Social Information Filtering for Music Recommendation

by

Upendra Shardanand

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degrees of

Bachelor of Science in Electrical Science and Engineering

and

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 1994

Author ....
Department of Electrical Engineering and Computer Science
September 1, 1994

Certified by ....
Pattie Maes
Assistant Professor of Media Arts and Sciences
Thesis Supervisor

Accepted by ....
Frederic R. Morgenthaler
Chairman, Department Committee on Undergraduate Theses

Social Information Filtering for Music Recommendation

by

Upendra Shardanand

Submitted to the Department of Electrical Engineering and Computer Science
on September 1, 1994, in partial fulfillment of the
requirements for the degrees of
Bachelor of Science in Electrical Science and Engineering
and
Master of Engineering in Electrical Engineering and Computer Science

Abstract

Filters which select items for individual users based upon content suffer from several limitations. The items being filtered must be amenable to parsing by a computer. Furthermore, Content-Based Filters possess no inherent method for serendipitous exploration of the information space.

This thesis proposes Social Information Filtering (SF), a general, novel approach to information filtering. SF systems filter items based upon other users whose tastes are similar to your own. SF overcomes the limitations of Content-Based Filters.

An implementation employing Social Information Filtering techniques is presented. Ringo is a music recommendation system accessible to users via electronic mail. Users rate musical artists and then are able to receive recommendations for further listening.

Several filtering schemes are described, analyzed and compared. Experimental results demonstrate the capabilities of SF and its potential for immediate application.

Thesis Supervisor: Pattie Maes
Title: Assistant Professor of Media Arts and Sciences

Social Information Filtering for Music Recommendation

by

Upendra Shardanand

Submitted to the Department of Electrical Engineering and Computer Science
on September 1, 1994, in partial fulfillment of the
requirements for the degrees of
Bachelor of Science in Electrical Science and Engineering
and
Master of Engineering in Electrical Engineering and Computer Science

Abstract

Filters which select items for individual users based upon content suffer from several limitations. The items being filtered must be amenable to parsing by a computer. Furthermore, Content-Based Filters possess no inherent method for serendipitous exploration of the information space.

This thesis proposes Social Information Filtering (SF), a general, novel approach to information filtering. SF systems filter items based upon other users whose tastes are similar to your own. SF overcomes the limitations of Content-Based Filters.

An implementation employing Social Information Filtering techniques is presented. Ringo is a music recommendation system accessible to users via electronic mail. Users rate musical artists and then are able to receive recommendations for further listening.

Several filtering schemes are described, analyzed and compared. Experimental results demonstrate the capabilities of SF and its potential for immediate application.

Thesis Supervisor: Pattie Maes
Title: Assistant Professor of Media Arts and Sciences

Acknowledgments

"Your friend is your needs answered."

— K. Gibran, *The Prophet*

I'd like to thank my advisor, Pattie Maes, for being simply amazing. The other night, I was talking to some fellow graduate students who were complaining about their advisor and their group. On and on and on they went. Finally they turned to me and asked, "So, do you have any gripes to share about your group?" "No," I replied flatly. "Pattie's incredible. I couldn't ask for a better advisor." True, I would have said that anyway just to make them jealous. But, it happens to be the truth. Pattie is the greatest.

And, so is our agents group, namely Bruce Blumberg, Christie Davidson, Lenny Foner, Michael Johnson, Henry Lieberman, Max Metral, And yes, I'll even include Yedzi Lashkari in that distinguished bunch. But I'll misspell his name on purpose. Thanks for making 305 such a cool place.

Thanks to Lee Zamir, my right-hand man for the past summer. You've survived my rule by love, fear, and ridicule. Ringo is now in your hands, take good care of it.

Many thanks to John McNulty, for his comments on drafts of the thesis and his encouragement ("You're screwed! You'll never graduate!"), Ken Mungan, for lending me his keen mathematical mind, Eric Jordan, for being a calming influence, and Ed Hwang, for letting me crash at his place while I was finishing my thesis (after MIT housing decided that my lease should expire two weeks before my thesis deadline. Just wait till they ask for alumni donations).

On those occasions when I have things in perspective, I feel truly blessed to have all these amazing people as friends.

Thanks to the News in the Future Consortium for providing financial support for this project.

Haven't you spent enough time reading the acknowledgments? Read the thesis!

Contents

List of Figures

List of Tables

Chapter 1

Introduction

"KNOWLEDGE IS POWER."
— Hobbes

"IGNORANCE IS BLISS."
— Gray

—⋆—

1.1 Information Overload

Recent years have seen the explosive growth of the sheer volume of everyday things. The number of products, books, music, movies, news, advertisements, and the flow of information in general, is staggering. This truly is an "information age." The volume of things is considerably more than any person can digest. A person could not possibly filter through every item in order to select the ones that he or she truly wants and needs.

Consider these statistics:

- An average Blockbuster video store has 6,000 videos to choose from.
- The BMG Compact Disc club has over 14,000 titles for sale.
- The average USENET site stores over 113,000 articles at any given time[7].
- The Library of Congress houses 101,395,257 items, including 15,700,905 books.
- Mark Goodson alone has produced over 38,000 game show episodes.

People handle this information overload through their own effort, the effort of others and some blind luck. Most items and information are filtered from the stream simply because they are either inaccessible or invisible to the user. A large amount of filtering is done for us. Newspaper editors select what articles their readers want to read. Bookstores decide what books to carry. Television only offers a limited number of options. However with the dawn of the information age, this barrier will become less and less a factor. As the world becomes more digital and interactive, with the promise of the "information superhighway," more and more options will become available. We then will use what little time we have to select items worthy of our time and effort.

There is a need for technology to help us wade through all the information to find the items we really want and need, and to rid us of the things we do not want to be bothered with. Much research in the field of *agents* - adaptive, personalized, software assistants - has focused on information filtering. For example, NewT selects articles from USENET news groups for its user, based upon a profile of a person's reading habits built and modified over time [10]. Still, more research in techniques for information filtering needs to be done.

1.2 The Limitations of Content-based Filtering

The common and obvious technique used to tackle the problem of information filtering is *Content-based Filtering.* The filter selects items for the user's consumption based upon correlations between the *content* of the items and the user's preferences. For example, NewT uses content information about an article such as keywords in the article's text and the newsgroups in which the article is posted to determine whether to deliver the article.

However, Content-based Filtering has limitations [3]:

- The items must be of some machine parsable form, or attributes must have been assigned to the items by hand. With current technology, media such as sound, photographs, art, video, physical items, and some multimedia cannot be analyzed automatically for relevant attribute information, in the manner that text can be analyzed. It is not practical or possible to parse other items due to limitations of resources. For example, the contents of the Library of Congress could take decades to digitize. Content-based filtering does not lend itself well to such media and circumstances.

- Content-based Filtering has no inherent method for generating serendipitous finds. People rely on exploration and luck to find new items that they did not know they wanted. A person may not know they like watching day time talk shows until they accidentally turn to 'Oprah.' However, if the individual's previous tastes provide no indication of this new penchant, the Content-based Filter will never select such an item for consumption. Without the capability for exploration, the range of items that the user sees can never expand. In practice, additional hacks often have to be added to Content-Based Filters to introduce some element of serendipity.

A complementary filtering technique is needed to address these issues. The approach must be applicable to a larger range of media domains. In addition, it must benefit from some form of fortuitous exploration.

1.3 Social Information Filtering

This thesis presents Social Information Filtering (SF), a general, novel approach to Information Filtering. SF systems filter items for a user based upon the likes of other people with similar tastes. Social Information Filtering essentially automates the process of "word-of-mouth" recommendations.

SF overcomes some of the limitations of filters which select items based upon content. The items being filtered need not be amenable to parsing by a computer. Furthermore, it inherently explores the information space for items of interest to the user.

The contributions of this thesis are as follows:

- Social Information Filtering as a method for personalized information filtering is described.
- An implementation of a SF system, called Ringo, which provides personalized music recommendations is discussed.
- The results of the use of this system by thousands of actual users are presented.
- Various different filtering schemes are described, analyzed and compared. Experimental results demonstrate the capabilities of SF and its potential for immediate application.

1.4 Overview of this Document

The following chapter presents *Social Information Filtering*. Previous research in this area is discussed. Chapter 3 describes Ringo, a social information filtering system for music recommendation. This chapter details Ringo's implementation and user interface. Chapter 4 gives a technical description of several filtering algorithms experimentally tested, analyzed and compared, within the context of the Ringo system. Chapter 5 presents qualitative results gathered through the administration of Ringo as a real system used by thousands of users. Chapter 6 details possibilities for future work on Ringo and in the area of SF in general. Chapter 7 presents the conclusions gathered from this project.

Chapter 2

Social Information Filtering

"I AM HE AS YOU ARE HE AS YOU ARE ME AND WE ARE ALL TOGETHER."

— J. Lennon and P. McCartney, *I am the Walrus*

—⋆—

Social Information Filtering exploits the similarities between the tastes of different users to filter items. It relies on the fact that people's tastes are not randomly distributed: there are general trends and patterns within the taste of a person and between groups of people. This is best illustrated by example:

- Han loves Sci-Fi books. Therefore, it would be likely that he would be interested in seeing the new "Star Wars" movie.

- James likes to listen to "Ice-T" but dislikes "Yanni" He would probably be more interested in listening to "Ice Cube" than "Enya."

- Hillary and Bill very often agree in their opinions of movies. If Hillary enjoyed the movie "Dave," chances are so will Bill.

If people's preferences were random, no such predictions could be made. But in reality, after getting some idea about a person's likes and dislikes, we can often predict what they would like based upon intuition that we have about patterns in people's tastes.

We again resort to the use of a real-life example, not only to illustrate Social Information Filtering but to contrast it with Content-Based Filtering.

Mel wants to see a movie. Out of the movies currently playing, he decides to see "Maverick" because it is a western, stars Jodie Foster, and has a matinée show at the local theater. He thus employed content-based filtering, since he selected a movie based upon its *content*, i.e, attributes known about the movie.

Mel might also have asked two friends, Martin and Francis, for their recommendations. Martin suggests "Raging Bull," while Gary suggests "The Godfather." From past experience, Mel knows that Martin and he have similar tastes, while Francis and he don't always see eye to eye. He therefore accepts Martin's suggestions and decides to watch "Raging Bull." This decision was made through Social Information Filtering, independent of the content of the movies.

Of course, Mel probably used both techniques: We can view his filtering process as a combination of the two phenomena.

Social Information Filtering essentially automates the process of "word-of-mouth" recommendations. Except that instead of having to ask a couple friends about a few items, a SF agent can ask thousands of other people, and consider thousands of different items, all happening autonomously and automatically.

Social Information Filtering agents take advantage of this phenomenon in order to select items for its users. The basic procedure is as follows:

- Over time, construct our user's *profile*, a record of the user's traits based on past history.
- Compare this profile to the profiles of other people collected by other agents, and weight each profile for their degree of *similarity* with our user's profile. The metric used to determine similarity can vary.
- Take a group of the most similar profiles, and use them to construct an answer to some query for our user.
- Relay this information to the user in an appropriate form.

The same can also be done for the items themselves. Profiles for items, i.e. a record of how different users liked the item, can be constructed. The profiles of different items can then be compared to find correlations. These correlations can then be used to answer user queries, as they enable us to find how different items relate to the user's known tastes.

Such a technique can be applied to a variety of problem domains. Of course it can be used to filter items ranging from music to movies, technical journals, office equipment, restaurants, detergents, financial information, and more. When the "information superhighway" becomes a reality and a viewer has 8,000 channels to choose from, a social filtering agent will be necessary for suggesting channels.

With the advent of the "information superhighway," advertisers will be able to broadcast advertisements to individuals, as opposed to large audiences. They will be able to send a particular commercial, for example, to a particular set of users. At present, marketing teams rely on large demographic studies in order to direct advertising. "White males from ages 15-19, who watch *Beavis and Butthead* would respond favorably to this ad about sneakers." With Social Information Filtering, there is no longer a need for such large scale probabilistic advertising. They can deal in demographic groups with a population of one. Each ad can be selected for a television audience of one.

An SF system becomes more competent as the number of users in the system increases. The more people, the greater the chance of finding close matches to any particular users. A SF system may need to reach a certain *critical mass* of collected data before it becomes useful. Thus, over time, as data is collected, such a system will become more accurate and will be able to answer more kinds of queries.

2.1 Previous research

Some current and recent research in the field of information retrieval and information filtering which relate to this work are described below.

2.1.1 Tapestry

Tapestry is another information filtering system which accepts the ratings or *annotations* of users for items, in this case electronic documents such as e-mail and Netnews[4]. As users read documents, they may attach annotations to documents. The filters that search the annotations for interesting articles however are constructed by the end user, using a query language. The query may involve many different criteria, including keywords, subject, authors and the like, and annotations given the document by other people. Therefore, they make it possible to request documents approved by others. However, should the user want to retrieve documents approved by people similar to themselves, they must themselves know who these similar people are and specifically request documents annotated by those people. That is, in this system you still need to know who your friends are with tastes like yourself. Thus, the social information filtering is left to the user: Tapestry only provides an on-line architecture for facilitating the process "word-of-mouth."

2.1.2 GroupLens

A project similar to ours is the *GroupLens* system[8]. GroupLens is a system applying social filtering[1] to the personalized selection of Netnews. It provides an open architecture wherein people can rate articles, and their ratings are distributed through the net to other user's agents through a specialized newsgroup which only carries article ratings. GroupLens employs *Pearson r* correlation coefficients to determine similarity between users. Our problem domain and the algorithms we tested differ, but our approaches are very similar. We discovered some SF schemes which, for our problem domain of music recommendation, perform better than the GroupLens algorithm.

2.1.3 Movie Select

Paramount Interactive offers a commercial application called *Movie Select*[6]. In addition to being an excellent database about movies, it is designed to recommend

[1] *collaborative* filtering, as they call it.

movies to individual users. Its documentation describes its approach as follows:

> How does it work? Hundreds of moviegoers were interviewed to get their list of favorite movies. Movie Select Video Selection System uses artificial intelligence techniques to match you to people who have similar taste. It's like having a panel of people whose taste you really trust always available to recommend movies.

The user is only required to provide the following input: First, they select their three favorite movies from a full listing of the movies in the database. Movie Select then provides a short list of 20-30 movies, from which the user selects the ones they like. Using this data, Movie Select generates a list of movies that are either "recommended" or "highly recommended."

We can only speculate about the algorithms they use. However, we can comment on the form of the package's output. Movie Select does offer a wealth of information about movies, and is extremely easy to use. Its recommendation feature, however, does not seem to produce out of the ordinary results. It generates an extremely large list of recommendations. The lists produced do not feel very personalized. They seem more like Top 20 lists, as opposed to insights into the quirks of the user's tastes.

Our approach, as well as others listed here such as GroupLens and videos@bellcore (discussed below), digs for greater detail about the user's preferences, and thereby produces greater detail in the predictions. Furthermore, Movie Select is a static package, using a data set that does not change over time. Nor does it record any history of a person's past use.

2.1.4 Claire V.

Another commercial movie recommendation service is *Claire V.* offered by Entertainment Decisions, Inc[2]. Claire V. is a kiosk installation designed for video rental stores. Customers would use Claire V. on a regular basis, rating movies and receiving recommendations. At this time I have no technical information on the project. However, their portfolio indicates some form of process which seems to be Social Information Filtering in nature.

2.1.5 videos@bellcore.com

videos@bellcore.com is a personalized movie recommendation system[5]. A user accesses it by e-mail.[2] It too uses social filtering techniques. Unfortunately, at the time no details were available about its inner workings. Its help file describes its operation as follows:

> After you return your ratings, videos@bellcore.com sends you a list of suggested movie titles. Behind the scenes, it finds other viewers who like and dislike movies the way you like and dislike movies. It then computes what movies they have seen and liked that you have not yet seen.
>
> On occasion, when other viewers request suggestions, your ratings will match theirs and then videos that you've seen and liked that they have not yet seen will be suggested to them.

When videos@bellcore.com sends a user suggestions, it also lists those users whose tastes correlate highly with your own. At the time of this writing, the videos project has 290 participants.

The user interface for videos@bellcore.com system was used as a reference when designing the e-mail interface for Ringo, described in the following chapter.

Our system, in our opinion, tackled the much more difficult problem domain of music. Furthermore, the scale or our project was much greater, serving 2100 users in less than two months, with 3000 artists and 9000 albums at the time of this writing.

2.1.6 ICSI Personalized Movie Recommendations

David Anderson at the International Computer Science Institute also has an Internet service providing personalized movie recommendations[1]. His Internet advertisement for the service reads as follows:

> Do your tastes differ from those of the masses, so that population-averaged movie ratings are of little use in picking movies to see? If so, read on.
>
> I have developed a system to predict what movies you will like, based on your ratings of the movies you have already seen. It uses a large (200,000+ entry)

[2] to videos@bellcore.com.

> database of ratings from the Internet population. To predict your rating of a movie, the program computes the correlation (+ or -) between you and each of the people who have rated that movie. These correlations are used to weight the ratings. The results are about 30% more accurate, on the average, than the population average.

After rating movies, a user receives a reply with the 50 or so top recommendations and the 25 for which the deviation from average is largest.

Unfortunately, little information is available about the inner workings of the system. From the outside, we can only draw the aforementioned comparisons of the problem domains between this system and Ringo.

Chapter 3

Ringo: A Personal Music Advisor

"THOUGH THIS BE MADNESS, YET THERE IS METHOD IN'T."

— W. Shakespeare

—⋆—

3.1 Introduction

Ringo[1] is an agent for personalized music recommendation. People describe their listening pleasures by rating some music artists. These ratings constitute the person's *profile*. This profile changes over time as the individual rates more artists. Ringo uses these profiles and a Social Filtering approach to generate advice to individual users. What should they listen to? What should they avoid? Along with the recommendations, Ringo dispenses personalized music reviews, as written by the users themselves. Furthermore, Ringo is a dynamic, user-grown database of musical artists and albums.

Ringo is an on–line service accessed through electronic mail. Users may sign up with Ringo by sending an e-mail to *ringo@media.mit.edu* with the word "join" in the body. People interact with Ringo by sending commands and data to a central server via e-mail. Once an hour, the server processes all incoming messages and sends replies as necessary.

[1] If you can come up with a good acronym which spells 'Ringo,' let me know.

Alternatively, users will be able to interact with Ringo via its World Wide Web interface, which is under construction. Figure 3-1 depicts a prototype Ringo Web Page.

A number of factors motivated the creation of Ringo:

- To prove that a social filtering approach to item selection is both possible and tractable.
- To observe how humans respond to an artificial, intelligent advisor.
- To create a continuing service that can grow and becomes more competent and useful over time.

3.2 The Problem Domain: Music

A number of problem domains were considered for the social filtering testbed. Music was selected for a number of reasons. First, we speculated that people's tastes in music have a wide range, much more so than movies, for example. Therefore, music would be a greater test of the potential of social filtering.

Second, people feel very personally and very strongly about their listening tastes, much more so than with books, articles, or movies. People tend to tie part of their identity to their music. Therefore, we hoped that people would react strongly (either negatively or positively) to Ringo's performance. This assumption was certainly borne out.

Finally, a personal music advisor is an application that had not yet been created, to our knowledge, and which definitely serves a need.

3.3 Using Ringo

The following sections provide an overview of what interactions can occur between a person and Ringo. For the sake of simplicity, the actual command syntax and much of the structural details of Ringo are avoided in favor of a black box view of

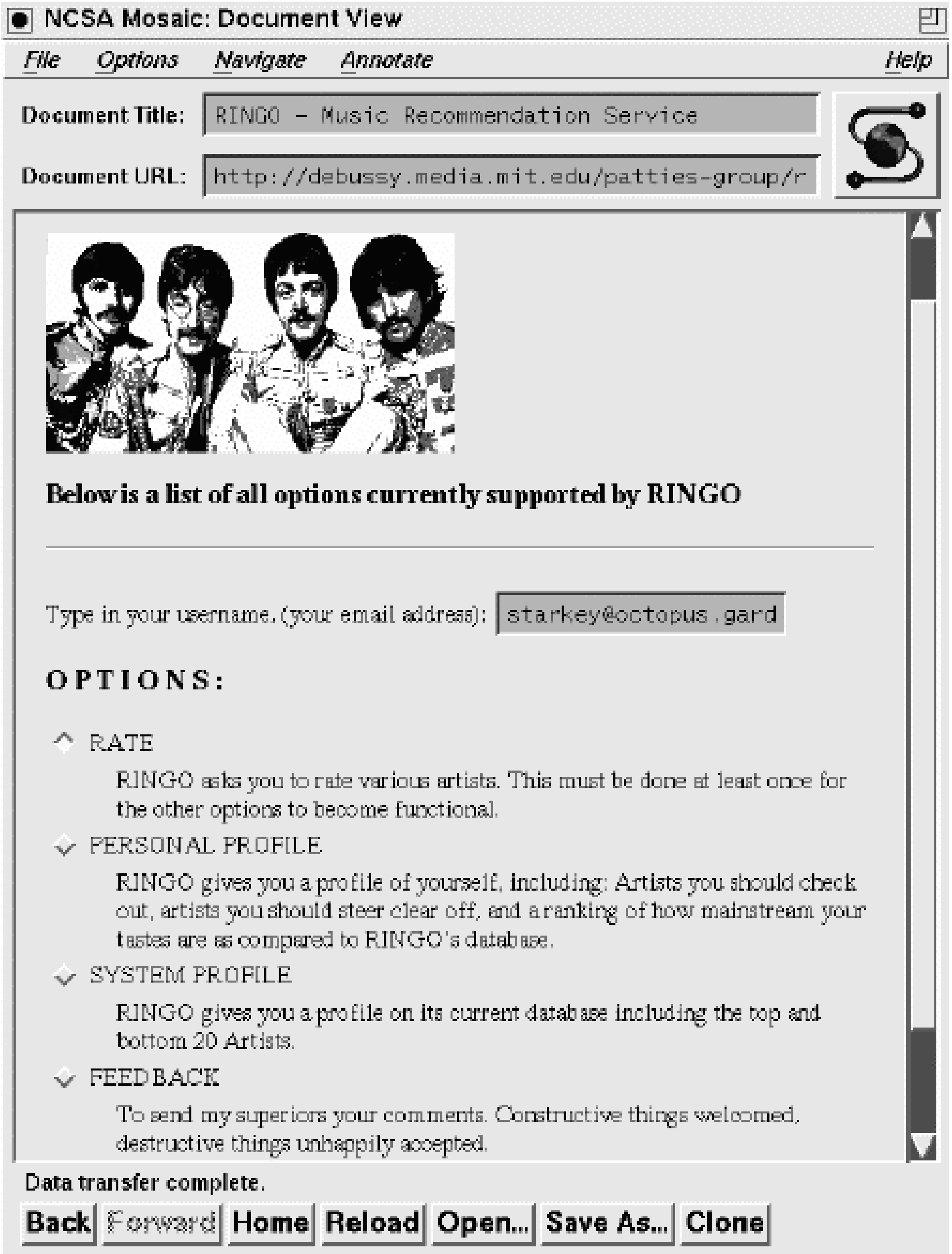

Figure 3-1: A page from Ringo's World Wide Web interface.

the interactions. The exception to this is the actual social filtering engine, which is detailed in Chapter 4.

3.3.1 Scoring Items

When a person first sends mail to Ringo, he or she is sent a list of 125 artists. He or she rates artists for how much they like to listen to them. If they are not familiar with a group or do not have a strong opinion, the user is asked not to rate that item. Users are specifically advised to rate artists for how much they like to *listen* to them, not for any other criteria such as musical skill, originality, or other possible categories of judgment. Part of an actual set of ratings is depicted in Figure 3-2.

The list of artists sent to a user is selected through a simplistic method. Part of the list is generated from a list of the most often rated artists. This ensures that a new user has the opportunity to rate groups which others have also rated, so that there is some commonality in people's profiles. The other part of the list is generated through random selection. Thus, artists are never left out of the loop. This procedure leaves much room for future improvement and research, but has been sufficient for our early tests.

Once a person's initial profile has been submitted to Ringo, Ringo sends a help file to the user, detailing all of Ringo's commands.

A user may also request a list of any group's albums, and rate that group's albums on an individual basis.

The Scale

Users were asked to rate artists on a scale from **1** to **7**, with **1** being the lowest and **7** being the highest, seen in Figure 3-3. When Ringo predicts how much a user will (dis)like an artist, it calculates a "predicted rating" on the same scale.

A 7–point scale was selected since, according to one source, the reliability of data collected in surveys does not increase substantially if the number of points is increased beyond 7 [9]. A couple of users did request one or two more points in the scale, since some people felt that they could distinguish between artists more precisely, especially

6	"10,000 Maniacs"
3	"AC/DC"
3	"Abdul, Paula"
2	"Ace of Base"
1	"Adams, Bryan"
	"Aerosmith"
	"Alpha Blondy"
6	"Anderson, Laurie"
5	"Arrested Development"
	"Autechre"
3	"B-52s"
	"Babes in Toyland"
	"Be Bop Deluxe"
5	"Beach Boys, The"
	"Beastie Boys"
4	"Beat Happening"
7	"Beatles, The"
1	"Bee Gees"
	"Bigod 20"
	"Biz Markie"
5	"Bjork"
4	"Blondie"
	"Blues Traveler"
1	"Bolton, Michael"
1	"Bon Jovi, Jon"
	"Bowie, David"
	"Brown, Ray"
6	"Bush, Kate"

Figure 3-2: Part of one person's survey.

7 :	BOOM! One of my FAVORITE few! Can't live without it.
6 :	Solid. They are up there.
5 :	Good Stuff.
4 :	Doesn't turn me on, doesn't bother me.
3 :	Eh. Not really my thing.
2 :	Barely tolerable.
1 :	Pass the earplugs.

Figure 3-3: Ringo's scale for rating music.

in the **5–6** range. Therefore, shortly after Ringo's inception, users were allowed to use decimal scores to express ratings which fall "in between" the standard 7 points. The degree to which the extra granularity made a difference to the performance of the algorithm is minimal, but it did make some users more comfortable in rating artists.

The scale is meant to be absolute. In other words, a score given by one user is assumed to have the same 'worth' as the same score given by other users. The obvious question then is "Why is that necessary? Why can't you just normalize responses around each person's individual mean?" First of all, for normalized responses to be approximately equivalent, the mean scores given by any two users would need to have the same 'worth' to each user. For the means to be equivalent, the users would have to consistently rate both artists that they like and dislike. However, we speculated, and were proven correct, that people would rate a list of artists in different styles. Some people would only rate items they like and leave everything else blank. Some people would give almost all '1's and just rate a few highly. Others were somewhere in between. Clearly their mean scores will vary. Yet that mean score will have a different 'worth' to different users.

Therefore, the absolute scale was constructed. Descriptions are provided for each rating point to make it clear what each number means (see Figure 3-3). We then assume that the same score given by different users means approximately the same thing.

3.3.2 Reviews

While rating an artist or album, a person can also write a short review, which Ringo stores. When a user is recommended to try or to avoid an artist, any reviews for that artist written by similar users is provided by Ringo as well. Thus, rather than one "Siskel and Ebert" review being given to the entire audience, each user receives personalized reviews, from people that have similar taste.

Every user always has the option of signing their review or remaining anonymous.

A couple of actual reviews are included in Figure 3.3.2 for your amusement:

Tori Amos has my vote for the best artist ever. Her lyrics and music are very inspiring and thought provoking. Her music is perfect for almost any mood. Her beautiful mastery of the piano comes from her playing since she was two years old. But, her wonderful piano arrangements are accompanied by her angelic yet seductive voice. If you don't have either of her two albums, I would very strongly suggest that you go, no better yet, run down and pick them up. They have been a big part of my life and they can do the same for others. —- user@place.edu

I'd rather dive into a pool of dull razor blades than listen to Yoko Ono sing. OK, I'm exaggerating. But her voice is *awful* She ought to put a band together with Linda McCartney. Two Beatles wives with little musical talent.

Figure 3-4: Two sample reviews written by users.

3.3.3 Obtaining Suggestions

An individual can ask Ringo for predictions based upon their personal profile. Specifically, a person can ask Ringo to:

- Suggest artists new to the user that they will enjoy.
- List artists that the user will hate.
- Make a prediction about a specific artist.

When Ringo receives such a request, it does quite a bit of processing using its social filtering engine, detailed in the next chapter. It then sends e-mail back to the person with the result of their requests. Figure 3.3.3 provides an example of Ringo's suggestions.

3.3.4 Miscellaneous Features

In addition, a number of other miscellaneous features are designed into Ringo, in order to increase the appeal of the system:

- The user may add new artists and their albums into the database. This feature has been responsible for the growth of the database from 575 artists at inception

Artist	Rating	Confidence
"Orb, The"	6.9	fair
"Negativland"	6.5	high
Reviews for "Negativland" They make you laugh at the fact that nothing is funny any more — user@place.edu		
"New Order"	6.5	fair
Reviews for "New Order" Their albums until 'Brotherhood' were excellent. Since then, they have become a tad too tame and predictable. And sadly, they have been overplayed. — lost@elsewhere.com		
"Sonic Youth"	6.5	fair
Reviews for "Sonic Youth" Confusion is Sex: come closer and I'll tell you.		
"Grifters"	6.4	fair
"Dinosaur Jr."	6.4	fair
"Velvet Underground, The"	6.3	low
Reviews for "Velvet Underground, The" The most amazing band ever.		
"Mudhoney"	6.3	fair

Figure 3-5: One of Ringo's suggestions.

to over 2500 artists in the first 6 weeks of Ringo's on–line existence.

- Ringo, on request, provides dossiers on any artist. The dossier includes a list of that artist's albums, straight averages of scores given that artist and the artist's albums. It also includes any added history about the group, which can be submitted by anyone and are manually included by Ringo's administrators.
- People can also view a "Top 30" and "Bottom 30" list of the most highly and poorly rated artists, on average.
- If they wish, a user can view the entire database of artists.
- Ringo can provide more lists of artists for the user to rate, in order for Ringo to get to know the user's profile better.
- People can subscribe to a periodic newsletter keeping them up to date on changes and developments in Ringo.

As of the time of this writing, there is quite a long wish list of features that we wish to add and that users have requested we add to Ringo. As always, so much to do, so little time.

Chapter 4

Algorithms and Results

"One should not multiply explanations and causes unless it is strictly necessary."

— Umberto Eco, *The Name of the Rose*

—⋆—

4.1 Inception

Ringo became available to the Internet public July 1, 1994. The service was originally advertised on only four USENET newsgroups. After a slow start, the number of people using Ringo grew fantastically. Word about the service spread rapidly, as people would tell their friends, or send messages to mailing lists. Ringo reached the 1000 user mark in less than a month, and had 1900 users after 7 weeks. At the time of this writing Ringo has 2100 users and processes almost 500 messages a day.

Like the membership, the size of the database grew quickly. Originally, Ringo had only 575 artists in its database. As we soon discovered, users were extremely eager to add artists to the system. At the time of this writing, there are over 3000 artists and 9000 albums in Ringo's database.

Thanks to the overwhelming user interest, we have at our disposal an enormous amount of data on which to test various social information filtering algorithms. This chapter presents four such schemes.

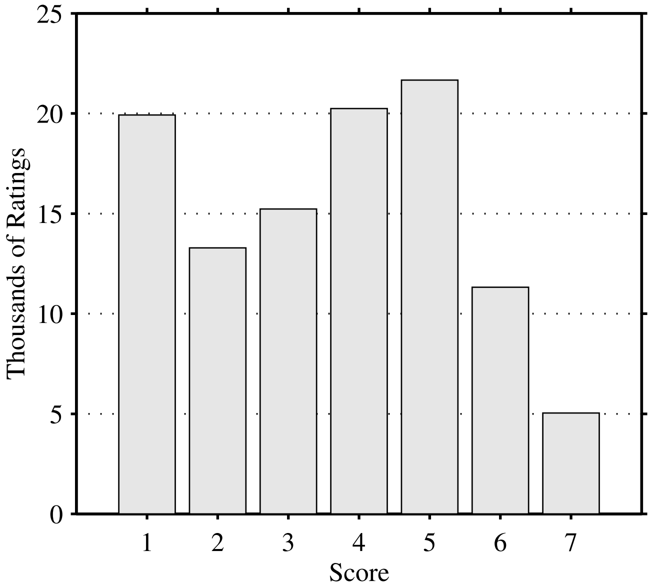

Figure 4-1: The distribution of scores given artists by users.

4.2 The Data

For our tests, the *profiles* of 1000 people were taken. A profile is a sparse vector of the user's ratings for artists. 1,876 different artists were represented in these profiles.

20% of the ratings of each person's profile were then randomly removed. These ratings comprised the *target set* of profiles. The remaining 80% form the *source set*.

To evaluate each algorithm, the test for each algorithm would be: predict a value for each rating in the target set, using only the data in the source set. Three such target sets and data sets were randomly created and tested, to check for consistency in our results. For brevity, the results from the first set are presented throughout this chapter, as results from all three sets only differed slightly.

26,181 ratings formed the target set, while 106,271 ratings comprised the source set. The distribution of scores in the source set is shown in Figure 4-1. Note that people are much more generous with scores of **1** than **7**.

In the source set, each person has rated on average 106 artists of the 1,876 possible. The median number of ratings was 75, and the most ratings by a single person was 772! The mean score of each profile, the average score given artists by a user, was on average 3.7. Figure 4-2 shows the distribution of mean scores.

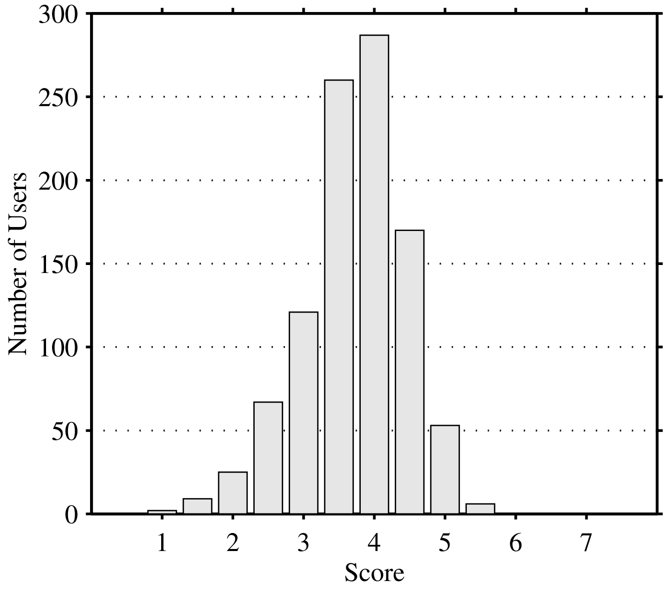

Figure 4-2: The distribution of mean scores of user profiles.

4.3 Evaluation Criteria

The following criteria will be used to evaluate each prediction scheme:

- The mean absolute error of each predicted rating must be minimized. If $\{r_1, \ldots, r_N\}$ are all the real values in the target set, and $\{p_1, \ldots, p_N\}$ are the predicted values, and $E = \{\varepsilon_1, \ldots, \varepsilon_N\} = \{p_1 - r_1, \ldots, p_N - r_N\}$ are the errors, then the mean absolute error is

$$\overline{|E|} = \frac{\sum_{i=1}^{N} |\varepsilon_i|}{N} \tag{4.1}$$

  The lower the mean absolute error, the more accurate the scheme. We cannot expect to lower $\overline{|E|}$ below the error in people's ratings of artists. If you provide the same list of artists to a person at different points of time, the resulting data collected will differ to some degree. The degree of this error has not yet been measured. However we would expect the error to be between ± 0.5 and ± 1 unit on the rating scale.

- The standard deviation of the errors,

$$\sigma = \sqrt{\frac{(\sum(E - \overline{E})^2)}{N}} \tag{4.2}$$

should also be minimized. The lower the deviation, the more consistently accurate the scheme is.

- The number of target values for which the scheme is able to compute predictions should be maximized. Some schemes may not be able to make predictions in all cases.

- It would also be advantageous to have some measures of *confidence* in each individual prediction. The level of confidence would reflect how reliable the particular prediction is, i.e., what the chances of Ringo being wrong is. The ability to distinguish between "sure-bets" and "long shots" would make the system more informative and useful for a user.

4.4 Mathematical Notation

Every attempt has been taken to express all calculations in this thesis in both mathematical notation and plain English. Therefore, you can ignore the math and still understand the concepts involved.

For those readers who enjoy reading mathematical symbols, we now introduce some more notation:

$\mathcal{S} = \{s_{ij}\}$ and $\mathcal{T} = \{t_{ij}\}$ are the source set and target set, respectively. s_{ij} and t_{ij} is the score given artist j by user i in $\mathcal{S}$ and $\mathcal{T}$, respectively. Note that given the sparse nature of the user-artist matrix, most possible s_{ij} and t_{ij}'s are undefined. In our equations, it is always assumed that we perform operations on those values that exist.

We let $c_{ij} = [1, 0]$ if s_{ij} [is,is not] defined. Likewise, $d_{ij} = [1, 0]$ if t_{ij} [is,is not] defined.

$U = \{u_1, \ldots, u_{N_u}\}$ is the set of all N_u users. $U_i =< u_{ij} >$ is the profile of user i. $\overline{U_i}$ is the mean score of that user's profile.

$A = \{a_1, \ldots, a_{N_a}\}$ is the set of all N_a artists. $A_j =< a_{ij} >$ is the vector of all scores given artist j, i.e. artist j's profile. $\overline{A_j}$ is the mean score of the scores given artist j.

4.5 The Base Algorithm: Averages

A point of comparison is needed in order to measure the quality of the social filtering schemes. As a base case, for each artist in the target set, the mean score received by an artist in the source set is used as the predicted score for that artist. A social filtering algorithm is neither personalized nor accurate unless it is a significant improvement over this approach.

Our predicted score, p_{ij} for each t_{ij} in $\mathcal{T}$ is then

$$p_{ij} = \overline{A_j} = \frac{\sum_n^{N_U} s_{nj}}{\sum_n^{N_U} c_{nj}} \tag{4.3}$$

The distribution of the means of the scores for each artist, $\overline{A_j}$ is shown in Figure 4-3. On average, 57 people rated each artist. However, the median number of ratings present for each artist is only 16. The dark gray bars in Figure 4-3 represent those artists with more than the median number of ratings. The mean score for each artist generally approaches **4** as the number of ratings for the artist increases. Although some artists seem to be universally hated, none seem to be universally loved (once you disregard those artists with less than the median number of ratings) When you consider only those artists which at least 5% of the users have rated, 27 artists have mean scores less than **2**, while no artist has a mean score greater than **6**. In fact, the greatest mean score is only **5.3**.

Figure 4-4 depicts the distribution of the errors, E. $\overline{|E|}$ is 1.3, and the standard deviation σ is 1.6. The distribution has a nice bell curve shape about 0, which is what desired.

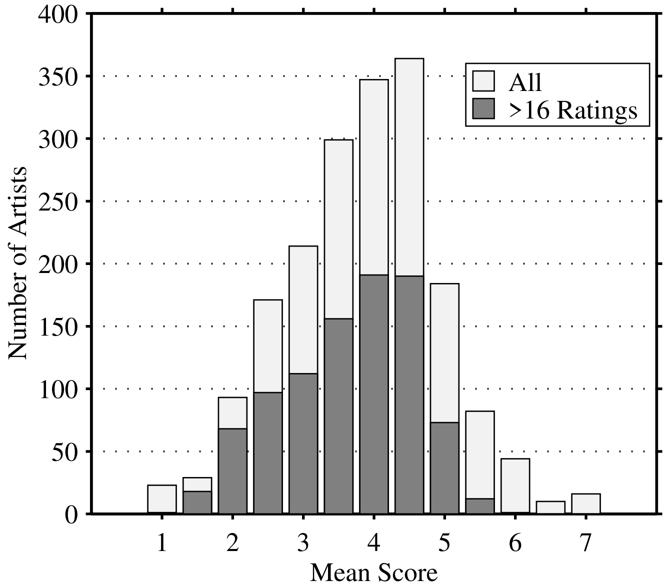

Figure 4-3: The distribution of the means of the scores for each artist.

At first glance, it may seem that this mindless scheme does not behave too poorly. However, let us now restrict our examination to the *extreme* target values, where the score is **6** or greater or **2** or less. These values, after all, are the critical points. User's are most interested in suggestions of items they would love or hate, not of items about which they are ambivalent.

The distribution of errors for extreme values is shown by the dark gray bars in Figure 4-4. The mean error and standard deviation worsen considerably, with $\overline{|E|} = 1.8$ and $\sigma = 2.0$. Note the lack of the desired bell curve shape. It is in fact the sum of two bell curves. The right hill is mainly the errors for those target values which are **2** or less. The left hill is mainly the errors for those target values which are **6** or greater.

For the target values **6** or greater, the mean absolute error is much worse, with $\overline{|E|} = 2.1$.

Why the great discrepancy in error characteristics between all values and only extreme values? As we noted earlier, the mean score for each artist converges to approximately **4**. Therefore, this scheme performs well in cases where the target value is near **4**. However, for the areas of primary interest, the base algorithm is

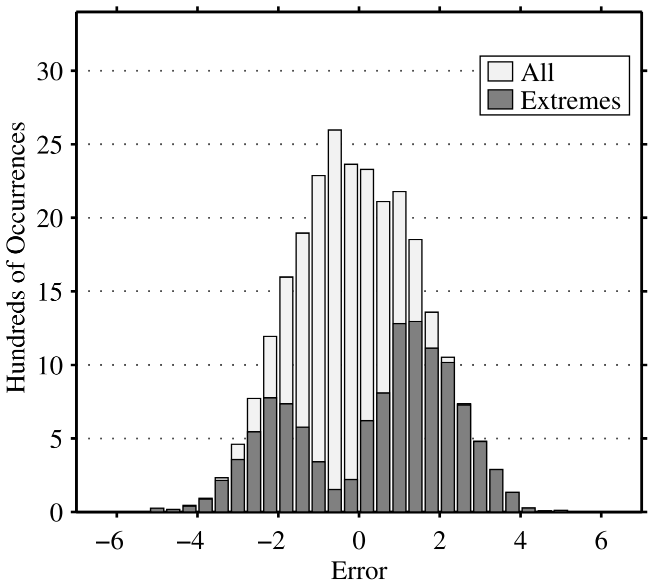

Figure 4-4: The distribution of errors in predictions of the Base Algorithm.

useless.

To its credit, this algorithm could generate predictions for almost all the target values. The exceptions were the cases where there were very few scores for an artist and all the scores were in the target set.

4.6 The Mean Squared Difference Algorithm

Carl Feynman has proposed a Social Filtering scheme where the degree of dissimilarity between two user profiles, U_x and U_y is measured by the *mean squared difference* (MSD) between the two profiles[3]. For every artist that the two people have both rated, take the difference in their ratings and square it. The average of these values is then the mean squared difference :

$$D_{xy} = \overline{(U_X - U_Y)^2} = \frac{\sum_n^{N_a} c_{xn} \times c_{yn} \times (s_{xn} - s_{yn})^2}{\sum_n^{N_a} c_{xn} \times c_{yn}} \tag{4.4}$$

where D_{xy} is the MSD between user x and user y. The mean of these squared differences would then provide a measure of similarity: the lower the mean squared difference, the greater the similarity.

4.6.1 The Algorithm

Using this metric, a simple algorithm to predict the value of some t_{ij}, how user i would rate artist j is:

- (Pre)compute D_{iu}, the mean squared differences between user i and all other users:

- Take all users whose MSD is less than a certain threshold L. These users will comprise the *neighborhood* of people "similar" to user i, $\mathcal{N}_i$.

- Compute the *weight*, w_{ik}, of each user k in $\mathcal{N}_i$. One method for generating weights from the MSD's that has worked well in practice is:

$$w_{ik} = \frac{L - D_{ik}}{L} \tag{4.5}$$

  In all cases, $0 \leq w_{ik} \leq 1$. In all the schemes presented, we will generate weights with values between 0 and 1, for convenience.

- The predicted value p_{ij} for t_{ij} is then a weighted average of all the similar users ratings for item j:

$$p_{ij} = \frac{\sum_k^{\mathcal{N}_i} w_{ik} \times s_{kj}}{\sum_k^{\mathcal{N}_i} w_{ik} \times c_{kj}} \tag{4.6}$$

4.6.2 Results

Figure 4-5 shows the distribution of mean squared differences for our data set. We tested various thresholds, and the results with $L = 2.0$ are described below.

Figure 4-6 shows the distribution of errors. Once again, the darker bars represent the errors for extreme values only. The mean error, $\overline{|E|}$ is 1.0 and σ is 1.3 for all values. For extreme values, $\overline{|E|}$ is 1.2 and σ is 1.6. This is a significant improvement over the Base Algorithm, in all respects.

Note as well that the distribution of errors for the extreme values now more closely resembles a Gaussian distribution, which is desired.

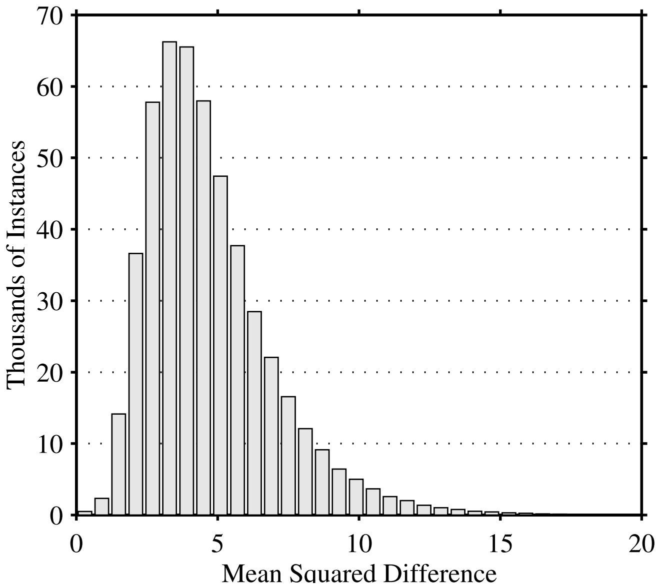

Figure 4-5: The distribution of mean squared differences between people.

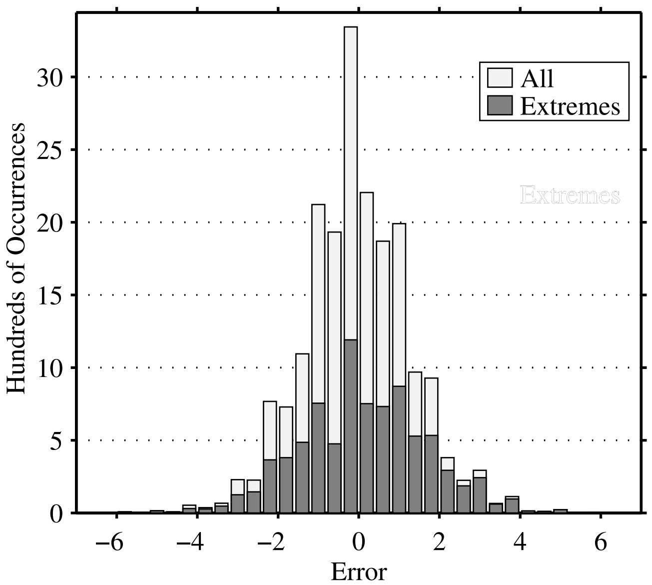

Figure 4-6: The distribution of errors in predictions of the MSD Algorithm.

In this test case, setting $L = 2.0$ seems to be approximately optimal, depending upon what criteria you value most. If we increase the threshold, the accuracy begins to drop fairly quickly, especially for the extreme values. If we lower L, the number of predictions that the algorithm is able to produce drops extremely rapidly. This particular test could only produce predictions for 70% of the over 26,000 values in the target set. This scheme could not produce predictions in cases where none of the members of a particular neighborhood $\mathcal{N}_u$ have rated the artist in question. The lower the threshold, the smaller the neighborhood, and thus the fewer artists the collective pool of similar users has rated. For example, when $L = 1.5$, $\overline{|E|}$ for the extreme values only drops by 0.025, an insignificant amount, while the percentage of target values for which it could make predictions drops to 49%.

4.6.3 Variations

Several variations of the algorithm were tested, with marginal improvement in accuracy (at most $\overline{|E|}$ decreasing less than 0.1 for extreme values). These include, in no particular order:

- Altering the functions for generating weights from the mean squared differences, through different scaling functions.
- Incorporating knowledge of the *commonality* between two users when computing the weights. The level of commonality between two user profiles is proportional to the number of artists both users have rated in common. The level of commonality can also be measured between two artist profiles, in which case it reflects the number of users who have rated both artists. A distribution of the number of artists that any two users have rated in common is shown in Figure 4-7. We suspect that the greater the commonality between two profiles, the more reliable our measure of similarity between the two. However, the incorporation of commonality into our weights, along with the mean squared difference, provided no significant changes in our results.
- Of course, changing the threshold parameter L, as discussed above.

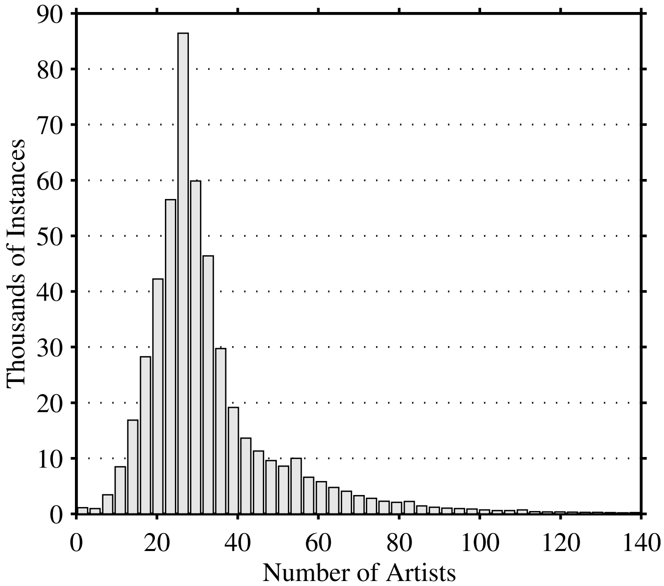

Figure 4-7: The distribution of the number of artists both people have rated in common.

4.7 The Pearson r Algorithm

The GroupLens project, described in Chapter 2, proposes the use of a standard *Pearson r* correlation coefficient to measure similarity between user profiles [8]. The Pearson r correlation coefficient, r_{xy}, between two vectors U_x and U_y, is defined as

$$r_{xy} = \frac{Cov(U_x, U_y)}{\sigma_x \sigma y} \quad (4.7)$$

$$= \frac{\sum(U_x - \overline{U_x})(U_y - \overline{U_y})}{\sqrt{\sum(U_x - \overline{U_x})^2 \times \sum(U_y - \overline{U_y})^2}} \quad (4.8)$$

$$= \frac{N \sum U_x U_y - (\sum U_x)(\sum U_y)}{\sqrt{[N \sum U_x^2 - (\sum U_x)^2][N \sum U_y^2 - (\sum U_y)^2]}} \quad (4.9)$$

with $-1 \leq r_{xy} \leq 1$. A negative r value indicates a *negative correlation*, while a positive r value indicates a *positive correlation*. An r of value 0 indicates no correlation. The greater the magnitude of r, the greater the (dis)similarity between the two profiles.

4.7.1 The Algorithm

Using this metric, an algorithm to predict the value of some t_{ij}, how user i would rate artist j is:

- (Pre)compute all r_{iu}, correlation coefficients between user i and all other users.
- The predicted value p_{ij} for t_{ij} is then a weighted average of all the user ratings for item j:

$$p_{ij} = \overline{U_i} + \frac{\sum_k^{N_u} r_{ku} \times (s_{kj} - \overline{U_k})}{\sum_k^{N_u} r_{ku} \times c_{kj}} \tag{4.10}$$

This algorithm makes use of negative correlations as well as positive correlations to make predictions. The Mean Squared Difference Algorithm, by comparison, only uses those profiles which are similar, or positively correlated, in effect.

4.7.2 Results

A distribution of the correlation coefficients generated between our 1,000 user profiles is depicted in Figure 4-8. The most striking feature is the dearth of highly negatively correlated instances. One explanation resides in the nature of the problem domain itself. We certainly expect that for most people, there will be some others similar tastes in music. There are definite tastes in music. However, would we expect to find people with the *opposite* taste in music? What does it mean to be negatively correlated? That means that everything Mary loves, Joe hates, and everything Mary hates, Joe loves. Furthermore, it means that there cannot be many artists which they are *both* ambivalent about, since that would contribute to a positive correlation.

Let us think about this issue using the following primitive model: there are various distinct genres of musical tastes. Every person loves everything in one or two genres, and despises everything else. So say Mary loves Techno music, and Joe loves Rap. Mary hates Rap, and Joe hates Techno. Restricting our view to these two genres, the two would be negatively correlated. Now include all the other genres. Mary and Joe *both* hate Industrial, Country, Reggae, Pop, Ska, New Age, Folk, Disco, World,

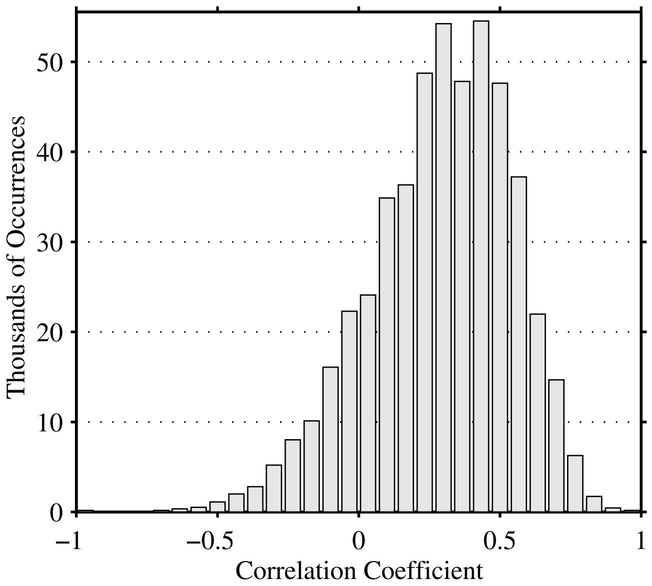

Figure 4-8: The distribution of Pearson r correlation coefficients.

and the list goes on. With respect to all these other genres, they are positively correlated, since their tastes agree. Overall, then, they will have some degree of positive correlation.

This sort of phenomenon could partially explain what we see in Figure 4-8.

The errors for an execution of this algorithm are shown in Figure 4-9. The distribution for extreme values has characteristics of the "double hill" shape that we saw in the Base Algorithm, as opposed to the desired bell curve. The mean error, $\overline{|E|}$ is 1.1 and σ is 1.4 for all values. For extreme values, $\overline{|E|}$ is 1.5 and σ is 1.7. The numbers are not impressive. It could, however generate predictions for 99.9% of the values in $\mathcal{T}$.

In an attempt to improve the performance of the algorithm ,we apply *thresholding*, as we did with the Mean Squared Difference Algorithm. When generating a prediction for a person, we only use data from those individuals who are very positively or very negatively correlated with him/her. The algorithm now becomes:

To predict the value of some t_{ij}, how user i would rate artist j:

- (Pre)compute all r_{ik}, correlation coefficients between user i and all other users:
- Take all users k whose $|r_{ik}|$ is greater than a certain threshold L. These people

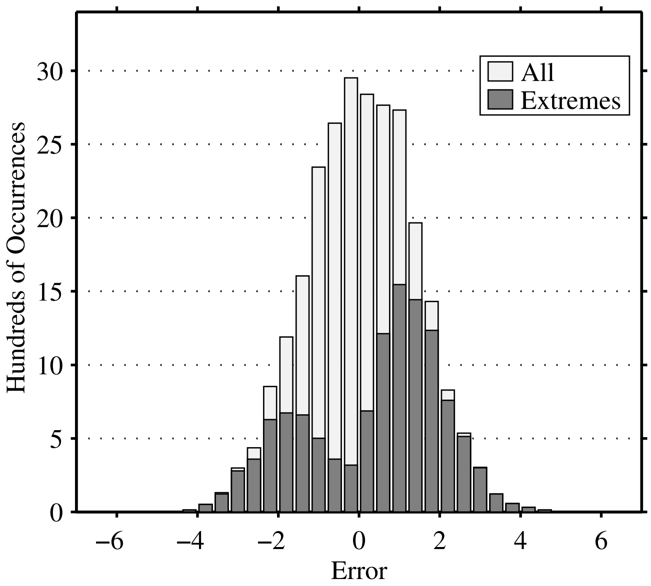

Figure 4-9: The distribution of errors for the Pearson r algorithm.

will form the *neighborhood* of users "similar" to i, $\mathcal{N}_i$.

- Compute the *weight*, w_{ik}, of each user k in $\mathcal{N}_i$:

$$w_{ik} = \begin{cases} \frac{r_{ik}-L}{1-L} & \text{if } r_{ik} > 0 \\ \frac{r_{ik}+L}{1-L} & \text{if } r_{ik} < 0 \end{cases} \tag{4.11}$$

In all cases, $|w_{ik}| \leq 1$.

- The predicted value p_{ij} for t_{ij} is then a weighted average of all the user ratings for item j:

$$p_{ij} = \overline{U_i} + \frac{\sum_k^{\mathcal{N}_i} w_{ku} \times (s_{kj} - \overline{U_k})}{\sum_k^{\mathcal{N}_i} w_{ku} \times c_{kj}} \tag{4.12}$$

L is set to 0.35. At this threshold level, $\overline{|E|} = 1.0$ and $\sigma = 1.3$ for all values, while $\overline{|E|} = 1.4$ and $\sigma = 1.6$ for extreme values. The "double hill" shape of distribution of errors for extreme values is diminishing (Figure 4-10). 99% of all target values can be predicted, even with the thresholding.

With a threshold level of $L = 0.5$, the error distribution for an execution of this

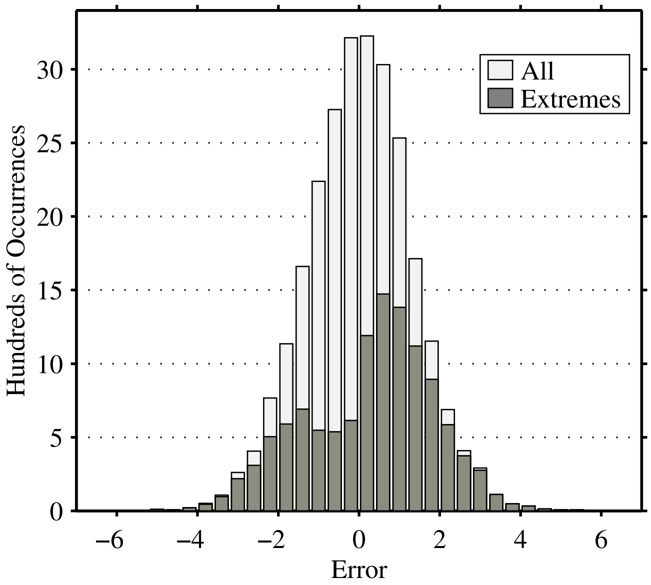

Figure 4-10: The distribution of errors for the Pearson r algorithm with L =0.35

algorithm is shown in Figure 4-11. The mean error, $\overline{|E|}$ is 1.0 and σ is 1.3 for all values. For extreme values, $\overline{|E|}$ is 1.3 and σ is 1.6. The thresholding definitely improved the results. Furthermore, at this threshold level, predictions for 95% of $\mathcal{T}$ can still be made.

As we increase L to 0.65, the performance of the algorithm deteriorates. The mean error, $\overline{|E|}$ is 1.1 and σ is 1.4 for all values. For extreme values, $\overline{|E|}$ is 1.3 and σ is 1.6. The distribution is shown in Figure 4-12. At this level, the prediction of only 73% of the target values is possible.

This downward trend is very surprising. One would naturally expect that the number of possible predictions would drop, since we are decreasing the size of our neighborhood. But one would also expect that the accuracy would improve, as only the more "similar" users are being consulted.

We continue to $L = 0.75$, to see if the trend continues. The $|\overline{E}|$'s worsen by insignificant fractions. But σ for all values drops to 1.5, and to 1.7 for extreme values (see Figure 4-13). Plus, only 41% of $\mathcal{T}$ is predictable¡.

Why is this happening is unclear. Possibly the Pearson r may not be as good an indicator of similarity as we would like.

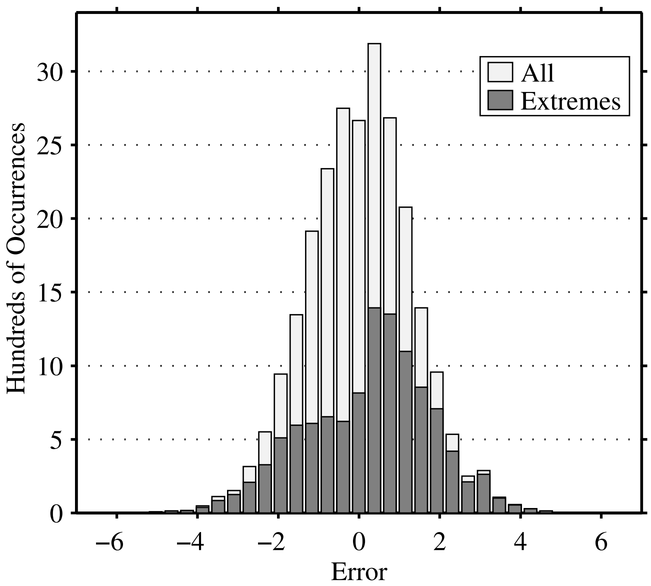

Figure 4-11: The distribution of errors for the Pearson r algorithm with $L = 0.5$.

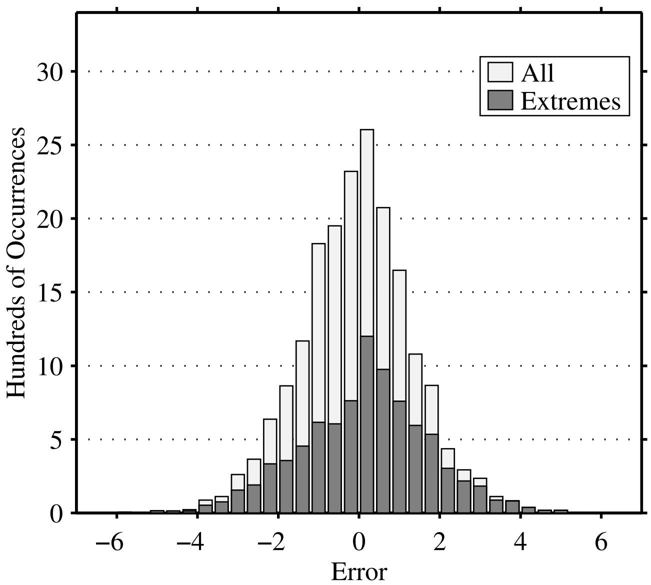

Figure 4-12: The distribution of errors for the Pearson r algorithm with $L = 0.65$.

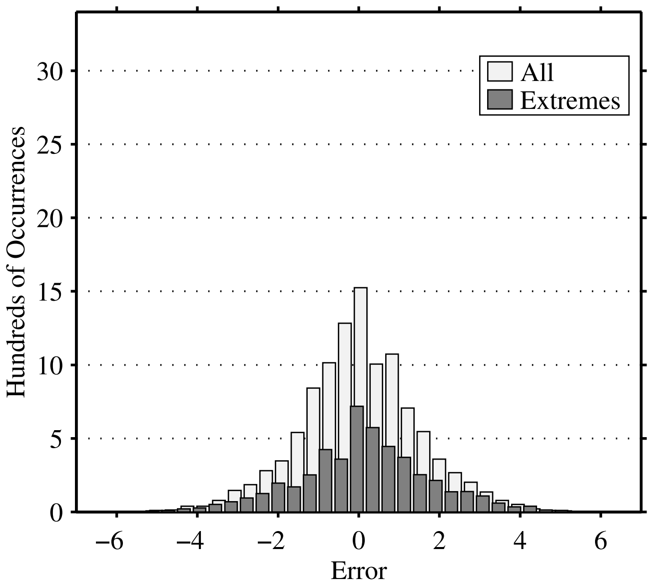

Figure 4-13: The distribution of errors for the Pearson r algorithm with $L = 0.75$.

4.8 The Constrained Pearson r Algorithm

We now turn our attention to Figure 4-14. The x and y-axis represent scores in profiles U_x and U_Y, respectively. In this space, we can plot (x, y) pairs, pairs of ratings given the same artist by our two users. We can visually examine the plotted points of the (x, y) pairs and see the relation between the two users rating habits.

Three trivial sample plots are shown, A, B and C. The data is shown in tabular form in Table 4.1. With the standard Pearson r coefficient calculations, any data whose trend is positively sloped will have a positive correlation, irrespective of position and scaling in the x-y space. In all three examples, the profile pairs have a Pearson $r = +1.0$. This is despite the fact that in cases A and B, one user has low ratings when the other has high ratings.

This effect (or lack thereof) prompted an experiment with the addition of an additional constraint to the Pearson r calculation. Equation 4.8 is altered to become

$$\beta_{xy} = \frac{\sum(U_x - 4)(U_y - 4)}{\sqrt{\sum(U_x - 4)^2 \times \sum(U_y - 4)^2}} \tag{4.13}$$

where β_{xy} is our new measure of "correlation" between profiles U_x and U_y.

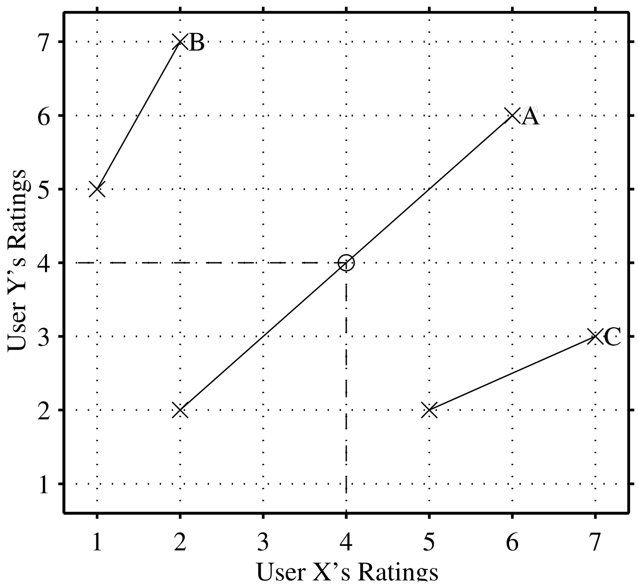

Figure 4-14: Three trivial profile-profile plots.

Plot A	*User X*	*User Y*
Artist 1	2	2
Artist 2	6	6
$r_A = +1.0$ $\beta_A = +1.0$		

Plot B	*User X*	*User Y*
Artist 1	1	5
Artist 2	2	7
$r_B = +1.0$ $\beta_B = -.79$		

Plot C	*User X*	*User Y*
Artist 1	5	2
Artist 2	7	3
$r_C = +1.0$ $\beta_C = -.85$		

Table 4.1: Three trivial sample profile-profile pairs.

The magnitude of β will always be less than 1. This is proven using the *Schwarz Inequality*:

$$[E(XY)]^2 \leq E(X^2)E(Y^2) \tag{4.14}$$

Let $X = (U_x - 4)$ and $Y = (U_y - 4)$. then, following from the Schwarz Inequality:

$$\begin{aligned}
[E(XY)]^2 &\leq E(X^2)E(Y^2) \\
\left(\frac{\sum XY}{N}\right)^2 &\leq \frac{\sum X^2}{N} \times \frac{\sum Y^2}{N} \\
(\sum XY)^2 &\leq \sum X^2 \times \sum Y^2 \\
\frac{(\sum XY)^2}{\sum X^2 \times \sum Y^2} &\leq 1 \\
\pm\frac{\sum XY}{\sqrt{\sum X^2 \times \sum Y^2}} &\leq 1 \\
\left|\frac{\sum(U_x - 4)(U_y - 4)}{\sqrt{\sum(U_x - 4)^2 \times \sum(U_y - 4)^2}}\right| &\leq 1 \\
|\beta_{xy}| &\leq 1 \quad \square
\end{aligned} \tag{4.15}$$

The difference between r and β can be expressed as follows. Say we are comparing user X and user Y.

In the case of r: If X has rated some artist with a score above the average score that X usually gives, and Y also has rated some artist with a score above the average score that Y usually gives, then this pair of values will *increase* their correlation measure r by some amount.

In the case of β: If X has rated some artist with a score above **4**, and Y also has rated some artist with a score above **4**, then this pair of values will *increase* their similarity measure β by some amount.

So this new measure takes the *positivity* and *negativity* of the rating into account. Since it is an absolute scale, we "know" that values below **4** are poor, while values above **4** are good. In this constrained Pearson r scheme, only an instance where both people have rated an artist positively, above **4**, or both negatively, below **4**, will the measure β increase.

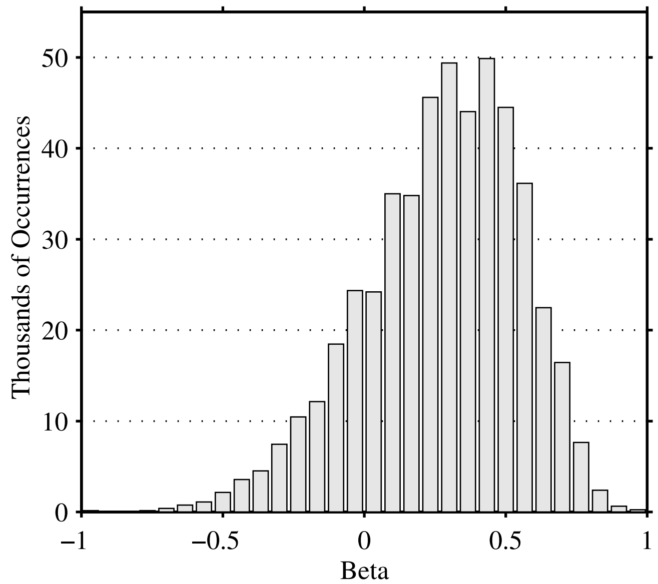

Figure 4-15: The distribution of β-values.

Graphically, referring back to Figure 4-14, the closer our positively sloped line is to the point (4,4), the higher the β-value. The further it is, the lower the β-value. The β-values for our three cases are shown in Table 4.1.

In this respect, the β coefficient behaves like a constrained Pearson r correlation coefficient.

The distribution of β-values is shown in Figure 4-15. Its distribution is very similar to that of the Pearson r's in Figure 4-8. The scales of the two figures are identical, for ease of comparison. Figure 4-16 is a plot of the Pearson r values versus the β-values between the same pair of users. On average, the absolute difference between the r-value and the corresponding β-value is only 0.11.

4.8.1 The Algorithm

Using this metric, we tested a number of algorithms. A simple, effective algorithm to predict the value of some t_{ij}, how user i would rate artist j is:

- (Pre)compute all β_{iu}, coefficients between user i and all other users:

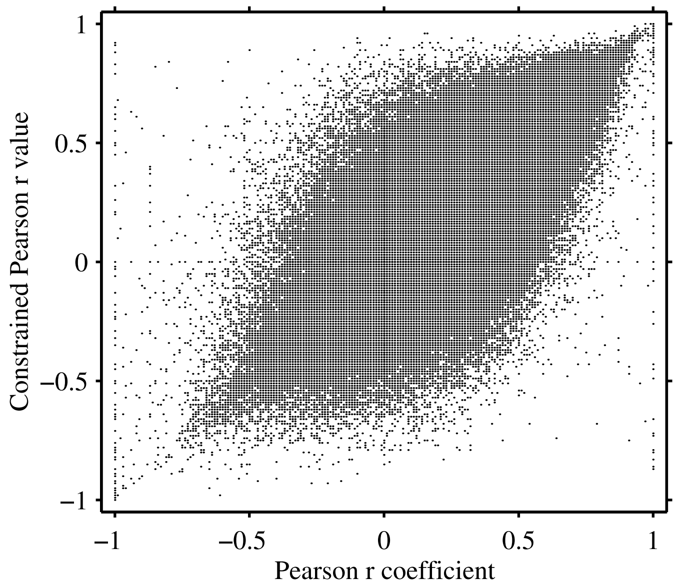

Figure 4-16: Pearson r coefficients vs. β-values for the same user pairs.

- Take all users whose β value is greater than a certain threshold L. These users will comprise the *neighborhood* of "similar" users, $\mathcal{N}_u$.

- Compute the *weight*, w_{iu}, of each user u in $\mathcal{N}_u$. One method for generating weights from the MSD's that has worked well in practice is:

$$w_{iu} = \left(\frac{\beta_{iu} - L}{1 - |L|} \right)^2 \tag{4.16}$$

  In all cases, $|w_{iu}| \leq 1$.

- The predicted value p_{ij} for t_{ij} is then a weighted average of all the similar users ratings for item j:

$$p_{ij} = \overline{U_i} + \frac{\sum_k^{N_u} w_{ku} \times s_{kj}}{\sum_k^{N_u} w_{ku} \times c_{kj}} \tag{4.17}$$

This algorithm as described does not make use of negative "correlations," or negative β-values, like the Pearson r algorithm. There are few very negative β-values, so including them makes little difference.

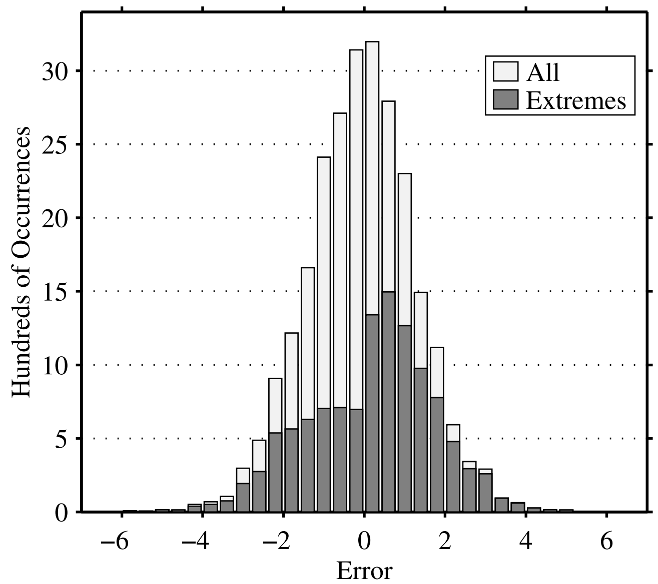

Figure 4-17: The distribution of errors for the Constrained Pearson r algorithm with $L = 0.5$.

4.8.2 Results

Our first test case uses a threshold of $L = 0.5$. Predictions for 97% of the target are possible. The distribution of the errors for all values, shown in Figure 4-17, has the nice bell curve shape that we desire. For all values, $\overline{|E|} = 1.1$ and $\sigma = 1.3$. For extreme values, the distribution has a rather odd distribution. The reason for it is not yet clear. For extreme values, $\overline{|E|} = 1.3$ and $\sigma = 1.6$, which are good results.

We increase L to 0.6. The distribution of errors is shown in Figure 4-18. The distribution for extreme values is approaching that of a bell curve. The statistics for all values and extreme values are $\overline{|E|} = 1.1$, $\sigma = 1.4$ and $\overline{|E|} = 1.2$, $\sigma = 1.6$, respectively. These results are quite excellent, especially as the mean absolute error for extreme values approaches that of all values. At this threshold level, 91% of $\mathcal{T}$ is predictable. Note however that σ for all values oddly deteriorated from the previous case.

Figure 4-19, the distribution of errors with $L = 0.7$, is interesting. The shape of the distributions still is somewhat Gaussian, but have lost its smoothness. Statistically, the results are no significant improvement over that of $L = 0.6$. Furthermore, at this

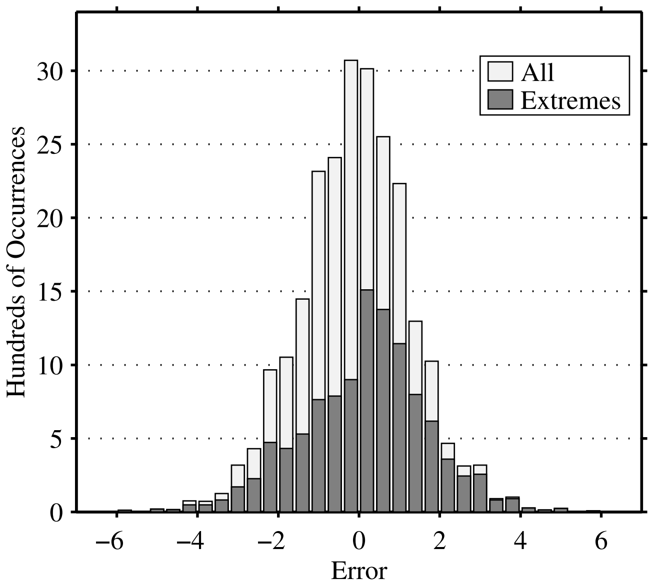

Figure 4-18: The distribution of errors for the Constrained Pearson r algorithm with $L = 0.6$.

L level, predictions can be made for only 70% of the values in $\mathcal{T}$.

4.8.3 Variations

A number of variations of the Constrained Pearson r algorithm were also tested. These variations were similar in nature to those described in Section 4.6.3:

- Through the use of different scaling functions, the methods for generating weights were altered.

- The methods for generating weights were also altered through the incorporation of commonality.

- The threshold level, L, was varied.

The use of these variations have not yet produced any significant improvements.

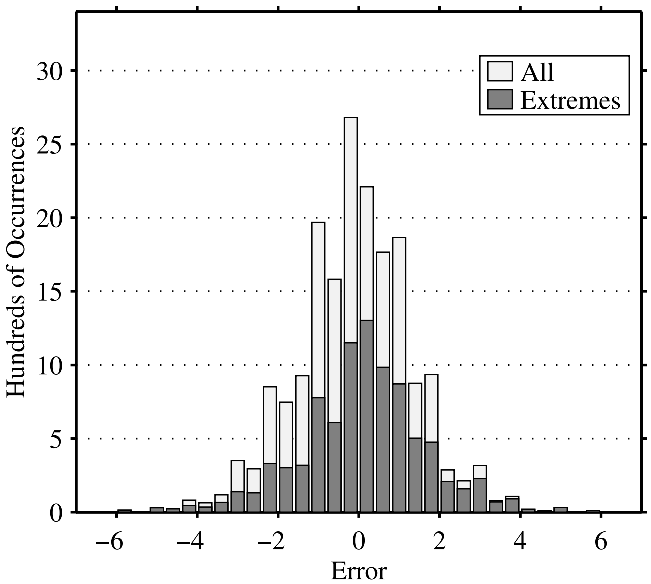

Figure 4-19: The distribution of errors for the Constrained Pearson r algorithm with $L = 0.7$.

4.9 The Artist-Artist Algorithm

The preceding algorithms discussed have dealt with measuring and employing similarities between *users*. Karl Sims proposed the use of correlations between *artists* to generate predictions[11].

The idea is simply an inversion of the previous methodologies. Say Ringo needs to predict how a user, Murray, will like "Harry Connick, Jr." Ringo would examine the artists that Murray has already rated. It weighs each one with respect to their degree of correlation with "Harry Connick, Jr." The predicted rating is then simply a weighted average of the artists that Murray has already scored.

4.9.1 The Algorithm

One implementation of such a scheme, which employs β-values, is presented below.

The correlation between artist g and h is described by

$$\beta_{gh} = \frac{\sum(A_g - 4)(A_h - 4)}{\sqrt{\sum(A_g - 4)^2 \times \sum(A_h - 4)^2}} \tag{4.18}$$

Using this metric, we tested a number of algorithms. A simple algorithm to predict the value of some t_{ij}, how user i would rate artist j is:

- Precompute the matrix of all β values between every two artists in the database.

- Precompute the matrix of all *offset* values between every two artists in the database. The offset between two artists is the difference between their mean scores. The offset, o_{gh} between artist g and artist j is calculated as

$$o_{gh} = \overline{A_g} - \overline{A_h} \tag{4.19}$$

  Note that $o_{gh} = -o_{hg}$.

- Generate the neighborhood $\mathcal{G}_j$, composed of those artists m that the user has rated and whose $\overline{\beta_{mj}}$ are greater than a certain threshold L.

- Compute the *weight*, w_{mj}, of each artist m in $\mathcal{G}_j$. One method for generating weights from the β values is:

$$w_{mj} = \begin{cases} \beta_{mj}^2 & \text{if } \beta_{mj} > 0 \\ -\beta_{mj}^2 & \text{if } \beta_{mj} < 0 \end{cases} \tag{4.20}$$

  In all cases, $\overline{w_{iu}} \leq 1$.

- The predicted value p_{ij} for t_{ij} is then a weighted average of all the ratings by user i for the "similar" artists in $\mathcal{G}_j$:

$$p_{ij} = \frac{\sum_k^{G_j} \beta_{kj} \times (s_{uj} - o_{kj})}{\sum_k^{N_j} \beta_{kj} \times c_{uk}} \tag{4.21}$$

4.9.2 Results

Figure 4-20 shows the distribution of all the artist–artist correlation coefficients. Puzzling spikes appear at ± 1.0. The reason for their existence becomes clear once you examine Figure 4-21. This is the distribution of only those coefficients where κ, the

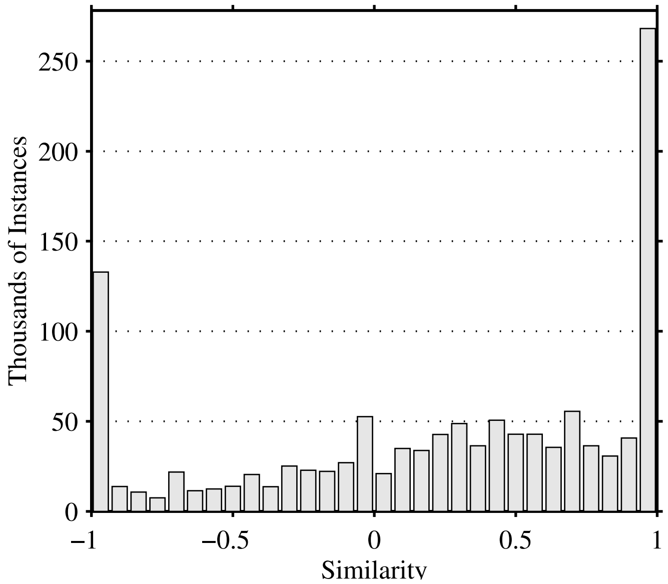

Figure 4-20: The distribution of artist–artist correlation coefficients.

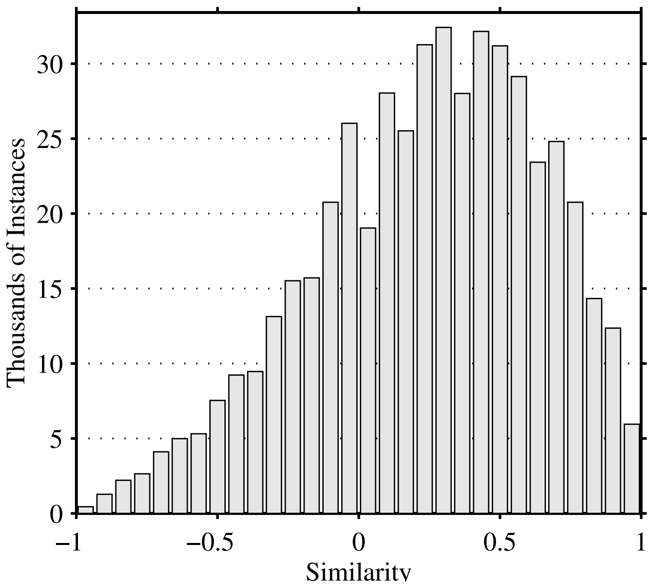

Figure 4-21: The filtered distribution of artist–artist correlation coefficients.

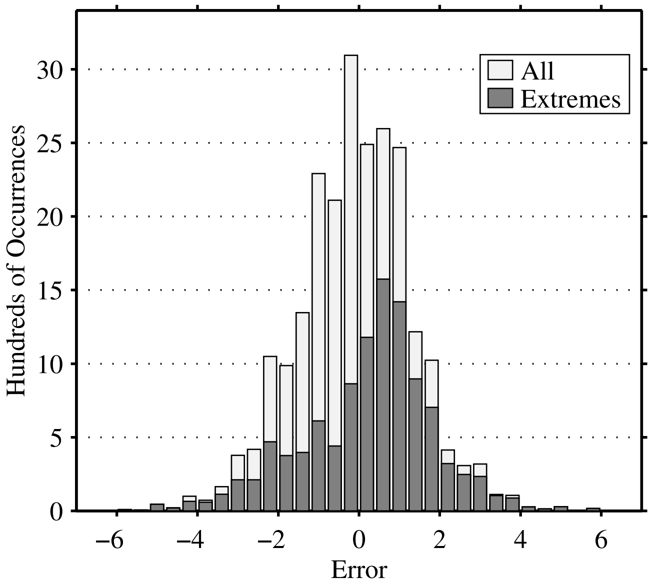

Figure 4-22: The distribution of errors with $L = 0.6$, $\kappa > 5$

number of users who have rated both the artists being correlated, is at least 5. The distribution now is more like what one would expect. There were a sizable number of artists which had extreme correlations due to their low sample of data.

When the algorithm is executed with no threshold ($L = 0$), the algorithm behaves very poorly, its distribution is very much like our base algorithm.

Surprisingly, the algorithm performs worse, with approximately a 0.1 increase in $\overline{|E|}$ on average, when offsets are included. The algorithm behaves much better when offsets are ignored.

We now use a threshold $L = 0.6$. In addition, we require that $\kappa > 5$, meaning that we ignore all β-values where the two artists have less than 5 users in common. Offset values are also not included. The error distribution is shown in Figure 4-22. $\overline{|E|} = 1.1$ and $\sigma = 1.4$ for all values, while $\overline{|E|} = 1.3$ and $\sigma = 1.6$ for extreme values. Predictions for 89% of the target set were generated.

Now, $L = 0.7$, and again $\kappa > 5$. The error distribution is shown in Figure 4-23. $\overline{|E|} = 1.1$ and $\sigma = 1.4$ for all values. Furthermore, $\overline{|E|} = 1.1$ and $\sigma = 1.5$ for extreme values! However, this setting could only generate predictions for 65% of the target set.

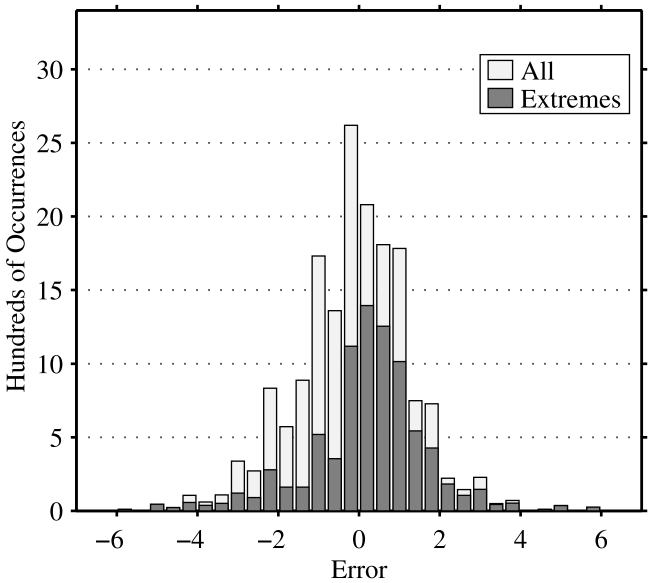

Figure 4-23: The distribution of errors with $L = 0.7$, $\kappa > 5$

As the threshold level L is increased from 0, the distribution of errors slowly changes form, from the double-hill distribution, like the Base Algorithm, to a more Gaussian distribution, as in Figure 4-22. This transition can be more cleanly seen when you look at the error distribution for an intermediary position, like $L = 0.5$, in Figure 4-24. Notice the left hill beginning to sink away.

4.10 Measure of Confidences

The last criterion to be discussed is the issue of confidences– can we create some measure of how reliable a particular prediction is? For the algorithms discussed above, three characteristics may be of value in our different confidence functions:

- Commonality, as described in section 4.6.3.

- The number of user scores to generate the prediction. In other words, the number of users in the neighborhood $\mathcal{N}_u$ who have rated the artist in question. In the case of the Artist-Artist algorithm, the number of artists scores used to generate a prediction would be used.

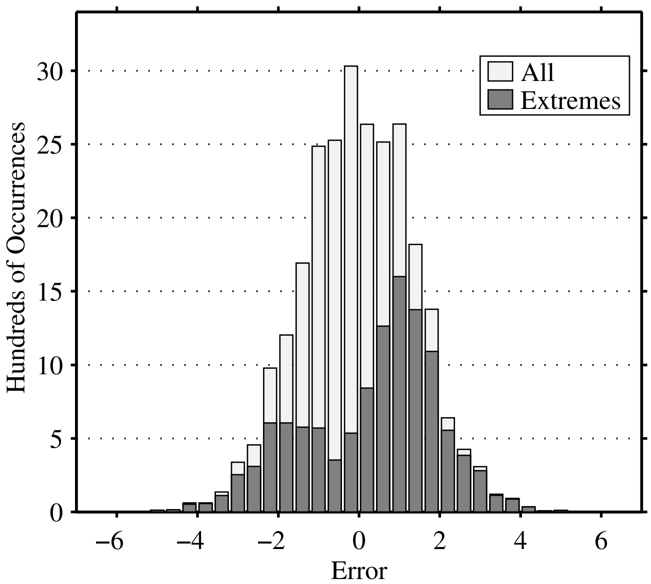

Figure 4-24: The distribution of errors with $L = 0.5$

- The relative weights of the scores used to generate the prediction.
- The *variance* in the scores used to generate the prediction. In other words, how well the scores of the similar profiles agree.

Commonality did not prove to be a significant indicator of confidence for the three user-user algorithms. Commonality was generally inversely proportional to the mean absolute error only when the number of artists in common was less than some small number, usually around 5. Therefore, some limited thresholding based upon the commonality, as was done in the Artist-Artist algorithm, can improve accuracy. However, there is not enough correlation between commonality and correlation to be useful.

The number of scores used seems to be the most useful indicator. Figure 4-25 shows a scatter plot of error vs. the number of users used to generate the prediction for the MSD algorithm. As the number of user scores used to generate a prediction increases, the deviation in error decreases significantly.

With regards to the use of weights and variance for the computation of confidence levels, not enough is known at this time.

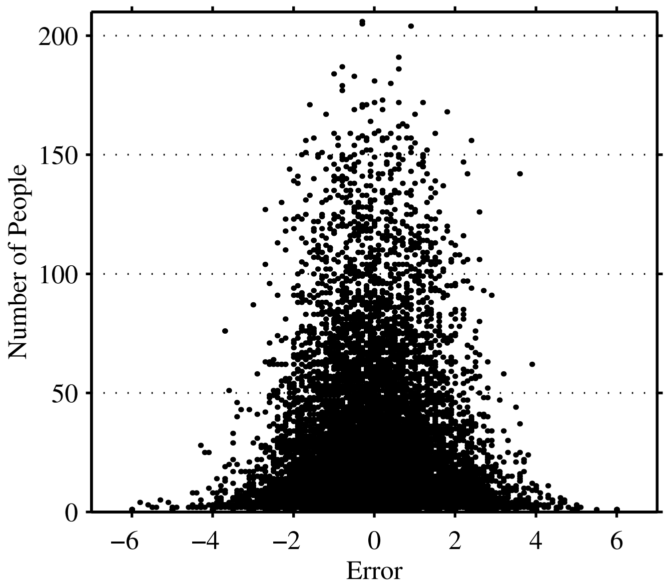

Figure 4-25: The scatter plot of the error vs. the number of people consulted to make the prediction.

4.11 Summary

A summary of some of our results are presented here in table form (Table 4.2). Overall, in terms of accuracy and the percentage of target values which can be predicted, the Constrained Pearson r algorithm seems to work the best. Keep in mind that these results are only indicative of the relative value of these algorithms with respect to this data set. Their effectiveness may indeed vary depending upon the problem domain.

Method	All $|E|$	All σ	Extremes $|E|$	Extremes σ	$\%\mathcal{T}$
Base Case	1.3	1.6	1.8	2.0	90
MSD, $L = 2.0$	1.0	1.3	1.2	1.6	70
Pearson r	1.1	1.4	1.5	1.7	99
Pearson r, $L = 0.35$	1.0	1.3	1.4	1.6	99
Pearson r, $L = 0.5$	1.0	1.3	1.3	1.6	95
Pearson r, $L = 0.65$	1.1	1.4	1.3	1.6	73
Pearson r, $L = 0.75$	1.1	1.5	1.3	1.7	41
Constrained Pearson r, $L = 0.5$	1.1	1.3	1.3	1.6	97
Constrained Pearson r, $L = 0.6$	1.1	1.4	1.2	1.6	91
Constrained Pearson r, $L = 0.7$	1.1	1.3	1.3	1.6	70
Artist-Artist, $L = 0.6$	1.1	1.4	1.3	1.6	89
Artist-Artist, $L = 0.7$	1.1	1.4	1.1	1.5	65

Table 4.2: Summary of results.

Chapter 5

Qualitative Results

"THERE ARE THREE KINDS OF LIES: LIES, DAMNED LIES, AND STATISTICS."

— B. Disraeli

—⋆—

5.1 The Human Element

Ultimately, what is more important than the numbers in the previous chapter is the human response to the technology. Ringo is a tool, to be used by real people. The idea of SF is relatively straightforward. What is more interesting than the algorithms is the human element. What happens when you add the human into the loop? How will they respond to Ringo? Will they find it competent? Will they trust it?

As of this writing over 2000 people have used Ringo. Our source for a qualitative judgment of Ringo is the people themselves. The Ringo system operators have received a *staggering* amount of mail from users— questions, comments, fortunately very little criticism and unfortunately a bit too many bug reports. In addition, we periodically send the users our electronic publication, The Ringo Newsletter, to keep our users informed on Ringo's status, and to answer commonly asked questions.

Before discussing the feedback we have received from people, we present two disclaimers:

- Our users are not representative of the world at large. We do however have users who hail from educational institutions, companies, and who-knows-where in the case of the commercial net-access users. There are users from 27 countries on 5 continents, including New Zealand, South Africa, Israel, India, all of Scandinavia and even Croatia. Still, they are all people who have Internet access and have some free time on their hands.

"I'm impressed by the ease of use and the apparent accuracy of your system."

- We have only heard from the people who *chose* to write us. What conclusions can be drawn about that group, we do not know. So it is possible that many people have had bad experiences but chose not to write us.

5.2 Emergence

Ringo's competence "emerges" with time, as more people use the system. Understandably then, in the first couple weeks of Ringo's life, Ringo was relatively incompetent. During these days we received quite a few messages letting us know how poorly Ringo was performing.

From the onset, Ringo used a slightly cruder version of the Mean Squared Difference algorithm presented in the previous chapter. We were never sure it would work well, since early on we never had enough data with which to test it. Despite the early feedback, Ringo continued on, its algorithm intact.

Slowly, the feedback began to change. More and more often we received mail about how "unnervingly accurate" Ringo was, and less about how it was incorrect.

A SF system by nature performs better overall as the number of users grows. The greater the number of users, the greater the chance that close matches will be found between people[1]. Although we did no analysis with data sets of different sizes to confirm this phenomenon, the positive trend in user feedback would indicate this to be true.

However, I believe many people are still disappointed by Ringo's performance. We are often told that a person must do one or two iterations of rating artists before Ringo

[1] It really is essentially a dating service.

is *really* accurate. A person would rate the initial set, then receive predictions. If the user knows any of the predicted artists are not representative of their personal tastes, they rate those artists. This will radically alter the members of the user's "similar user" neighborhood. After one or two of these iterations, Ringo works satisfactorily. Many people however do not figure this out or do not bother to carefully read the help file, which explains this procedure. They then get their initial predictions back, and are discouraged.

The major reason for this is the non-intelligent mechanism Ringo has for generating the initial list of artists for the user to rate. What is needed is some simple clustering to determine "critical" artists, which, when rated, will rapidly define a user's tastes. Thus, an optimal list of artists can be sent to users to rate. Then, after a user rates this optimal list, the predictions they receive back should be more satisfactory.

Beyond the recommendations, there are other factors which are responsible for Ringo's great appeal and phenomenal growth. The additional features, such as being a user-grown database, and the provisions for reviews and dossiers add to its functionality.

Foremost, however, is the fact that Ringo is not a static system. A user does not use Ringo once and then stop using it. The database and user base is continually growing. As it does, Ringo's recommendations to the user changes. As the user rates more items, again Ringo's recommendations change. For this reason, people enjoy Ringo and use it on a regular basis.

"Ringo is great fun! It's already pointed me to several albums that I wouldn't have known about otherwise."

5.3 Critical Mass

One lesson to be learned is the volume of data required. The system must reach a certain critical mass of people in any particular genre before it works well overall. Likewise, a certain number of "key" artists must be rated by someone before Ringo has a good fix on that person's tastes. The amount of data will vary from problem domain to problem domain. Music, in my opinion, is a much more complex and varied problem space than movies, for example. There are many genres, and combinations of tastes. Thus, the critical mass of users is quite large. For Ringo, we reached stability in most of the bigger genres much later than I expected. Based upon user feedback, we reached "critical mass" for mainstream music first, after roughly 250 users were in the system. Then came college rock and alternative. Since then Ringo has also reached critical mass for Industrial Music. Other genres have not yet been verified.

"A friend of mine, who is also a frequent RINGO user, celebrates his 30th birthday tomorrow. If you could send me ... a suggestion from RINGO for [him], I could go out and actually buy those CDs for him first thing tomorrow morning."

5.4 The Black Box

People have varying degrees of "black box" views of Ringo. Ringo has many users who are extremely technically knowledgeable. However, we are also very fortunate to have users who were not knowledgeable of algorithms or even computers[2]. This provided ample opportunity to see how different people interact with Ringo.

Before the project was publicized, all of Ringo's messages to the user were written in the first person: "I recommend that you check out these artists..." There was however some concern that users may think that the system would be able to communicate in natural language. All of Ringo's messages were then changed to be more impersonal. Also, the instructions make the format for messages explicitly clear, and announce that e-mail to Ringo is not read by a human but parsed, rather, by a computer program.

Despite this, Ringo receives a number of messages where users would write messages to "Ringo" in English. Users have also rated artists with scores such as, "**5** or **6**", or "**7+**", not realizing it uses a simple parser.

It is difficult *not* to refer to Ringo as an entity. I, too, although I'm possibly the most cynical about Ringo, having created it and knowing its insides, usually refer to Ringo as an entity, as does most everyone else. But here is this system which operates 24 hours a day, answers e-mail, grows, and in a way "learns." It is very easy to start referring to it as a live entity.

Even among people with technical backgrounds, there are varying perceptions

[2] Which shows you the degree to which the Internet is reaching "ordinary" people.

of how Ringo works. Ringo's help file explains briefly the concept behind Social Information Filtering. I believe most people understand the basic idea. However, we often receive questions indicating an extremely strange model of Ringo's workings that people have constructed. When it comes to personal assistant agents, this actually may be a desired quality. If people understood the inner working in detail, then they would expect a certain, desired result for every action they take. This would mean that the agent is simply a fancy macro, not an "intelligent" assistant. Of course, in the case of Ringo, no one could possibly predict exactly what result certain actions will produce. The data space is much too rich for a person to comprehend.

5.5 Trust

There have been a few occasions where people have written us saying that they bought a Compact Disc based on Ringo's suggestions. Fortunately, everyone who wrote back telling us of their purchases indicated that they liked what they had just bought.

"I just got my first recommendation back from Ringo! Incredibly right on target!!! Ringo basically picked out bands that I totally love, which I hadn't rated yet, and rated them about what I would have!

I'm totally psyched!"

This kind of incident brings up the issue of *trust*. Any personal assistant agent is little more than a fancy macro unless its users trust its decisions and allow it to make decisions on their behalf.

What factors lead a user to trust an agent?

Trust must be earned. This occurs when the agent makes recommendations that are correct. In Ringo's case, it makes a few predictions. The user might recognize some of the artists recommended. If he or she agrees with Ringo's assessment, then Ringo has just earned some trust.

To a lesser extent, the presentation of the agent can certainly have an effect upon a user's reaction to the agent. The agent's "image" will help determine how open the user is to accepting the agent. The perception that the agent is an intelligent entity, whose inner workings are not fully understood, may help. It provides a "human," boundless quality to the system. People now begin imagining that the system is doing all kinds of complex operations en route to providing assistance. Paradoxically, I also believe that the fact that it is artificial may also help lead some users to trust it. The system's mechanical nature can contribute to the sense of that it is somehow reliable and consistent, unlike people.

Once the person trusts the agent, for whatever reasons, only then can the agent be valuable and useful.

5.6 The Mirror

"Some of the groups in the database are truly bizarre I was quite happily surprised. Please keep the service going. Its the best thing since 23.5 oz. cans of ice tea."

Before Ringo became a real system, a smaller system was tested. The database at the time only had 25 profiles that I had solicited from friends and lab-mates. Naturally the predictions were generally terrible, given such a small user-base, unless you were lucky and matched 1 of the 25 people closely. Despite this, during a Lab Open House, I gave small presentations about the system to any visitor who was willing to listen.

Over the course of the open house, a couple people sat down individually, scored some artists, and received suggestions from the system. I was always quick to explain that the user base is quite small, and therefore would not work very well. As it turned out, I did not need to make excuses. People would see the suggestions, and say things like "That one's about right, that's right, well that one I'd probably rate lower, but wow, it's working." I watched them score artists, then I saw the poor suggestions, and was thinking, "You've *got* to be kidding." It was as if they expected it work, and therefore it did. Over these past two months of Ringo's life, we've received e-mail from people who similarly *expect* Ringo to be correct. Why do people have this much faith?

A fellow Media Labber commented that possibly it is because people see Ringo's suggestions as a "reflection of themselves." Ringo takes their likes and dislikes, and from them produce these suggestions. Everything that the user tells the system effects the output. Once you have decided that there must be a logical connection between what you have told the system about your tastes and the system's recommendations, then you are much more inclined to believe the predictions to be right. The phenomenon in a way is like horoscopes. If you believe that they are true, then you can find a way to interpret them so that they are true.

Part of the reason for Ringo's success is due to the problem domain. People often strongly identify with their musical tastes. Thus, they are much more liable to have a strong opinion on Ringo's performance, positive or negative.

We occasionally receive very skeptical, cynical e-mail, which essentially says something like "There is no way your system can predict what I like. My tastes are too unique and varied for it to be right." Maybe it will, maybe it won't. It depends upon whether there are people like them in the system. But, the person *a priori* does not want to believe that Ringo can work. Admitting that it did would be admitting that their tastes are not as unique as they believed.

As long as people believe that an agent is a reflection of themselves, they will enjoy using the agent. Who doesn't like looking in a mirror occasionally?

Chapter 6

Future Research

"REMEMBER WHEN WE SAID THERE WAS NO FUTURE? WELL, THIS IS IT."
— B. Reg, *Big Time Television*

—⋆—

The only regret I personally have with the Ringo project is that I must leave it just as the user interface is becoming stable. There is ample data and growth, and one can now really delve into many different aspects of Social Information Filtering.

However, were I still involved with Ringo, here are some areas of future research that I would love to explore:

- **Rigorous analysis of Algorithms.** Due to the fact that the Ringo project has only existed for a few months, much of the algorithm analysis has been executed by manual, human exploration. No one should believe that the actual numbers discussed in Chapter Four for threshold levels, etc., are some kind of physical constant. They will vary widely, depending upon the problem domain, the volume of data, the particular user base, and other factors. It is a relatively straightforward matter to have a meta-SF-algorithm continually testing different parameters to find the optimal settings, or possibly even select the optimal algorithm, for that point of time.

  It may be interesting to try "evolving" characteristics of Social Information Filtering algorithms through genetic algorithms. The fitness functions can be well-defined, and the data space is extremely structured and simple in form, which are features amenable to Genetic Algorithms.

- **Virtual Users.** Ringo currently compares every person's profile to that of everyone else in the system. There are certainly more intelligent ways to go about seeking the most similar users. Rather than doing a comparison to every other user, we would like to determine methods for rapidly finding other similar users.

  Over time, as each user is entered into the system we normally would compute all its similarity measures, with each of the other N_u users in the system. The similarity measures among the previous N_u in the system have already been computed.

  We would like to speed up this process by limiting the number of profile comparisons. One possible approach is through the use of what I call "Virtual Users."

  It is possible that any particular person reduces to a linear combination of generic tastes, within some error.

  A Virtual User is a one of a collection of generic (and sub-generic) profiles, such that any user's tastes can be reduced to a linear combination of these virtual users. The Virtual Users would be continually evolved over time. We thus can search this space of Virtual Users to find the best matches for our new user. Then, we can either just use these virtual profiles, or for more detail use the real users who form the selected virtual profiles.

- **Clustering.** Using the data collected about user tastes, it should be possible to apply clustering to the artist space and find groups of related artists. I suspect that groups will not only form based on similarity in style, but also on similarity of quality. For example, there would form a group of "bad" artists, such as New Kids on The Block, Vanilla Ice, Pat Boone, and Billy Ray Cyrus, the four lowest rated artists currently in Ringo (with mean scores ranging from

1.3 to 1.5). The reason for this is that not enough people have rated them positively for the system to be able to distinguish them. With time, of course, as more and more people rate an artist, if all mean scores approach **4**, groups based on quality will slowly disappear.

Just out of general interest, it would be very interesting to find out what kind of genres and sub-genres clustering produces. If that was not enough, it can also greatly speed our algorithms, by helping us focus on particular search spaces. For example, we might determine that a particular user only likes certain genres, and limit our searches primarily to those areas. Or we may compare users with respect to a particular genre. For example, for a particular user, recommendations for Jazz might be based on Miles' profile, and recommendations for Pop are based upon Michael's.

Knowledge of clusters will also allow more kinds of queries. For example, users will be able to ask "Recommend an artist that is like *Dead Can Dance*", or even "Recommend something that is unlike anything I've ever heard." These additions will greatly enhance Ringo's functionality to the user.

- **Emergence.** As an SF system collects data, it becomes more competent and more "intelligent." The nature of this "emergence" needs to be studied. How much and what kind of data needs to be in the system before it can do various particular tasks?

- **Confidence.** More work needs to be done to find better indicators of the reliability of any particular prediction. What factors, such as commonality, the number of users, etc., correlate highly with the error? Once adequate factors are determined, how can we translate them into one confidence value?

- **Distributed Ringo.** Ringo as it stands is a centralized server. It might also have been implemented as a distributed system. Every user would have its own agent on their home machine. This agent would communicate with other agents on the net, looking for similar people, and retrieving information accordingly. Once it finds a few agents that it "trusts", it can receive referrals to other agents from those agents. Such a scenario will eliminate the need for a centralized server and distribute computation among different machines.

- **Integration with Content-Based Filtering.** Both Content-Based Filtering and Social Information Filtering have their strengths and weaknesses. As the example with Mel picking a movie in Chapter 2 demonstrated, the two processes are complementary in human practice. In computer applications, the two approaches may also be used in conjunction for greater effect. So far the two approaches have been implemented separately, but soon the integration of the two will need to be explored.

- **Avoiding Abuse.** Unethical individuals or groups may enter fictitious users in order to bias the system. For example, a company could flood the system with fictitious users of different tastes who all happen to like the company's latest album release. Methods for detecting and dealing with such violations need to be developed.

Chapter 7

Conclusions

"We seek the truth, no matter who gets hurt."
— E. Dorkin, *Truth or Dare*

—⋆—

The bottom line is whether Social Information Filtering works as a general approach to information filtering. The answer is yes. Ringo has demonstrated that SF methods can be used to provide personalized information. It has been tested and used in a real-world application and received a positive response. The same techniques can be used to recommend books, movies, news articles, products, and more.

SF is still very much in the infantile stages of research. However, due to its simplicity and its potential for immediate application, development of the technology should occur very rapidly.

With Ringo, much of the development of algorithms occurred while the system was operational. This was of course out of necessity, since we needed real data to test the algorithms, and the best way to collect the data was to have the system available for use. It is also appropriate since SF research cannot easily be separated from the problem domain. At this stage of research, the two are coupled. Therefore, the results of any analysis must keep the problem domain in mind.

The analysis presented focused on how accurate the techniques were, *on average*. However, depending upon the end application, the "fitness function" may vary. For example, Ringo only suggests the eight artists that it predicts the user would like the most. Given this, the objective of an ideal SF for Ringo is to predict a small number of artists that the user would like, *with no room for error.* It must be able to recommend eight artists with great confidence. Had Ringo presented a person a list of forty suggestions, you are then assuming that the user will apply his own "filtering" to that list, and you can afford to have some errors. If the application is news filtering, and a list of articles is provided the reader, then it is acceptable to have some misses. When you are recommending purchases, you cannot have any misses.

Furthermore, a SF scheme may perform well on average, but in a commercial application, there is no room for one unsatisfied customer. The individual customer does not care if the system works on average, but rather, if it works for them.

The research and application of SF will lead to a better understanding of human tastes, with respect to certain areas. From a certain viewpoint, it is as sociological as it is technological.

Sometimes I am asked whether I think such a technology could affect people's buying habits in the future. My answer always is, "without a doubt."

Consumers today are very cynical. We are bombarded by advertisements specifically designed to convince us that we *need* something that we know we do not need. So an ordinary consumer views ads with a lot of distrust and cynicism.

Enter the social filtering agent, like Ringo, making suggestions for people. This neutral, mechanical agent is not designed to deceive anyone. It does not serve the needs of some big corporation or studio. Its purpose is only to serve its user, with no selfish intents. The agent "knows" its user, and everything it says is tailored around the user's past history. It further bases its suggestions upon the tastes of other, ordinary people. These other people enter their likes and dislikes to the system honestly, with no intent to deceive. The user can read reviews or comments written by a variety of other, ordinary people. Today people love to read Consumer Reports or listen to Siskel and Ebert or ask their best friend before making a purchase or watching a movie. This agent facilitates such a flow of information.

The end result is that the user trusts its agent, and makes purchases accordingly. As long as the agent is consistently right, the user will continue to buy things based upon what the agent says. They will be much more willing to buy things, as the risk of making a bad purchase is minimized. The way people buy things will be totally different.

The issue of trust is recurring in the field of autonomous agents. If Ringo is any indication of things to come, agents can become reliably integrated into the lifestyles of people.

Appendix A

Ringo's Help File

```
   ALL ABOUT RINGO

Hi there, Music Lover! Welcome to Ringo, your personal music
recommendation service.  There is a world of music out there, some of
it meant for *you*, some of it not. What should you check out, what
should you avoid? Ringo is here to help answer that question, to
recommend artists that you personally will like, to help you wade
through the ocean of CDs that's out there. Plus, there are lots of
other cool features. All described in this somewhat lengthy and
growing help file.

QUESTIONS, COMMENTS, ETC. to ringo-sysops@media.mit.edu
RATINGS AND ALL OTHER COMMANDS to ringo@media.mit.edu

.. . .. . .. . .. . .. . .. . .. . .. . .. . .. . .. . .. . .. . .. .


This document is organized as follows:

    * Quick Overview of (Most) Commands

    * Commonly Asked Questions

    * Detailed descriptions of (All) Commands

-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-

    QUICK OVERVIEW
```

Here is a quick description of most of the commands available, just to get you started playing. Refer to the rest of the help file for more info.

You can put as many commands as you like in the body (NOT the subject) of an e-mail. The only restriction is that each command be on a line by itself.

Command	Description
profile	What artists/albums have you rated, along with reviews?
suggest	What artists should you check out?
avoid	What artists should you avoid?
charts	Who are the highest (and lowest) rated artists?
rate	Send more artists to rate, so Ringo knows you better.
? "artist name"	Ask Ringo for a prediction about a specific artist.
dossier "artist name"	Send the entire file about this artist.
full	Send a list of all artists, and your ratings of them.
subscribe	Subscribe to the Ringo Newsletter (default setting).
unsubscribe	Unsubscribe to the newsletter.

Double quotes around the name of the artist is essential!
(Ex: dossier "Depeche Mode")

The detailed description below include information on how to:

* Subscribe to the Ringo Newsletter
* Add artists and albums to the database
* Submit information to be included in dossiers
* Include reviews of artists
* and more details about the above commands.

-=-

COMMONLY ASKED QUESTIONS

Before we get down to details, let's answer some commonly asked questions:

Q: How does it work?

A: Well, how do you do select albums ordinarily? You listen to songs that some D.J. plays, or you hear about stuff from your friends who have tastes similar to your own.

That's how it's done here. People all over the Internet, from all over the globe, tell me about their listening tastes. Ringo then finds people who are similar in their tastes to you. If they really like some artists that you haven't heard yet, Ringo will recommend those artists to you.

Which is why the more users in the system, the better the recommendations. So spread the word!

Q: How does Ringo know who I am?

A: People are identified according to their usernames. So be consistent as to where you send e-mail from.

Q: My friend wants to get on the ringo bandwagon. What should I say?

A: Tell your friend to send an e-mail to ringo@media.mit.edu with only the word 'join' in the body of the e-mail, and s/he will be sent the latest greatest survey, and all the relevant info.

Q: Why isn't "So and so" in the database?

A: Right now there are over 3000 artists in the database, and its growing. If you did not see your favorite band in the initial survey, look in the full listing of the database (instructions on how to get it are in this file). If "So and so" still is not there, you can add them to the database (instructions again included).

Q: How long does it take Ringo to respond to mail?

A: E-mail is processed in batches, and responses are compiled once an hour.

Q: Can I ask for recommendations? Can I add items? Write reviews? Get cool info about artists and their albums? Can I help update info about artists?

A: Yes yes yes! Read the descriptions of the commands below.

Q: What's next?

A: More services will be added as time goes on, and Ringo will constantly be modified to better serve you. You'll be kept up to date in the newsletter that you will receive, should you subscribe to it. I will give you one hint.. think 'Ringo on the World Wide Web!'

Q: Who is building Ringo?

A: The Agents Group at the MIT Media Lab. The original idea was formulated by graduate student Carl Feynman and professor Pattie Maes. Graduate Student Upendra Shardanand is responsible for the design and original implementation of Ringo. Undergraduate Lee Zamir has aided with system development and database maintenance.

Q: This system is great! What can I contribute?

A: Well, this is part of our ongoing research. So let us know how well it works. If you find a bug, please let us know ASAP. Give us some feedback (to ringo-sysops@media.mit.edu) on what you would like to see added or changed. Thanks.

Q: How does Ringo handle individual album ratings vs. overall artist ratings?

A: It depends... If two users who have both rated an artist's albums are being compared, then each album rating is matched up between the two users. If, on the other hand, one user has rated an artist's albums while the other has just rated the artist overall, then the first users album ratings are averaged. This average is taken as his/her overall rating for that artist.

Q: What do I do if I find an error in the database?

A: The ringo system operators have the ability to fix any problems that might exist in ringo's database. Examples of problems include duplicate albums, duplicate artists, or non-existent entries that are currently in the database. If you find such an error, please send mail to ringo-sysops@media.mit.edu (NOT ringo@media.mit.edu) notifying us of the error. It may take a few days for us to fix the error but it will get done.

Q: What do I do if I find an error in my profile?

A: Users have the ability to correct most errors in their personal profiles. To re-rate and artist/album, just rate it as you normally would and the new rating will replace the old one. If you want to clear a rating entirely, just rate it a 0 (that's the number zero, not the letter O). The same thing goes for reviews. You can change a review by reviewing the artist/album as you normally would. In the case of artists, the new review replaces the old one. In the case of albums, the new review gets concatenated to all the other album reviews for that artist.
At present, however, users are unable to clear a review. We are working

```
   on this feature and as soon as it becomes available, we will let you know.

Q: I got back my suggestions, and Ringo is partly or way off base!

A: If this happens, it means that based on your current profile, you
   have been matched with some users who you really shouldn't be
   matched with. To solve this problem, just rate the artists that
   Ringo suggested to you. For example, if ringo suggests to you "Band
   X" with a score of 6.3 and you hate them, then rate "Band X" with a
   low score, and in the next wave of suggestions, ringo should know
   you much better.

========================================================================

        ALL OF RINGO'S COMMANDS
        -----------------------

You can put as many commands as you like in the body (NOT the subject)
of an e-mail.  The only restriction is that each command be on a line by
itself.

help
----
..sends you this long message.

subscribe
---------
..to our periodic newsletter, with cool information about RINGO, a
personalized report on *you*, including recommendations, new artists
for you to rate.  Newsletters will be sent out once every couple
weeks.  By default, you are subscribed to the newsletter when you join
the system.

If you have something interesting and music-related to say and want to
say it through the newsletter, let us know.

unsubscribe
-----------
..to the newsletter.

profile
-------
..gives you a list of all the artists you have rated, and what you gave them.

suggest
```

```
------
..sends you a list of artists that you have to check out.

avoid
-----
..sends you a list of artists that you really want to steer clear of.

? "artist name"
---------------
..How does Ringo think you'd rate the artist? Ringo will send back a
prediction of how much you'll love/hate that artist. Plus, Ringo will
include any reviews written about the artist by people with tastes
like yours.  (Ex: ? "Fu-Schnickens")

charts
------
..sends you the latest, hippest charts. Top 20, Bottom 20, and whatever else..
If you have an interesting chart you want sent to everyone, submit it to us
(ringo-sysops@media.mit.edu) and we'll put it in.

rate
----
..sends you a list of 100 more artists to rate. The more artists you
rate, the better Ringo gets to know your tastes. Plus, the better
Ringo can make predictions for other folks. So once in a while, if
you're bored, rate some more artists.  Ringo keeps track of what
artists you've rated (or left blank) recently, so you get a new set of
artists every time.

dossier "artist name"
--------------------
..sends you a document about the artist. Basically some simple
statistics, such as what ratings the artist's albums get on average,
and any extra info, like discographies, tour dates, pointers to FAQs,
mailing lists, news, or whatever else that is submitted by users.

If you want to add stuff to an artist's dossier, send the stuff to be
added to ringo-sysops@media.mit.edu.

full
----
.. sends you a (long, and getting longer) list of all the artists in
the database, plus your ratings (if you rated them).

add
```

```
---
.. denotes that the following artists and albums are to be added to
the database. PLEASE use this command with *great care*, as all adds are
executed immediately.  If you stop adding albums and execute other
commands, use the 'end' command. The format is as follows:

(rating) "Artist" "Album"

   When adding artists, you must include at least one album.  You can
rate each album individually as you go along, if you please.  You must
include album titles. Use the following examples for reference:


add
  "Prince"          "Graffiti Bridge"
  "Gabriel, Peter"  "So"
  "Gabriel, Peter"  "Passion"
          "Gabriel, Peter"  "Security"
7 "Beatles, The"    "Rubber Soul"
end

Adds are immediately entered into the database and used by all, so for
correctness and consistency *please* use the following guidelines:

 1. Last name first.
 2. Put the "The" at the end, like "Beatles, The"
 3. No CD Singles, No Soundtracks, No Bootlegs, promo albums,
    compilations, etc. You get the idea. Just ordinary albums.
 4. Consult the 'full' listing of artists before adding artists.
 5. Consult the listing of an artist's albums before adding albums.
 6. Watch your spelling, capitalization, and the like.

end
---
.. used to end things. For example, to end a section of albums to be
added. Not necessary if the thing you wish to end is the last item in
the your e-mail.

-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-

When you enter ratings
----------------------

When you receive a list of artists (or artists and their albums), there
are certain things you can do:
```

"artist"

Rate the artist. Using that nifty scale:

    7: BOOM! One of my FAVORITE few! Can't live without it.
    6: Solid. They are up there.
    5: Good Stuff.
    4: Doesn't turn me on, doesn't bother me.
    3: Eh. Not really my thing.
    2: Barely tolerable.
    1: Pass the earplugs.

Any name of an artist or an album must be in quotes.
(Ex: 7 "Enigma")

For discerning users: If you find yourself often saying "This group is somewhere between 5 and 6," you can express this in-betweenness by using a decimal rating (up to one place).
(Ex: 1.5 "Gerardo")

"artist" "album"

Rate the artist's album. Using the same nifty scale.
(Ex: 7 "Campbell, Tevin" "I'm Ready")

* "artist"

What albums does the artist have? Ringo will send back a list of the artist's albums, and you can rate them individually if you like.
(Ex: * "U2")

When given an artist to rate, you also have the option of using the '?' command to ask Ringo for a prediction, or using the 'dossier' command (both commands were described in the previous section).

Entering Reviews

When you receive suggestions of artists that you really love or really hate from Ringo, it includes reviews written by people like yourself. So if you really, really feel strongly one way or another about an artist, write a short review of them, their work, their albums, etc. Then, when that artist is recommended (or not) to someone else, your review will be included.

Here's how you do it:

```
7 "Prince"
[Prince is the MAN!. I loved that Purple Rain thing he did. He's
kinda short, though. Blah blah blah.  - mday@time.com]

"Tritt, Travis"
[Travis is like the king of country. You gotta check out his
Christmas album.. a classic. Blah blah blah.]

"Beatles, The" "Help!"
[This album is John's desperate cry for.. well, help.]
```

You can rate the artist or not.. your call. Everything between the '[]' will be reproduced as is. If you want people to know who wrote it, sign your name. If not, don't.

Reviews are stored and sent to folks automatically, as needed.

-=-

Hope this helps. If anything is unclear, or you have suggestions on how to improve Ringo, write us: ringo-sysops@media.mit.edu.

Finis

Bibliography

[1] David Anderson. Personal Communication.

[2] Entertainment Decisions, Inc., Evanston, IL. Clair V. Business Portfolio.

[3] Carl Feynman. Nearest neighbour and maximum likelihood methods for social information filtering. Massachusetts Institute of Technology, internal report, Fall 1993.

[4] Goldberg, et al. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, pages 61–70, December 1992.

[5] Will Hill. Personal Communication.

[6] Paramount Interactive. Movie select. Software Application.

[7] Brian Reid. Usenet flow analysis for jun 94: Who stores how much news. USENET article news.groups #107733, July 1994.

[8] Resnick, et al. Grouplens: An open architecture for collaborative filtering of netnews. Sloan Working Paper, February 1994.

[9] Robert Rosenthal and Ralph Rosnow. *Essentials of Behavioral Research: Methods and Data and Analysis*. McGraw Hill, second edition, 1991.

[10] Beerud Sheth. A learning approach to personalized information filtering. Master's thesis, Massachusetts Institute of Technology, February 1994.

[11] Karl Sims. Personal Communication.