	0.6250	1.0875	40	Horror
0.0500	1.0309	0.6543	1.2469	81	Horror
0.1000	0.9380	0.7403	1.3566	129	Horror
0.1500	1.0204	0.7500	1.3367	196	Horror
0.2000	1.0566	0.8164	1.2930	256	Horror
0.2500	1.0157	0.8583	1.3110	508	Horror
0.3000	0.9718	0.7599	1.1472	781	Horror
0.3500	0.9961	0.8373	1.0947	1420	Horror
0.4000	0.9040	0.7736	1.0279	2151	Horror
0.4500	0.9085	0.7844	1.0138	3179	Horror
0.5000	0.8776	0.7744	0.9730	4800	Horror
0.5500	0.8265	0.7676	0.9605	5882	Horror
0.6000	0.7473	0.7247	0.9135	4146	Horror
0.6500	0.6947	0.7024	0.8973	2774	Horror
0.7000	0.6808	0.7008	0.8699	1925	Horror
0.7500	0.6927	0.7142	0.8779	1069	Horror
0.8000	0.6538	0.6850	0.8413	800	Horror
0.8500	0.5904	0.6507	0.8258	531	Horror
0.9000	0.5607	0.6084	0.8150	346	Horror
0.9500	0.5357	0.5357	0.7610	182	Horror
1.0000	0.6395	0.5756	0.7907	86	Horror

Table A.6: Experiment 4: Average Errors - Expertise (Part-2)