Evaluating the Accuracy and Utility of Recommender Systems

vorgelegt von
Master of Science
Alan Said

Von der Fakultät IV – Elektrotechnik und Informatik –
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften
– Dr.-Ing. –

genehmigte Dissertation

Promotionsausschuss:

Vorsitzender: Prof. Dr. Müller
Berichter: Prof. Dr. Albayrak
Berichter: Prof. Dr. Cremonesi
Berichter: Dr. Jain

Tag der wissenschaftlichen Aussprache: 21. März 2013

Berlin 2013
D 83

Evaluating the Accuracy and Utility of Recommender Systems

Thesis for the Degree Doktor der Ingenieurwissenschaften
Alan Said

Fakultät Elektrotechnik und Informatik
TECHNISCHE UNIVERSITÄT BERLIN
Berlin Germany, 2013

Evaluating the Accuracy and Utility of Recommender Systems
ALAN SAID

Review Committee
Chairperson: Prof. Klaus-Robert Müller
Reviewer: Prof. Sahin Albayrak
Reviewer: Prof. Paolo Cremonesi
Reviewer: Dr. Brijnesh J. Jain

Keywords: Recommender Systems, Evaluation, Collaborative Filtering, User Study

ACM Categories: H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval - Information filtering, Retrieval models, Selection process; H.3.4 [**Information Technology and Systems Applications**]: Decision Support

Fakultät Elektrotechnik und Informatik
Technische Universität Berlin
Sekretariat TEL 14
Ernst-Reuter-Platz 7
D-10587 Berlin
Telephone + 49 (0)30-314 74000/02

Berlin, Germany. 2013

Evaluating the Accuracy and Utility of Recommender Systems

Alan Said

Berlin 2013

Technische Universität Berlin
DAI-Labor
Fakultät Elektrotechnik und Informatik
Sekretariat TEL 14
Ernst-Reuter-Platz 7
D-10587 Berlin
Telephone +49 (0)30-314 74000/02

Abstract

Recommender systems have become a ubiquitous feature on the World Wide Web. Today, most websites use some form of recommendation to heighten their users' experience. Over the last decade, vast advancements in recommendation have been done, this has however not been matched in the processes involved in evaluating these systems. The evaluation methods and metrics currently used for this have originated in other related fields, e.g. information retrieval, statistics, etc. For most cases, these evaluation processes are able to show how well a recommender system performs – to some point. However, after a certain threshold, it is not often clear whether a lower error, or higher accuracy metric accounts for an actual quality improvement.

This dissertation focuses on the research question how can we further estimate whether a measured accuracy level actually corresponds to a quality improvement from the user's perspective, or whether the measured improvement is lost on the end user. We introduce some of the concepts related to recommendation quality and user perception, and continue on to show that currently widely-used evaluation metrics do not capture the quality of recommendation when the algorithm is specifically tuned to offer recommendation of a higher diversity. Following this we present a formalization of the upper limit of recommendation quality, a *magic barrier* of recommendation, and evaluate it in a real-world movie recommendation setting.

The work presented in this dissertation concludes that current recommendation quality has outgrown the methods and metrics used for the evaluation of these systems. Instead, we show how qualitative approaches can be used, with minimal user interference, to correctly estimate the actual quality of recommendation systems.

Zusammenfassung

Empfehlungssysteme sind heutzutage ein allgegenwärtiger Bestandteil des World Wide Web. Viele Webseiten nutzen Empfehlungssysteme, um Benutzerfreundlichkeit und das Nutzererlebnis zu verbessern. Während der letzten zehn Jahre wurden viele Fortschritte in der Forschung zu Empfehlungssystemen gemacht, allerdings haben die Methoden zur Evaluierung damit nicht Schritt gehalten. Die Methoden und Metriken zur Evaluation, die derzeit verwendet werden, sind in den meisten Fällen von anderen, verwandten, Forschungsrichtung wie z.B. Information Retrieval oder Statistik, adaptiert. Bis zu einem gewissen Grad klappt dieses Vorgehen auch. Mit den aktuell verwendeten Evaluationsverfahren kann man bewerten, ob eine neue Methode generell funktioniert oder nicht. Man kann allerdings nicht bewerten, ob Verbesserungen wie ein niedrigerer Fehler oder eine verbesserte Genauigkeit in der realen Anwendung wirklich zu einer verbesserten Benutzerfreundlichkeit oder einem besseren Nutzererlebnis führen.

Diese Arbeit beschäftigt hauptsächlich sich mit der Frage, wie man die Evaluation verbessern kann, um besser Vorhersagen machen zu können, ob Verbesserungen eines neuen Empfehlungsalgorithmus auch beim Nutzer ankommen.

Wir benennen einige der wichtigsten Konzepte im Zusammenhang mit der Empfehlungsqualität und Nutzerwahrnehmung und zeigen dann, dass die derzeit genutzten Evaluationsverfahren nicht die Qualität von Empfehlung erfassen, wenn der Algorithmus speziell auf Empfehlungen mit höherer Diversität ausgelegt ist. Danach präsentieren wir eine Formalisierung der *Magic Barrier*, einer oberen Grenze für die beste erreichbare Empfehlungsqualität aufgrund von fehlerhaften oder schwammigen Nutzerfeedbacks. Wir evaluieren die Magic Barrier mittels einer Nutzerstudie, in einer Anwendung zu Filmempfehlungen mit realen Nutzer und Daten.

Die in dieser Dissertation präsentierten Ergebnisse führen zu dem Schluss, dass die aktuell verwendeten Evaluierungsmethoden und Metriken nicht Schritt halten mit aktuellen Ergebnissen in der Forschung zu Empfehlungsalgorithmen und der erreichten Qualität. Daraus folgend, werden im Rahmen der Arbeit qualitative Ansätze und deren Anwendung beschrieben, die bei minimaler Einbindung des Nutzers, den tatsächlichen Nutzen eines Empfehlungsalgorithmus einschätzen können.

Acknowledgments

Writing this dissertation would not have been possible without the continuous encouragement and support of a number of people.

During my time as a PhD student at the TU Berlin I was fortunate to work at the DAI-Lab with an amazing group of people, especially those in the Competence Center for Information Retrieval and Machine Learning. Without them, this dissertation would not have been possible. I would especially like to express my gratitude to Dr. Brijnesh J. Jain for continuous discussions, challenges and valuable insights. Other, current and former, members of CC IRML, including Till Plumbaum, Dr. Robert Wetzker, Dr. Leonhard Hennig, Prof. Ernesto W. De Luca, Sascha Narr, Benny Kille and Dr. Frank Hopfgartner for their valuable support, discussions and feedback. This gratitude also extends to my supervisor, Professor Sahin Albayrak who gave me the opportunity, environment and support to conduct the research which provides the basis for this dissertation.

I would like to extend my gratitude to all my coauthors over the last years for making me challenge my knowledge and push me in interesting directions, this includes (in no particular order) Dr. Ben Fields, Dr. Domonkos Tikk, Dr. Brijnesh J. Jain, Till Plumbaum, Sascha Narr, Prof. Ernesto W. De Luca, Dr. Robert Wetzker, Dr. Leonhard Hennig, Winfried Umbrath, Benny Kille, Dr. Martha Larson, Yue Shi, Prof. Paolo Cremonesi, Dr. Shlomo Berkovsky.

Additionally, I am very grateful to the people who proofread this dissertation at different stages of completion, this includes Dr. Brijnesh J. Jain, Dr. Frank Hopfgartner, Sascha Narr and Dr. Andreas Lommatzsch.

Lastly, and most importantly I would like to thank my mum, dad and sister. Thanks!

List of publications

The content of this thesis builds on the following publications:

I. (Ch. 3) Alan Said, Brijnesh J. Jain, Sahin Albayrak - Analyzing Weighting Schemes in Collaborative Filtering: Cold Start, Post Cold Start and Power Users. *Proceedings of the 27th ACM Symposium on Applied Computing* (SAC'12) ACM. [SJA12]

II. (Ch. 3) Alan Said, Ernesto W. De Luca, Benjamin Kille, Brijnesh J. Jain, Immo Micus, Sahin Albayrak - KMulE: a framework for user-based comparison of recommender algorithms. *Proceedings of the 2012 ACM International Conference on Intelligent User Interfaces*, 2012, ACM. [Sai+12d]

III. (Ch. 4) Alan Said, Benjamin Fields, Brijnesh J. Jain, Sahin Albayrak - User-Centric Evaluation of a K-Furthest Neighbor Collaborative Filtering Recommender Algorithm. *Proceedings of the 16th ACM conference on Computer Supported Cooperative Work* (CSCW'13) ACM. [SFJ13]

IV. (Ch. 4) Alan Said, Brijnesh J. Jain, Andreas Lommatzsch, Sahin Albayrak - Correlating Perception-Oriented Aspects in User-Centric Recommender System Evaluation. *Proceedings of the fourth Information Interaction in Context Symposium* (IIiX'12) ACM. [Sai+12a]
Nominated Best Poster

V. (Ch. 4) Alan Said, Benjamin Kille, Brijnesh-Johannes Jain, Sahin Albayrak - Increasing Diversity Through Furthest Neighbor-Based Recommendation. *Proceedings of the WSDM'12 Workshop on Diversity in Document Retrieval* (DDR '12), 2012. [Sai+12c]

VI. (Ch. 5) Alan Said, Brijnesh J. Jain, Sascha Narr, Till Plumbaum - Users and Noise: The Magic Barrier of Recommender Systems. *Proceedings of the 20th International Conference on User Modeling, Adaptation, and Personalization* (UMAP'12) Springer-Verlag. [Sai+12f]
Awarded Springer Best Paper Award

VII. (Ch. 5) Alan Said, Brijnesh J. Jain, Sascha Narr, Till Plumbaum, Sahin Albayrak, Christian Scheel - Estimating the Magic Barrier of Recommender Systems: A User Study. *Proceedings of the 35th international ACM SIGIR conference on Research and development in Information Retrieval* ACM. [Sai+12b]

Other publications by the author outside of the scope of this dissertation:

- Alan Said, Brijnesh J. Jain, Sahin Albayrak - A 3D Approach to Recommender System Evaluation. *Proceedings of the 16th ACM conference on Computer Supported Cooperative Work* (CSCW'13) ACM. [SJA13]

- Alan Said, Shlomo Berkovsky, Ernesto W. De Luca - Movie Recommendation in Context. *ACM Trans. Intell. Syst. Technol. (TIST)*, Vol.4, No.1, 2013, ACM. [SBDL13]

- Alan Said, Domonkos Tikk, Yue Shi, Martha Larson, Klara Stumpf, Paolo Cremonesi - Recommender Systems Evaluation: A 3D Benchmark. *Proceedings of the Workshop on Recommendation Utility Evaluation: Beyond RMSE (RUE '12)*, 2012. [Sai+12e]

- Alan Said, Domonkos Tikk, Andreas Hotho- The Challenge of Recommender Systems Challenges. *Proceedings of the sixth ACM conference on Recommender systems (RecSys '12)*, 2012. [STH12]

- Christian Scheel, Alan Said, Sahin Albayrak - Semantic Preference Retrieval for Querying Knowledge Bases. *Proceedings of the 1st International Workshop on Entity-oriented and Semantic Search* (JIWES '12), 2012, ACM [SSA12]

- Alan Said, Shlomo Berkovsky, Ernesto W. De Luca - Group Recommendation in Context. *Proceedings of the 2nd Challenge on Context-Aware Movie Recommendation*, 2011, ACM. [SBDL11]

- Alan Said, Ernesto W. De Luca, Sahin Albayrak - Inferring Contextual User Profiles - Improving Recommender Performance. *Proceedings of the 3rd Workshop on Context-Aware Recommender Systems*, 2011. [SLA11a]

- Alan Said, Till Plumbaum, Ernesto W. De Luca, Sahin Albayrak - A Comparison of How Demographic Data Affects Recommendation. *Adjoint Proceedings of the 19th international conference on User modeling, adaption, and personalization*, 2011. [Sai+11]

- Alan Said, Ernesto W. De Luca, Sahin Albayrak - Using Social and Pseudo Social Networks for Improved Recommendation Quality. *Proceedings of the 9th Workshop on Intelligent Techniques for Web Personalization & Recommender Systems*, 2011. [SLA11b]

- Alan Said, Shlomo Berkovsky, Ernesto W. De Luca - Putting Things in Context. *Proceedings of the Workshop on Context-Aware Movie Recommendation*, 2010, ACM. [SBDL10]

- Alan Said - Identifying and utilizing contextual data in hybrid recommender systems. *Proceedings of the fourth ACM conference on Recommender Systems*, 2010, ACM. [Sai10]

- Alan Said, Jérôme Kunegis, Ernesto W. De Luca, Sahin Albayrak - Exploiting hierarchical tags for context-awareness. *Proceedings of the third workshop on Exploiting semantic annotations in information retrieval*, 2010, ACM. [Sai+10]

- Alan Said, Ernesto W. De Luca, Sahin Albayrak - How social relationships affect user similarities. *Proceedings of the IUI'10 Workshop on Social Recommender Systems*, 2010. [SDA10]

- Alan Said, Robert Wetzker, Winfried Umbrath, Leonhard Hennig - A Hybrid PLSA Approach for Warmer Cold Start in Folksonomy Recommendation. *Proceedings of the RecSys '09 Workshop on Recommender Systems and the Social Web*, 2009. [Sai+09]

- Robert Wetzker, Winfried Umbrath, Alan Said - A hybrid approach to item recommendation in folksonomies. *Proceedings of the WSDM '09 Workshop on Exploiting Semantic Annotations in Information Retrieval*, 2009, ACM. [WUS09]
  Awarded Best Technical Paper

Contents

List of Tables

List of Figures

“It is a very sad thing that nowadays there is so little useless information.”
– Oscar Wilde, *A Few Maxims For The Instruction Of The Over-Educated*, 1894

CHAPTER 1

Introduction

Information search and consumption went through a drastic change with the advent of the Internet and the World Wide Web (WWW). Suddenly, vast information resources became available at one's fingertips, most often without noticeable delays, costs or effort. This made resources which were previously difficult or impossible to access, available instantly. Perhaps not unexpectedly, this was not without drawbacks. The problem of finding the right information suddenly turned on its head – instead of having too little information, there was now simply too much of it. The task of harnessing these large amounts of information soon became too big to handle manually, which in turn led to a rapid acceleration in the development of intelligent information management techniques online, e.g. Web search.

Information retrieval did however not begin with the World Wide Web, it is older than both the World Wide Web and the Internet. However, the increasing popularity of, and amount of information on the Web certainly accelerated the scientific advances in intelligent information management [Nat].

In the early stages of the Web, the process of finding relevant websites among the millions of pages available was achieved by text-based queries. This process included indexing of pages according to their content, the indexes were then matched to the queries posted by users. This type of *content-based* search was subsequently replaced by more elaborate search algorithms as such became available, e.g. PageRank [Pag+99]. As the research area of online information retrieval matured, Web services automating the processes involved in identifying relevant information began to appear. Prominent examples of this included the Lycos[1] search engine [Lyc] which was among the first search engines with automated Web crawlers and automated text classification. Another example, which later became known as *The Internet Movie Database* (IMDb), initially started as the Usenet group `rec.arts.movies` where users would write and manually send movie reviews and recommendations to each other [Nee]. Shortly, this turned into a more or less automated service, which was followed by the first version of the IMDb[2] recommendation website.

With the new technologies developed for the Web, accessing richer content and information became easier. Ultimately, this led to a paradigm shift in information consumption. Before the Web, obtaining information was a lengthy process, requiring planning in advance and considerable time for execution. Creating information and making it accessible for others was even more cumbersome and few people went through the trouble to share their creations. Today, publishing information requires little resources, is practically effortless, is instant, and has a potentially world-wide audience. Creating and publishing content is a natural part of many netizen's lives. The vast amounts of consistently created information, whether in the form of professionally redacted websites (e.g. newspapers), crowd-sourced content (e.g. wikis, forums, etc.) or nonreviewed personal blogs, changed the basics of (digital) information consumption and interaction.

More recent Web developments, i.e. the transition from the initial, consumption-oriented static Web to the more dynamic and content creation-focused Web 2.0, have led to the Web becoming the provider of not only traditional static media, e.g. newspapers, linear TV, etc. Today, users of the Web can download or stream music, movies, TV, etc. without having to adhere to a certain cinema or TV schedule. The information is available at any time, anywhere, for anyone. The large amounts of innovative services such as YouTube[3], Netflix[4], iTunes[5], Last.fm[6] and Spotify[7] have created cheap, or free, means of accessing movies, music, magazines and other media. The result of this has been a change in the consumption patterns of media [BHS10]. Instead of waiting for information (TV, newspapers) to come to the consumers; the consumers themselves dictate how, when, and where information is to be consumed. It is up to the service to provide what the users want, otherwise they simply turn to a service which

[1] `http://www.lycos.com`
[2] `http://www.imdb.com`
[3] `http://www.youtube.com`
[4] `http://www.netflix.com`
[5] `http://www.itunes.com`
[6] `http://www.last.fm`
[7] `http://www.spotify.com`

better is better at satisfying their needs.

Instant availability of resources changes the way people interact with them [AB12]. Whereas previously, consumption of an item required planning ahead, leaving the confinements of one's home and purchasing, renting or borrowing a physical item; today the sought for media is available within seconds or minutes, practically removing the planning and anticipation phases. The context has changed, the effort involved has been removed, the personal cost of a false positive choice[8] can be simply, cheaply, and quickly replaced by a more suitable item. Planning has been reduced to browsing, and in that process a new problem has been created; that of finding the one of few mostly relevant pieces of information in the vast amount of available information. This change in information interaction has created the need for better means of assessing what data is data to what user, with short (preferably no) waiting time.

This is where personalized *Recommender Systems* have seen their largest use case on the Internet. Recommender systems have seen an enormous growth in popularity online, becoming something of a ubiquitous and omnipresent tool available on almost all major websites[9]. Not only have recommender systems become common, they have seen a massive boost in measurable accuracy. With events such as the Netflix Prize[10], development and research in the area of recommender systems grew and popularized the topic, also bringing a greater knowledge about these systems to the greater public.

Over the last decade, many initial problems related to recommender systems were solved. Current state-of-the-art systems generally perform at adequate accuracy levels [Her+04], in terms of rating prediction or ranking (sometimes also referred to as *top-n* recommendation). Research topics in recommender systems today often focus on specific sub-problems, such as *context-awareness*, *group recommendation*, *social recommendations*, etc. These systems and approaches are commonly evaluated using techniques which predate the current state of information accessibility. Many of these methods have been transferred verbatim from information retrieval, statistics and other related fields. Some of the most common evaluation techniques include measures often used in information retrieval and pattern recognition (precision, recall), economics and statistics (Root-Mean-Square Error), and signal detection (receiver operating characteristic). These measures might, however, not always correctly reflect the *actual* levels of accuracy of the evaluated recommender systems, as experienced by the end user. They simply were not designed to capture some of the aspects

[8] For example, a poorly picked movie or a book that turns out to be bad.

[9] In June 2012, the top 20 websites on Alexa's Top 500 ranking (`http://www.alexa.com/topsites/global`) all offered some form of personalized recommender system to their users. Sites on the list include Google, Facebook, YouTube, Yahoo!, Amazon, etc.

[10] `http://www.netflixprize.com`

important in well-functioning recommender systems [Her+04], e.g. a higher precision does not need to directly translate to the fact that that recommendations will be perceived as better by the users.

1.1 Motivation

Following the introduction of services such as IMDb [Nee], Amazon[11], GroupLens [Res+94], etc., online recommender systems became an everyday tool for Web users. A tool which provided them with much needed help in discovering relevant information. As stated earlier, these systems have become ubiquitous in various fields from shopping (Amazon) to movies (Netflix) and music (Last.fm), simplifying personalized online information access.

Over the last two decades, a vast amount of research in recommender systems has lead to great progress in terms of prediction accuracy and scalability. One of the initial problems in the recommender systems research community was the lack of appropriate data. Without this, it was difficult, if not impossible to optimize and evaluate recommendation algorithms properly. This was however counteracted by initiatives such as the Movielens[12] movie recommendation website, which provided the research community with a large real life dataset to be used specifically for research personalized recommender systems [Lud]. Subsequently, several more datasets, such as the Netflix Prize dataset, created opportunities for researchers and practitioners to conduct experiments at even larger scales.

Today, the majority of the work on recommender systems is based on *top-n* (or ranking) recommendation or *rating prediction*; the former requires bi/unary interaction data between users and items, whereas the latter requires a dataset with ratings, and is thus very commonly used in the movie recommendation domain. Both types of recommender system evaluation also requires these datasets. For the process of evaluation, these datasets are commonly split into training and a testing subsets. Usually, the test set consists of items which are deemed to be representatives of *true positive* recommendations, e.g. movies which a user has rated highly, items which are often purchased, and other items labeled in a positive fashion. The training set consequently includes the remainder of the data and, as the name suggests, is used for training of the recommendation algorithm. Other means of evaluation, e.g. through user studies, are less commonly used. User studies specifically due to the nature of the methods and experiments involved, i.e. offline evaluation can be quickly run (and re-run) to

[11] `http://www.amazon.com`

[12] `http://www.Movielens.org`

evaluate different parameters and settings etc. A user study on the other hand requires the involvement of actual users, i.e. a different analytical approach and experimental setup [Kni12; STH12].

In top-n evaluation, the accuracies of the algorithms are measured by their ability to recommend items from the test set when recommending n items. This scenario is commonly evaluated through the classical information retrieval metrics *precision* and/or *recall*. The conviction brought by these metrics is: the more items from the test set that are recommended, the better the recommendation algorithm performs.

In rating prediction, the accuracy is instead measured through the ability to correctly predict the ratings given to items in the test set. Metrics used for rating prediction include *Root-Mean-Square Error* (RMSE) or *normalized discounted cumulative gain* (nDCG). Both of which express the rating prediction error by the recommender system, i.e. how poorly or how well the recommender was able to estimate the withheld ratings. In this scenarion, prediction accuracy can either be measured across the rating scale, or on a subset of items, e.g. only positively or only negatively rated items[13], depending on the recommendation scenario or goal.

Both top-n evaluation and rating prediction-based evaluation build on several assumptions which could have detrimental effects on recommendation algorithms optimized solely based on these techniques [APO09; Cre+11a; Her+04; Hil+95]. These assumptions are:

- there is an absolute *ground truth* which the recommender system should attempt to identify,
- users are primarily interested in the items which have received the *highest ratings*,
- higher top-n accuracy or lower rating error levels translate to a higher *perceived usefulness* from the users.

The assumptions make the claim that it is the datasets, not the users, that dictate how well or poorly the recommender system performs. Even though the datasets reflect the historical interactions of the user, the assumption neglects that users, people, do not always act predictably.

By ground truth in the first assumption is meant that people rate items[14] conse-

[13] In the Netflix Prize, the to-date most popular recommendation challenge, the rating prediction quality was measured across the rating scale.

[14] In the scope of the dissertation these will always be represented by movies.

quently, that *taste is deterministic* provided that all relevant information about the users and their context(s) is available. This assumption also states that, in addition to being deterministic, users additionally also are consistent in their rating behaviors. The second assumption relates to the notion of *higher is better*, i.e. that users are always interested in the highest rated items and that a low rated item is always an incorrect. Furthermore, this assumption suggests that the ratings are static, i.e. that a user always has the same opinion regarding an already rated item. The last assumption, related to perceived usefulness, simply stipulates that the more an algorithm can mimic the history of a user, in terms of item interactions and ratings, the better will it be received by the user.

This evaluation paradigm neglects the inconsistent and non-deterministic aspects of human beings [Czi89]. In doing so, it does not necessarily correctly reflect the users' perception of the accuracy and quality of a recommender system.

This dissertation shows that perceived usefulness as well as noise estimates in underlying data are factors which need to be taken into consideration when evaluating recommender systems. The focus of the work is specifically on movie recommendation systems which allow users to rate items, e.g. Movielens, Netflix, Moviepilot, Filmtipset[15], etc.

Additionally, the work aims at highlighting some of the problems (e.g. inconsistencies in quantitative vs. perceived evaluation) involved in, and drawbacks of, current evaluation techniques used with rated datasets, as well as proposing methods of overcoming the shortcomings of today's recommender systems evaluation.

Furthermore, the dissertation provides a theoretical model for the maximum optimization possibility using offline evaluation of recommender systems.

The overall goal of the work presented throughout this dissertation is to create simple and *cheap* (in terms of additional work required when compared to standard evaluation) methods of estimating the quality of recommender systems. The methods apply to both offline evaluation setting, e.g. using training/test set splits for predicting historical data, as well as in online settings, e.g. where users are directly involved in the evaluation through user studies or questionnaires.

[15] `http://www.filmtipset.se`

1.2 Thesis Contributions

This thesis attempts to answer the following research challenges:

1. How does popularity bias affect recommendation and evaluation?
2. Is diversity in recommendation possible to achieve without sacrificing a recommender system's predictive and subjective accuracy?
3. Do metrics commonly used for recommender systems evaluation accurately correlate to the subjective perceptions of the users who receive the recommendations?
4. Is there an overlap in the perception of different subjectivity metrics?
5. Is there a ground truth, an optimal prediction accuracy value, towards which recommenders should be optimized?

For these purposes, a number of experiments and studies have been conducted and evaluated in the scope of this dissertation.

The main contributions from these are:

- The correlation of offline evaluation towards the perceived usefulness of said recommendations in order to accurately estimate the perceived quality of a movie recommender system in an offline evaluation scenario. Due to the simplicity of offline evaluation measurements of quality, being able to assess quality of a recommender system, as perceived by the user, prior to a large scale user study can lower the time and work load involved in planning, conducting and iterating user studies. This contribution is presented in Chapter 3. The nature of the popularity concept makes it difficult to mitigate in online evaluation. Even when using elaborate mitigation approaches, offline evaluation will in most cases show a positive bias towards popular items, despite the fact that they might have a lower utility for the end users.

- A standard method for online evaluation of recommender systems through user studies to be used when planning and evaluating user studies for estimating the quality of a recommender system. This contribution is presented in Chapter 4. In contrast to the concepts in Chapter 3, this work describes an online study in order to subjectively evaluate the utility of recommendations.

- A correlation of standard concepts used to measure user-centric evaluation attributes, e.g. how is the concept of diversity vs. the concept of novelty perceived by users. This contribution is presented in Chapter 4.

- The formalization of a maximum level of rating prediction accuracy that a recommender should attain in offline evaluation before further optimization becomes useless, i.e. an RMSE level to strive towards which is not necessarily 0. This concept is known as the *Magic Barrier* of recommender systems and is presented in Chapter 5. This concept is infeasible to prove as it would require infinite re-rating for each user-item pair, the study conducted does however show the concepts involved and how an estimate of the magic barrier can be found through minimal user interference. The work is supported by a user study conducted in a real-life recommendation system.

Each research chapter (Chapters 3 to 5) focuses on one or several of these challenges and contributions.

1.3 Approach

The approaches used in this thesis correspond to the current state-of-the-art in recommender systems evaluation. Results have been achieved by qualitative as well as quantitative studies of datasets and users in virtual and real-world environments and situations, with specific focus on the latter. The aim has been to create methods for correctly evaluating recommender systems in the current state of research and development.

The process involved the following steps:

- Initial tests of current evaluation methods were conducted in the scope of Chapter 3 to serve as a basis for the work which followed.

- Analyses of currently used measures were performed in the scope of Chapter 4 in order to understand the correlation of traditional (offline) evaluation methodologies and the more novel (online) methods used today.

- Correlations of common measures used in modern evaluation systems were analyzed in the scope of Chapter 4 in order to create conceptual guidelines for online evaluation of recommender systems.

- Finally in Chapter 5 a concept for the lower boundaries of traditional evaluation techniques (RMSE minimization) was formalized and evaluated in a real-world movie recommender engine.

Together these steps form the outline of the dissertation.

1.4 Application Scenarios

The potential applications of evaluation approaches for recommender systems in rated item sets are manifold. This dissertation focuses on the user-centric evaluation of movie recommender systems. The evaluation approaches are general and can be applied in other recommendation scenarios, but the focus of this dissertation is specifically movie recommendation. User-centric recommender systems evaluation attempts to create an evaluation model for recommender systems not only based on traditional information retrieval methods, but also on qualitative aspects such as perception quality, and statistical aspects such as the magic barrier.

Classical information retrieval- and machine learning-based evaluation needs to be adapted to the concept of personalized recommendation, e.g. statically applied measures such as precision and recall, or rating error minimization methods like RMSE might reflect the general trend in which the quality of a recommendation algorithm is headed. However for optimized evaluation, several additional attributes need to be taken into consideration. The following section describes potential application scenarios for the work conducted in the scope of this dissertation.

Perception-based evaluation. When evaluating a recommender system's quality, a perception-oriented user study should be conducted in order to accurately estimate the perceived quality of the system, no matter its performance in terms of information retrieval-based metrics like precision and recall. Offline evaluation of a recommendation algorithm shows a general trend of the accuracy value for the recommendation system. A user study conducted in a deployed system provides a more accurate estimate of the recommendation algorithm as experienced by the system's users. Discrepancies between these two have been shown to exist and an in-place evaluation procedure (e.g. A/B tests) provides a more accurate picture of the recommender system's performance.

Perceived quality estimates. When planning and deploying a user study, certain commonly measured factors overlap; in order to minimize the effort on the users, some of these could be combined into one. By lowering the user effort, the probability of completing the user study is increased, which ultimately should lead to a more accurate quality estimate of the deployed system.

Self-assessment of a recommender system's quality. When deploying a recommendation system, the quality of the system's predictions can be measured through a comparison of predicted ratings and users' actual given ratings, e.g. the prediction error. Due to users' being nondeterministic by nature, the prediction error does not only correspond to the system's quality, it also encompasses the noise generated by users when rating. In order to accurately self-assess a system's quality, a set of re-ratings can be collected, i.e. asking users to rate already rated movies again. Doing this for an extended period of time will allow for a more accurate assessment of the system's magic barrier, and subsequently produce an estimate of the lowest possible prediction error the system should be able to attain.

This list is not intended to present the complete set of application scenarios, it serves as an example of the possible use cases the content of this dissertation can be applied to.

1.5 Outline of Thesis

The rest of this thesis is structured as follows. Chapter 2 starts by positioning the thesis in the current recommender system research landscape and presents the challenges tackled in the scope of this dissertation.

Chapter 3 continues to present the problems in the research field, focusing on issues related to evaluation of recommender systems based on popularity and skewness in the datasets that are created by recommender systems. Following a presentation and discussion of the problem, the chapter continues with a section on related work in this field and presents empirical results dealing with mitigation of popularity-induced effects in recommender systems. Some of the issues reported in this chapter are subsequently dealt with in Chapter 4.

Chapter 4 presents the second topic discussed in the scope of this dissertation, namely *perception-related* aspects of recommender system evaluation. The chapter discusses both human-computer interaction-related aspects as well as information retrieval oriented issues which affect the evaluation of recommender systems. Analogous to the previous chapter, Chapter 4 first introduces the topic and the challenges encountered, followed by an overview of the current state-of-the-art in related research. It continues with a description of the experiments and results performed and obtained on perception-oriented evaluation and correlation in the scope of the chapter. Finally, it concludes the results and lists a path for potential future work directions.

The last research chapter, Chapter 5, gives an overview of issues related to evaluation and optimization of recommender systems, specifically in the context of minimizing rating prediction errors (RMSE). Similarly to the previous chapters, Chapter 5 starts with an overview of the topic of recommender system optimization, positions the chapter in the current relevant state-of-the-art research. Following this, the research conducted in the scope of the chapter is presented. The chapter presents a brief conclusion and possible directions for future work.

Finally, the last chapter, Chapter 6, concludes the work presented in this dissertation by summarizing the contributions, presents currently ongoing and planned future work, and finalizes this dissertation with some closing remarks.

Chapter 2

Background

This chapter introduces the concepts and issues related to recommender systems and their evaluation. The three main areas presented are; a brief historical recap of recommender systems, a brief overview of the most commonly used datasets in recommender systems research, an introduction to some of the technical aspects of recommendation and a broad overview of aspects related to the evaluation of recommender systems.

2.1 General Overview

Online recommender systems on the Web are, as mentioned in the introduction of Chapter 1, actually older than the Web itself [BL; Gol+92; Nee]. Early recommender systems often had quite trivial ways of generating recommendations, by either manually aggregating lists of items [Nee], or by populating lists by comparing the size of the word intersection between a liked document and candidate documents [BS97], or simply by picking the most popular items (in this context, *documents* are a subset of the set *items*).

Work relevant to this thesis is however more recent. During the last decade, recommender systems have gone through an extensive evolution process. Specifically, during the three years (2006–2009) of the Netflix Prize, which created incentives and possibilities for recommender systems research [BL07; Ben+07]. For the duration of the Prize, Netflix invited participants to make use of a large dataset from their movie rental service, containing more than one hundred million movie ratings given by their users to a subset of items from their catalog. The challenge was to beat Netflix's rating prediction algorithm, *CineMatch*, by 10%. Each year a cash prize of $50,000 was awarded to the leading team. The goal was achieved in 2009, and the winning team was awarded $1,000,000 for their effort [Cop09; Net06].

The challenges faced in recommender systems are focused on how to deliver interesting items to the users, whether like in Netflix's case the items are movies, or other various items like on Amazon. Positioning recommender systems in the data science spectrum, one definition is *query-less information search*, or *implicit search*. The contrary, *explicit search*, is the kind of information retrieval one utilizes when posting a search query in a search engine.

The Netflix Prize not only gave recommender systems researchers the opportunity to work on a large real-life dataset, it also brought the field of recommender systems to the public's eye, which undoubtedly led to an increase of the amount of research conducted in the field[1]. Netflix was of course not the only source to real life datasets of significant size. The Grouplens[2] research group has offered datasets from its Movielens[3] movie recommendation website for an extended period of time. At roughly ten million ratings[4], roughly a tenth of the size of the Netflix Prize dataset, the dataset has been extensively used in recommender systems research as well. More recently, there have been several more recommendation datasets released. The biggest one to date was released as part of the 2011 KDD Cup[5] containing a quarter of a billion ratings [KDK11]. This was, however, in the music domain. Music consumption represents a somewhat different consumption use case than movies do, e.g. a song lasts a few minutes whereas a movie lasts several hours. Netflix itself have announced their users' ratings to be in excess of 5 billion in total [Ama12].

[1] In June 2012, when conducting a search for the term "recommender system" on the ACM Digital Library (`http://dl.acm.org`) and limiting the publication year to before 2007 the search returned 753 documents. When changing the publication year to 2007 and onwards, the search returned 2,633 publications (see screen shots in Appendix). This is of course not only due to the Netflix Prize, but the trend is clear.

[2] `http://www.grouplens.org/`

[3] `http://www.movielens.org`

[4] The size of the largest of the Movielens datasets

[5] `http://kddcup.yahoo.com/`

2.1.1 Datasets

To provide an overview of the different datasets used in this dissertation, this section briefly introduces the most popular ones, an overview of the dimension of the datasets is given in Table 2.1. Since some of the datasets represent different methods of interaction, e.g. music, video on demand, etc., a brief explanation of the main service provided by the website is given as well.

Filmtipset This dataset is based on a database dump from the Swedish movie recommendation service Filmtipset. The dataset is not publicly available but was provided by Filmtipset for research purposes within the scope of this dissertation. Subsets of the dataset have been subsequently released in the scope of the CAMRa 2010 challenge [SBDL13; SBDL10]. The dataset contains millions of ratings provided by tens of thousands of users on tens of thousands of movies. The movie ratings are on a scale from 1 (bad) to 5 (good) in steps of 1. Filmtipset is a movie recommendation website where users enter ratings of movies they have previously seen. The ratings are subsequently used to provide personalized recommendations based on a collaborative filtering algorithm [Fil]. The service additionally contains a multitude of information about the movie, actors, directors, etc. All information, from the ratings to the description of movies, is user generated. The Filmtipset dataset provides a uniquely rich set of attributes commonly not found in other recommendation-related datasets [Sai10].

Last.fm The Last.fm dataset is the only dataset in the scope of this work not related to movies, nor does it have ratings. It is however very widely used and was included as a reference to other recommendation domains. The dataset contains users, the artists they have listened to, and the number of times they have done so. It contains roughly 360 thousand users, 186 thousand identified artists, and 107 thousand unidentified artists. The dataset was collected and released by Òscar Celma [Cel10a] in 2010. Last.fm is music discovery website that creates a musical profile of the users listening habits (whether on the website itself or uploaded from a different service). The profile is then used to create a personalized playlist that can be played through the website [McC11].

Movielens The Movielens dataset is provided by the Grouplens research group and comes in three sizes, 100 thousand ratings, 1 million ratings and 10 million ratings. In the scope of this work, only the largest of the three has been used. It contains ten million ratings on 10 thousand movies by 72 thousand users. Ratings are on a five star scale with half star increments [Mov]. The datasets is one of the most widely used movie rating datasets. Movielens provides a service similar to Filmtipset mentioned above. Users are encouraged to rate seen movies in order to receive personalized recommendations [Mov]. The single biggest difference between Movielens and the other datasets listed here is the purpose behind the website. The creators claim that the goal of the website is to collect research data on personalized recommender systems [OZ51].

Moviepilot This dataset is based on a database dump from the German movie recommendation service Moviepilot. Similarly to the Filmtipset dataset, it is not publicly available but was provided by Moviepilot for research purposes. Subsets of this dataset have been released in the scope of the CAMRa 2010 [SBDL10] and the CAMRa 2011 [SBDL11] challenges. The dataset is similar in size to Filmtipset with the exception that ratings are on a scale from 0 (bad) to 10 (good) in steps of 0.5, uniquely for this dataset is that 0 is a valid rating. Just as Filmtipset and Movielens, Moviepilot is a recommendation-centered service where users are expected to rate movies and TV-shows in order to receive personalized recommendations. The website contains some additional information regarding movies, actors, etc. and provides trailers for selected movies.

Netflix The Netflix dataset was released as part of the Netflix Challenge in 2006 [Net06]. The dataset contains in excess of 100 million ratings on nearly 18 thousand movies by 480 thousand users. The ratings are on a five star scale in 1 star increments. The Netflix dataset (or subsets of it), together with the Movielens datasets, are arguably the most used rating datasets in recommender systems research. Netflix differs from the above mentioned movie-oriented websites as it not only collects ratings from it's users, it is also a video-on-demand service, where the recommended movies and TV-shows can be watched instantly, and rated upon having been seen. What additionally sets Netflix apart from the other services is that it charges its users a monthly fee, whereas the others are free of charge. The data is however from a period of time when the primary consumption method for movies was through DVDs which were sent through the postal service. The implication of this is that Netflix's customers needed to wait a few days between selecting the movie and actually watching it.

Dataset	Users	Items	Interactions	Type
Filmtipset	81, 282	64, 747	19, 812, 490	Information
Last.fm	359, 349	160, 154	3, 778, 552, 887	Personalized radio
Movielens	71, 567	10, 681	10, 000, 054	Information
Moviepilot	160, 973	27,148	7, 321, 151	Information
Netflix	480, 189	17, 770	100, 480, 507	Video on demand

Table 2.1: Dataset statistics. Type refers to the type of service the website offers. Interactions for the movie-related datasets refer to the number of ratings whereas in the Last.fm dataset the number refers to the number of times the tracks in the service have been listened to.

2.2 Recommender Systems

Most recommender systems today are, in broad terms, either based on collaborative filtering techniques, content-based techniques, or a combination of both, so-called *hybrid recommender systems* [AT05; BHC98; Goo+99; LR07]. The selection of techniques to use for recommendation is based on the recommendation use case, whether it comes to items (movies, music), words and text (auto completion tasks in programming and messaging in mobile phones), or bidirectional recommendations of people (dating websites, job offers). Most of the currently used techniques are based on similar techniques found in information retrieval and until recently, recommender systems were commonly classified together with information retrieval systems [CR11].

Collaborative filtering-based recommendation attempts to identify similarities between entities (users or items), based on the interaction history of these entities, e.g. movies which have a large intersection in terms of users who have seen them, or users who have a large intersection in terms of items they have purchased. Content-based systems, on the other hand, as their name implies, base recommendations on the content of items, e.g. documents with a large intersection of words, lists with similar items, etc. [BYRN99; Gol+92]. Hybrid recommenders, as mentioned above, combine collaborative and content-based approaches into one unified approach.

Examples of use-cases for content-aware recommender systems are systems which do not keep track of information about their users, e.g. only recommending items which are relevant to what is currently being presented to the users. This includes news websites (presenting articles related to the one currently being read), shopping websites (presenting books by the same author, DVDs of other seasons of the same TV series currently being browsed), etc [BS97]. Collaborative filtering-based recommendations are common in systems which store information about their customers, the most prominent example (especially in the context of movie recommendation) being Netflix, where recommendations are based on what other users with similar viewing profiles watch [RV97], i.e you are probably going to like movies liked by those with similar viewing habits to yours.

The two most common use cases for recommender systems are item ranking and rating prediction [DSM10; Her+04; SBZ11]. The latter, for obvious reasons, is only applicable in scenarios where the underlying data contains ratings. The former is a more general approach which can be applied to various types of data, e.g. explicit interaction (purchases, tags, ratings, etc.), or implicit data (e.g. viewed profiles, or even mouseovers[6]) [Sar+00a]. There are additional constraints brought by the user interfaces in which the systems are used, e.g. a website recommending a physical item for purchase has a different context than a telemarketer recommending a new telephone contract to a potential customer.

The scope of this dissertation does however not extend beyond the use case of a website recommending a movie, either for direct consumption through streaming or for consumption at a later point as the analysis and empirical studies performed have been on movie-related consumption data specifically. In this setting, recommendations are most often based on explicit data and the approaches used are then suitable for this specific use case. These types of recommender systems commonly use collaborative filtering techniques which are categorized as either *memory-based* or *model-based*. The two techniques differ in the way the recommendation model is created, i.e. memory-based approaches use information about the similarities between items or users and create neighborhoods of the most similar entities, and model-based techniques on the other hand use machine learning techniques to identify latent patterns (factors) in the data which are not identifiable otherwise.

One of the most commonly used memory-based algorithms, the k-nearest neighbor algorithm finds items to recommend through first building neighborhoods of the k most similar items. In user-based collaborative filtering (as opposed to item-based collaborative filtering [Sar+01]), similarities are based on user actions (e.g. ratings) which are represented as feature vectors. Using a certain similarity metric, e.g. cosine similarity, the k most similar users create a neighborhood. Recommendations are then found by identifying the most common and favorably rated items among all users in the neighborhood that the candidate user has not interacted with. An exhaustive characterization of the algorithm and common similarity metrics is given in Chapters 3 and 4.

Model-based recommendation algorithms became popular through the Netflix

[6] A mouseover is the act of hovering over a hyperlink or any other item

Prize, partly due to the simple fact that this type of algorithms performed extremely well in the scope of the challenge, and were ultimately part of the winning ensemble algorithm [AB12]. Among the more readily used algorithms part of this family are those based on matrix factorization, e.g. *singular value decomposition* (SVD) [DK90] and *latent dirichlet allocation* (LDA) [BNJ03]. SVD is currently implemented in most of the available open source frameworks for recommendation, e.g. Mahout[7], LenskitRS[8], MyMediaLite[9], etc. The selection between model-based and memory-based methods depends on data structure, size, and other practical as well as theoretical requirements.

In the scope of this dissertation, only memory-based algorithms have been used. This is due to the general applicability of the recommendation and evaluation approaches studied. The concepts presented are however easily transferable to the model-based family of algorithms.

2.3 Evaluation of Recommender Systems

Due to the similarities to information retrieval systems such as search engines, question answering systems, etc., the evaluation of recommender systems has, to a large extent, been based on information retrieval concepts such as precision, recall, F-measure, etc. These measures represent some form of quality of the system, e.g. the higher the precision and/or recall value is, the more accurate the system is. However, even though information retrieval systems and recommender systems are similar both in their use and implementation, there is a distinct contextual difference; whereas retrieving a known (but sought for) item is positive, recommending a known item has far lower utility. In respect to this, many of the traditional metrics used in information retrieval cannot accurately represent the quality of a recommender system, at least above a certain threshold [Her+04].

2.3.1 Metrics and Methods

Recommendation qualities are commonly expressed through a number of metrics and methods. The choice of these is often based on the type of dataset used in the system, the use case, expected outcome, etc. Arguably the most common metric in recommender systems (and information retrieval) is the precision

[7] `http://mahout.apache.org`

[8] `http://lenskit.grouplens.org`

[9] `http://www.mymedialite.net`

(P) and recall (R) pair [Her+04]. These metrics are usually applied in offline train/test scenarios, where algorithms are trained using a portion of the available data and then evaluated by comparing predictions to a withheld portion of the data, i.e. *true positive* recommendation.

Precision is the fraction of relevant retrieved documents. In a recommender system evaluation setting it corresponds to the true positive fraction of recommended items. Recall is the fraction of all relevant items which are retrieved. The formula for calculating precision is shown in Eq. (2.1) while recall is shown in Eq. (2.2). In both equations, *relevant* refers to the complete set of relevant items, and *retrieved* refers to the complete set of retrieved items.

$$P = \frac{|\{relevant\} \cap \{retrieved\}|}{|\{retrieved\}|} \tag{2.1}$$

$$R = \frac{|\{relevant\} \cap \{retrieved\}|}{|\{relevant\}|} \tag{2.2}$$

Commonly, precision is expressed as precision at k where k is the length of the list of recommended items, e.g. $P@1 = 1$ would indicate that one item was recommended, and the item was deemed to be a true positive recommendation, $P@2 = 0.5$ would indicate that two items were recommended and one them was deemed a true positive, etc.

Variants of precision used for recommender evaluation include *Average Precision* (AP) and *Mean Average Precision* (MAP). Both these metrics are used when more than one item is recommended. They extend the precision metric by taking into consideration the position of true positive recommendations in a list of recommended items, i.e. the position k in a list of n recommended items in Eq. (2.3). $rel(k)$ is a binary classifier taking the value 1 if the item at position k is relevant and 0 otherwise. Mean Average Precision additionally averages the scores at each k, i.e. as shown for position q in Eq. (2.4).

$$AP = \frac{\sum_{k=1}^{n} (P(k) \times rel(k))}{|\{relevant\}|} \tag{2.3}$$

$$MAP = \frac{\sum_{q=1}^{Q} AP(q)}{Q} \tag{2.4}$$

Other common metrics, used in the context of rating prediction, are the Root-Mean-Square Error (RMSE) and normalized Discounted Cumulative Gain (nDCG).

In contrast to precision-based metrics, RMSE attempts to estimate the recommendation algorithm's rating prediction error, e.g. by comparing predicted ratings to actual ratings. The lower the error, the better the algorithm performs. RMSE is calculated as shown in Eq. (2.5), where X, Y are two rating vectors (e.g. predicted item ratings vs. actual item ratings), where each position in the vector corresponds to the rating of a specific movie and n the size of the intersection of nonzero elements in both vectors.

$$RMSE(X,Y) = \sqrt{\frac{\sum_{i=1}^{n} (x_i - y_i)^2}{n}} \tag{2.5}$$

nDCG and DCG on the other hand measures the usefulness (relevance) of a document based on its position in the list of recommended items. In a rating scenario, this corresponds to how high the predicted ratings of the top-k items are, the formula is shown in Eq. (2.6) where the *gain* (the predicted rating) of each item i for each user u in a list of J items is represented by g_{ui_j}. nDCG is the DCG over the true DCG, i.e. *ideal* DCG (IDCG) – the actual ratings, as shown in Eq. (2.7).

$$DCG = \frac{1}{N} \sum_{u=1}^{N} \sum_{j=1}^{J} \frac{g_{ui_j}}{\max(1, \log_2 j)} \tag{2.6}$$

$$nDCG = \frac{DCG}{IDCG} \tag{2.7}$$

In the context of this dissertation, precision, recall and RMSE are used in offline testing. When evaluating recommendations through user studies, more subjective qualities of the recommended items can be collected. Commonly, in online evaluation of recommender systems, *A/B testing* is conducted in order to study how subsets of users respond to different recommendation algorithms. In the most basic case of recommender system testing, A/B testing is the process where separate groups of users are unknowingly exposed to different recommendation algorithms. Their interaction data with the system, e.g. ratings, is then analyzed and compared in order to find which algorithm performed best compared to the baseline. These tests require the populations exposed to the algorithms to be of substantial size in order to generate reliable data [KWK11].

Chapter 4 shows how a correlation of different perception-oriented qualities can be attained and discusses the evaluation of a system based on both offline and online metrics.

2.3.2 The Utility of Recommendation

Traditional recommender system evaluation is based on the train/test split concept, where a portion of already collected data is used to train an algorithm, and a portion of withheld data is used for validation. In the movie recommender use case this boils down to extracting a subset of a user's ratings, using the remaining data to train the algorithm and either seeing how well the recommendation predicts the withheld ratings (in rating prediction scenarios), or (in the top-k scenario) how highly recommended the withheld, highly rated, movies are. The lower the rating prediction error, or the higher the precision, the better is the recommendation algorithm [Her+04] deemed to be. This type of evaluation offers a seemingly accurate estimation of the *actual* recommendation quality. The actual quality is the quality as perceived by the end user. The challenge with this type of (offline) evaluation is the lack of direct user feedback, i.e. a movie which is recommended will only be deemed as a positive recommendation if it has already been seen by the user. In the cases where the recommender finds movies which would have been positively rated by users in a real case, the evaluation metric will still point the recommendation out as a false positive. This misalignment with the real world becomes a problem when recommendation algorithms become more and more accurate. At some point, when using traditional metrics to estimate the ability of recommenders, these metrics can create deterring results, over-optimizing on factors which only present themselves in synthetic evaluation settings [APO09; Ama+09; Sai+12b; Sai+12f]. Similar challenges exist in non-recommendation-centric information retrieval scenarios, where approaches like relevance feedback [SB97], continuous crawling [BP98], etc. [SC12] are used to mitigate the effects.

This type of evaluation, referred to as offline evaluation in the context of this thesis, has other qualitative drawbacks as well, e.g. it is strongly affected by any biases contained in the datasets, due to which it often can create even greater biases [SJA12] lowering the utility and the *perceived quality* of recommendations.

In order to overcome said drawbacks, the evaluation of recommender systems needs to be based on more recommendation-centric concepts, instead of focusing on information retrieval methods [Cos+03; Her+04; Sai+12f]. Recommendation-centric concepts such as novelty, diversity, and serendipity cannot be evaluated through the traditional methods and *objective metrics* used in information retrieval and recommendation evaluation. There are several more suitable metrics, e.g. *unexpectedness* [AT11], *intra-list similarity* [Zie+05], *rank* and *relevance* [VC11], etc. However, similarly to precision, recall, and other common accuracy metrics, these are objective measures, not taking the actual user feedback into consideration.

There is considerable difficulty in measuring concepts such as serendipity and novelty without *subjective metrics.* Subjective metrics, as their name suggests, measure the subjective opinion of the users. The reason these metrics are better suited to measure novelty, serendipity, etc. is due to the qualities the concepts represent. Diversity is the quality of finding something unexpected – when not looking for it. Novelty is the quality of something being new or unknown. Diversity, in the context of recommendation, is the quality which represents a heterogeneous set of recommended items. When putting these values into the context of offline evaluation, it becomes clear that these concepts cannot be captured objectively in datasets which represent historical events. For instance, we cannot deem whether an item will be a surprising recommendation for a user if this item is already part of the user's profile, i.e. the user has already purchased or rated the item. The same applies to novelty.

Diversity can be measured objectively, but will not reflect the users' subjective opinions on the quality of the recommender items – these opinions will most likely be different in different contexts, and as such should be evaluated in the context of the applied use case.

The perceived quality of recommendations does not solely rely on the prediction accuracy, but also of other subjective concepts like mentioned above. In order to measure the utility of a recommendation, these factors need to be accounted for as well. In Chapter 3 we look at the effects of popularity of items on recommendation quality and evaluation. In Chapter 4 recommendations are evaluated with utility-oriented concepts in mind in order to estimate how both offline and online evaluation can be implemented in order to properly estimate the quality of a recommendation. Additionally, the chapter focuses on aspects of correlation of these subjective measures, e.g. how serendipity novelty are related to each other in the eyes of the users.

Chapter 3

Popularity Bias

In this chapter we investigate how this affects the quality of recommendations presented to users in three different phases of usage; when the users have interacted with few items, so-called cold start; when the users have established a profile by having rated a few dozen items - a post cold start phase [PT09]; and finally when the users have rated many items. Our experiments confirm that recommendation quality during the post cold start phase is negatively affected by the popularity bias found in the most commonly used recommendation datasets. With this in mind, we propose that de-biasing weighting schemes be used for users in the post cold start phase. Additionally we speculate whether traditional offline evaluation can truthfully capture the actual quality of recommendations. point in time.

The chapter is structured as follows, first an introduction to concepts related to popularity bias in given in Section 3.1, this is followed by a survey of the state-of-the-art in popularity bias-related work in recommender systems in Section 3.2. After this, an analysis of popularity bias in commonly used recommendation datasets is presented in Section 3.3, following by an overview of popularity bias mitigating techniques in Section 3.4. These mitigating techniques are then applied in Section 3.5 upon which the chapter concludes in Section 3.6.

3.1 Introduction

The inherent popularity bias in consumption patterns has as side effect that just by presenting non-personalized lists of popular items will result in a certain level of satisfaction. However, when we ask ourselves "What items do I want to get recommended?" ultimately the answer will not be "items that I know of". If a recommender system recommends items which are known to me, the end user, the *utility* of said recommender system is low [Cre+11b], e.g. there is little point in recommending an item which the user is aware of (and might actually have neglected to interact with purposely).

Some items will, for whatever reason, be more popular than others. Whether due to advertisements, temporal aspects, word of mouth, or any other external factor. This property has been shown to exist in most data collections containing user-item interactions, whether in folksonomies [WZB08], books [Lin06], music [Cel10b; Cel08], movies [SJA12]. To illustrate this, Table 3.1 shows the five most popular items in three common recommender systems datasets (Movielens, Netflix, Last.fm) as well as two (Filmtipset, Moviepilot) real-world movie recommendation websites. The first three datasets are not complete, i.e. they are either collected by third parties through Application Programming Interfaces (Last.fm [Cel10a]), intentionally reduced (Movielens [Mov]), or contain a randomized subset of the complete dataset (Netflix [Net06]). The latter two are complete, frozen in time, snapshots from the respective services, thus guaranteeing the completeness of the data.

The table shows that the five most popular items in each of the datasets have been rated by between 40% (Netflix) and 80% (Filmtipset) of all users, and additionally the percentage of how many users rated the items positively (a rating ≥ 3 on rating scales between 1 and 5 and ≥ 5.5 on rating scales between 0 and 10). In the most extreme case, Movielens, almost all the ratings are positive (between 92% and 100%). Again, in a setting where similarities between users are used to generate recommendations, this can lead to (pseudo-personalized) recommendations, i.e. based more on popularity than on actual similarities in taste.

Given this popularity bias, the question that arises is whether items such as the movies "Pulp Fiction" or "Forrest Gump" (in the Movielens case) should be considered by collaborative filtering recommender systems when trying to identify similarities between users? Or whether these very popular movies *actually* imply a similarity in taste between any two arbitrarily selected users? An alternative approach is to take the popularity of items into consideration when calculating the similarities of two users (in user-based collaborative filtering approaches), this approach is investigated in the scope of this chapter.

Dataset: Movielens

Movie	users	positive ratings %
Pulp Fiction	48.7%	100%
Forrest Gump	48.1%	93.7%
The Silence of the Lambs	47.0%	96.9%
Jurassic Park	45.6%	92.0%
The Shawshank Redemption	43.5%	98.6%
Dataset: Netflix Prize		
Miss Congeniality	48.5%	79.7%
Independence Day	45.1%	88.1%
The Patriot	41.8%	87.4%
The Day After Tomorrow	40.9%	81.7%
Sam & Janet	40.4%	95.0%
Dataset: Last.fm (ratings are not available in Last.fm)		
The Beatles	21.5%	N/A
Radiohead	21.3%	N/A
Coldplay	18.6%	N/A
Pink Floyd	13.6%	N/A
Metallica	13.1%	N/A
Dataset: Filmtipset		
71770415	80.9%	96.0%
4277361	67.9%	93.5%
13285617	66.5%	66.3%
20412823	65.8%	38.4%
32580825	65.7%	44.3%
Dataset: Moviepilot		
Titanic	26.0%	58.6%
The Fellowship of The Ring	24.8%	83.0%
The Matrix	24.6%	83.9%
Der Schuh des Manitu	23.4%	66.7%
The Return of The King	21.6%	84.9%

Table 3.1: The 5 most popular items, the percentage of users who have rated or listened to them, and the percentage of ratings which were positive, i.e. $r \geq 3$ (for Movielens, Netflix and Filmtipset) or $r \geq 5.5$ for Moviepilot, in five recommendation datasets. As Last.fm does not have ratings, the column has been omitted.

The problems caused by this type of popularity bias, the ever increasing popularity of a few items, commonly known as *the long tail*, have been the focus of many recommender systems-related publications [And06; Her+04; PT08].

The work related to popularity and the long tail conducted in the scope of this dissertation focuses on *weighting schemes* for collaborative filtering-based recommender systems using *neighborhood*-based approaches.

3.2 Related Work

Popularity bias, commonly known as *the long tail*, a term popularized by Chris Anderson in 2004 [And04], is a well-known and widely-researched concept in recommender systems and information retrieval, e.g. [Goe+10; LL11; PT08; Yin+12]. Formally, the name refers to the graphical representation of a distribution where a large portion of the distribution is in the "tail", when compared to a Gaussian distribution, e.g. Fig. 3.1.

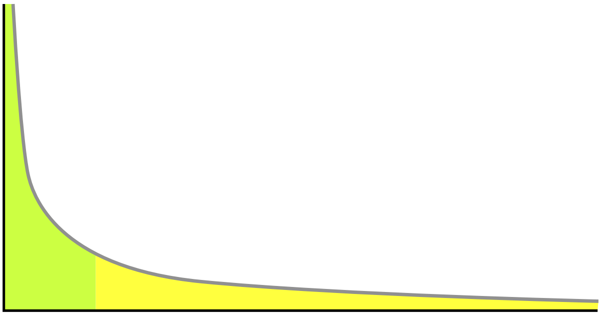


Figure 3.1: The *long tail* refers to the descending popularity distribution of the population, i.e. the yellow part of the curve.

Among the earliest works to mention the long tail, in the context of recommendation, described the Knowledge Pump system [GAD98] for sharing and recommending URLs in a workplace. The authors noted that, even in a small group of 13 users, there were popularity biases, and that e.g. "…a minority of the users do a majority of the work" [Gla+01].

Lam and Riedl [LR04] look at the problem from a somewhat different perspective, trying to identify what effect "unscrupulous producers" can cause in recommender systems when falsely recommending items. They conclude that items with low popularity are easier to manipulate in order to boost their popularity, in effect making them popular (which in turn has an effect on the evaluation of a recommender system similar to the one discussed in this chapter). A similar concept, whether recommender systems reduce diversity and cause some items to be popular, is explored by Fleder and Hosanagar [FH07]. In their work, the authors compare two types of recommendation systems (real and simulated) and found that both were biased by popular items, which created a "concentration bias" in the resulting datasets.

In terms of evaluation and popularity bias, Celma and Herrera [CH08] state that "Indeed, two out of five tested music recommenders contain John Lennon, Paul McCartney and George Harrison in their top-10 (Last.fm and the.echotron.com). Yahoo! Music recommends John Lennon and Paul McCartney (1st and 4th position, respectively) . . . "[1], the list continues listing other music recommendation systems that have similar recommendations. Furthermore, they analyze a dataset from Last.fm to find out what effects this popularity bias has on novelty in recommender systems. They compare the results of a purely content-based recommender system to that of a collaborative filtering-based recommender systems both in the offline case as well as through a user study. Their findings include an observation that the perceived quality of the collaborative recommender was higher than that of the concent-based one even though the content-based recommender was less prone to recommending popular items (not as sensitive to the effect of popularity as CF). Celma and Lamere presented similar concepts of music recommendation in the long tail in [CL08; CL11].

Another approach to recommending items from the longs tail was presented by Park and Tuzhilin [PT08]. In their approach, the authors split the complete set of items into two subsets, one from the *short head* (the green part of Fig. 3.1), and one from the long tail. Following this, recommendations are computed from each set. This approach creates recommendation results, with sustained, or even lower prediction errors when compared to a more traditional recommendation setting, showing the benefit of taking less popular items into consideration.

Several works that propose recommender systems should be evaluated differently in order to allow for less influence caused by popularity bias include Zhang [Zha09], where the authors suggests *concentration* as an evaluation metric. Jambor and Wang [JW10] instead propose a framework for optimizing recommendation accuracy on multiple objectives including a value which describes to which extent users are interested in popular items vs. the extent they are interested in items from the long tail. Cremonesi et al. [CKT10] argue that when evaluating top-n recommendations, the test sets need to be chosen carefully, otherwise a small number of very popular items can affect the predicted accuracy negatively. Steck [Ste11] presents a new evaluation measure for serendipitous items from the long tail motivated by the the fact that popularity is not only applicable

[1] `http://music.yahoo.com`

to the test metrics, but also the objective function of the training process of recommendation algorithms. The author additionally mentions that popularity bias also affects the amount of feedback items get, i.e. popular items get more feedback whereas less popular get less of it, not necessarily in proportion to each user's true interest. The proposed measure, *popularity-stratified recall*, weights the recall measure with a factor inversely proportional to the probability of an item being rated as relevant.

Koenigstein et al. [KDK11] approached the popularity bias in the Yahoo! Music dataset by using a known taxonomy of items in order to alleviate the low number of interactions for the vast majority of items. This approach was possible due to the music taxonomy in the dataset, i.e. genres containing artists, artists having produced albums, and tracks belonging to albums. Also in the music domain, Levy and Bosteels [LB11] instead used a weighting factor to minimize the effect of popular artists in a collaborative filtering-based recommender based on the weighted sum method [Sar+01].

Celma and Cano [CC08] approach the long tail problem from a network analysis perspective based on the average shortest path, the degree distribution and degree correlation in the graph expressing the user-item interactions in a dataset from Last.fm. The dataset is heavily biased (14% of artists account for 86% of playcounts). Their analysis shows that popularity is reinforced at the expense of less popular artists, creating a scenario where artists from the long tail become less and less accessible to the users.

Item re-ranking, or item weighting, is applied by Adomavicius and Kwon [AK09] successfully on the Movielens and Netflix Prize datasets to improve diversity, while still retaining comparable levels of prediction accuracy. The authors compare five different ranking functions (popularity, reverse rating value, average rating, absolute likeability and relative likeability) and show that with as little as 1% loss in precision, diversity (i.e. recommendations from the long tail) can be increased two-fold (from 16% to 32%) – and more than three-fold with as little as 5% decrease in accuracy.

The underlying assumption of feature weighting is that some items may provide more predictive information than others. In particular those items which are most popular are considered as less informative. To address this issue Breese et al. [BHK98] used the inverse user frequency as weights to devalue the contribution of popular items in the Pearson correlation coefficient. Herlocker et al.[Her+99] modified the Pearson correlation coefficient by incorporating an item-variance factor. In doing so, items with low variance, such as commonly liked items, are devalued.

Yu et al. [Yu+03] proposed an information-theoretic item weighting approach based based on mutual information. Jin et al. [JCS04], instead learn the item weights from the ratings by maximizing the average similarity between users. In addition, the weighting scheme was empirically validated not only for the Pearson correlation coefficient but also for other configurations of memory- and model-based approaches. Due to contradicting empirical results, Baltrunas [BR07] conducted an empirical comparison of different weighting schemes. In their work, the authors suggested a feature weighting based on a singular value decomposition in conjunction with the Pearson correlation.

Luo et al. [Luo+08] use feature weighting as one of several ingredients to model the relationship of users using local and global user similarity. By assuming that the ratings obey a Laplacian distribution, they construct so-called surprisal-based vector similarities. The term surprisal refers to quantities of information contained in the ratings in order to express the relationship between any two users.

Symeonidis et al. [SNM07] compare how weights on different features affect the user profile in terms of recommendation accuracy of collaborative filtering, content-based and hybrid recommendation approaches.

All of the above mentioned approaches do not differentiate between users in terms of their logged activity, i.e. number of ratings as in our case. In this work, we distinguish between different users based on their system usage, i.e. the number of movies they have rated. We identify three classes of users, *cold start* users, *post cold start* users and finally *power* users.

3.3 Analysis of Popularity Bias in Recommendation Dataset

The problem, that popular items tend to get higher importance in similarity calculations, is directly related to how users interact with items. The following sections describe the characteristics of three datasets and describe how similarity metrics could be altered in order to compensate for popularity-induced bias in different stages of users' system usage.

The datasets analyzed are; the Movielens 10M100K dataset[2] which contains 10 million movie ratings by 70 thousand users [Mov], the Moviepilot dataset[3] which contains 4.5 million ratings by 105 thousand users [SBDL10], and the Last.fm

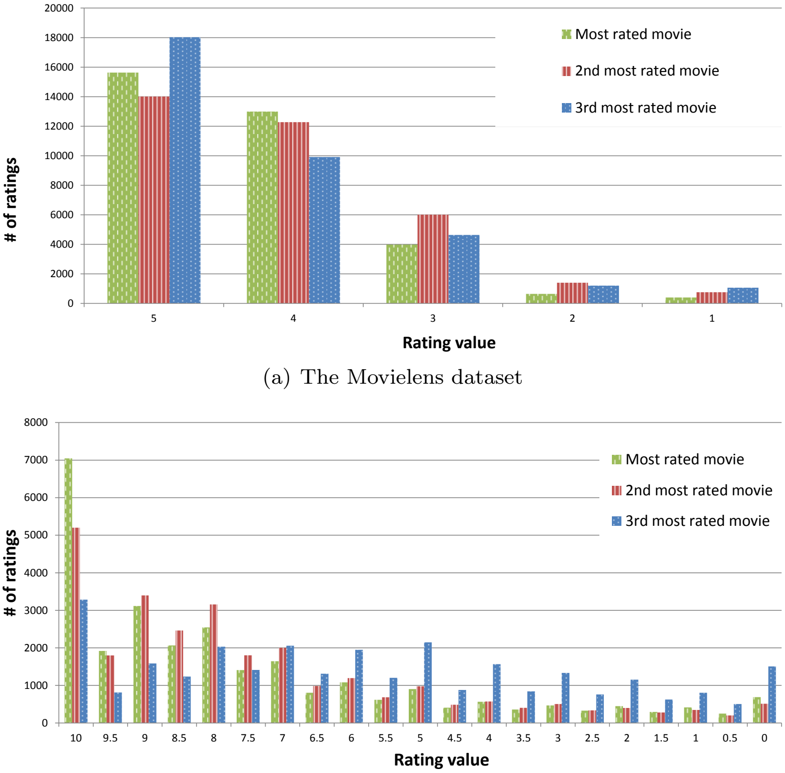
[2] `http://www.grouplens.org/node/470`
[3] `http://www.dai-labor.de/camra2010/datasets/`

360K dataset [Cel10a] which contains the user-artist pairs and the number of times the user has listened to the artist. The dataset contains roughly 360 thousand users and almost 300 thousand artists. The Last.fm dataset was included in the analysis to show that the popularity effects presented throughout this chapter are not isolated to the movie domain. The full datasets were used for analysis presented in this section, and smaller subsets for the experiments presented in Section 3.5.

The datasets were chosen due to their popularity, e.g. see Section 2.1.1, and because the datasets use different rating schemes. This allows for a more thorough analysis of when and how different similarity measures and weighting schemes are appropriate for improving recommendation accuracy. As mentioned previously, in Movielens, users can rate movies on a {1, 2, 3, 4, 5} star scale, and in Moviepilot the ratings are made on a {0, 0.5, 1.0, . . . , 9.0, 9.5, 10.0} scale making them somewhat more fine grained. As mentioned above, the Last.fm dataset does not contain ratings, instead it has the artist play counts per user.

When looking at co-occurring movies, those movies which are popular (even if rated differently by different users) will add to the similarity due to the simple fact that two users will have rated the same popular movies. When looking at the Movielens dataset, as shown in Table 3.1, roughly 50% of the users have rated the few most popular movies. Combining this with the rating distributions for the three most popular movies shown in Fig. 3.2(a) we see that the vast majority of the ratings given to popular movies are high, i.e. 4 or 5 stars. The implication of this is that, even if two users have rated very disjoint movies, they will have a circa 50% chance to have rated at least one movie in common. Additionally, since almost half of the ratings on each of the three most popular movies have the highest rating (i.e. 5), a potentially large number of users of users (up to circa 25%) have rated at least one of the three most popular movies identically. A similar trend is visible in the Moviepilot dataset, however, due to the different rating scale (more fine grained), the effect is not as discernible. Nevertheless, as shown in Fig. 3.2(b), it is distinguishable that most of the ratings on the most popular movies are in the upper end of the rating scale. The trend exists in the the Last.fm dataset as well.

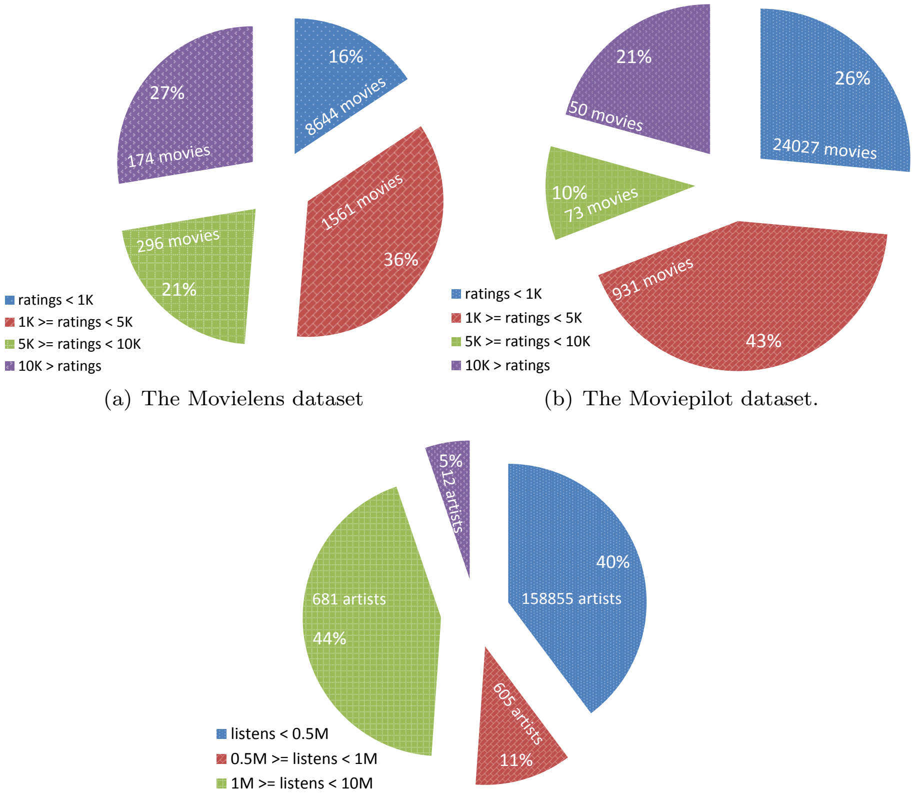
Looking at the percentage of ratings performed on movies grouped by their popularity, as shown in Fig. 3.3, it is visible that there is a clear dominance of (relatively) few items in terms of how many times they have been rated. In Movielens (Fig. 3.3(a)) 174 movies, each with more than 10, 000 ratings, correspond to 27% of all ratings performed. If additionally including movies with more than 5, 000 ratings each, the number of movies rises to 470 (out of a total of 10, 000), the rating percentage grows to 48%, i.e. slightly less than 5% of the total number of movies correspond to almost 50% of the total number of ratings. Similar numbers are shown for Moviepilot in Fig. 3.3(b) and



(a) The Movielens dataset

(b) The Moviepilot dataset.

Figure 3.2: The rating distributions of the two most popular movies in the Movielens (Fig. 3.2(a)) and Moviepilot (Fig. 3.2(b)) datasets. The Last.fm dataset is excluded as it does not contain ratings (refer to Table 3.1 for a count of plays per track). Note that Moviepilot uses a rating scale of 0 to 10 stars in steps of 0.5 (21 steps) whereas Movielens uses a 1 to 5 star scale. Due to the perception-related differences in different scales [SS01] the scales have not been normalized for comparison.



(a) The Movielens dataset

(b) The Moviepilot dataset.

(c) The Last.fm dataset.

Figure 3.3: Proportion of ratings (for movies) on number of movies and listens (on artists) on number of artists in the Movielens, Moviepilot and Last.fm datasets. The sizes of the pie slices indicate the percentage of ratings/listens performed on the most popular movies/artists. The numbers in the slices correspond to the number of movies/artists the ratings/listens in each slice are made on. Note that all three datasets have a relatively small number of very dominant items (the purple and green slices).

Last.fm, where 0.4% of the artists account for 49% of the play counts, as shown in Fig. 3.3(c).

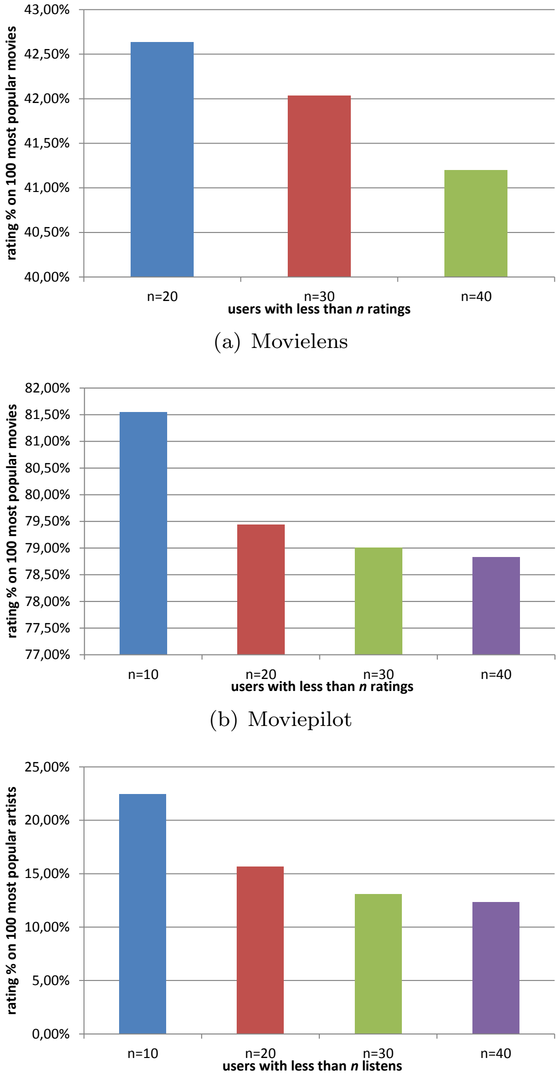
In addition, when looking at users with few ratings, i.e. the so-called cold start problem, the effect of popular items becomes even more apparent. Fig. 3.4 shows the percentage of ratings on the 100 most popular movies by users with less than 10, 20, 30 and 40 ratings respectively. The density of ratings on popular items is especially discernible in the Moviepilot dataset, Fig. 3.4(b), where users having fewer than 10 ratings have rated almost exclusively in the set of the 100 most popular movies, up to 81.55% of their ratings are given to these movies. The inclinations in the Movielens and Last.fm datasets, Fig. 3.4(a) and Fig. 3.4(c) are similar, although lower. The power law plots of the three datasets are available in Fig. 3 in the Appendix.

When using collaborative filtering in data sets with this type of heavy popularity bias, the resulting recommendations can themselves become heavily biased towards popular items. If the utility of a recommender system is to recommend items not known to the user, then recommending popular items can lessen the satisfaction of the users [AT11; And04; And06] and in the end discourage users from using it [BHS10; CVW11; CL08].

3.4 Weighting Schemes for Collaborative Filtering

Collaborative Filtering calculates the relevance of an item for a user based on other users' rating information on items co-rated by the group and the user. As mentioned in Section 2.2, Collaborative Filtering approaches are commonly categorized as either model-based or memory-based [BHK98]. In this chapter the focus is put on the latter, which creates item predictions for a user by finding users similar to the candidate user (in terms of co-rated items) in a training model, a neighborhood. Then, using information from the neighborhood, it predicts items which should be of interest to, and have not previously been rated by the user. Memory-based, or neighborhood-based, approaches commonly use measures such as the *Pearson correlation coefficient* or *cosine similarity* to create the neighborhoods.

Model-based approaches treat CF as a classification problem and provide item recommendations by initially creating a probabilistic model of user ratings using algorithms such as Bayesian networks, clustering, etc. where users are grouped into the same classes based on their rating history. Following this, the conditional probability of a user being in a rating class for a specific item is calculated.



(a) Movielens

(b) Moviepilot

(c) Last.fm

Figure 3.4: Percentages of ratings on the 100 most popular movies (Fig. 3.4(a) & Fig. 3.4(b)) and artists (Fig. 3.4(c)) by users with fewer than n ratings. In the Movielens dataset there are no users with $n < 20$ ratings, thus the corresponding percentage for the dataset cannot be provided.

The cosine similarity, Euclidean distance, and Pearson correlation coefficient have been thoroughly investigated and belong to the most common (dis)similarity measures applied in neighborhood-based CF [Her+04; Kwo08]. In the light of the discussion in Section 3.3, one problem of these similarity measures is that popular items dominate the similarity value between two users although they are potentially less informative. For example consider a worst case scenario, with two users that have rated a few items only. In this case it is likely that they have co-rated some popular items highly in accordance to our findings in Section 3.3. Additionally, it is less likely that they have co-rated *regular* items (i.e. not from the set of the most popular items), or that they agree on co-rated regular items. In this case, both users are considered as highly similar, because the similarity value is dominated by the co-ratings on the popular items. As a consequence, the recommender suggests further popular items to both users increasing their similarity even more.

Weighting Schemes In order to mitigate popularity-based effects on recommender systems, weighting schemes can be applied directly on the dataset in the recommendation process. In this context, a weighting scheme assigns a value to an item based on its popularity. In neighborhood-based collaborative filtering recommendation methods (introduced in Section 2.2) this means that the similarities caused by popular items are multiplied by some factor before the actual similarity of two users is calculated [SJA12].

One of the simplest weighting concepts is to set the multiplicative to a *linear user inverse* (LUI). The linear user inverse is inversely proportional to the item's popularity, i.e. $1 - \frac{1}{|interactions|}$. However, if the item is extremely popular, this can have detrimental effects, practically excluding the item from similarity calculations. A less obtrusive means of dealing with popularity bias is to use *inverse user frequency* (IUF) [BHK98]. Inverse user frequency is based on the *inverse document frequency* (IDF) concept introduced by Salton and McGill [SM86]. IDF, a common technique in text mining and natural language processing, divides the total number of documents by the number of documents containing a specific term. Taking the logarithm of the resulting quotient then produces the inverse document frequency, as shown in Eq. (3.1)

$$IDF(t, D) = \log \frac{|D|}{|\{d \in D : t \in d\}|}, \tag{3.1}$$

where $|D|$ is the total number of documents and $|\{d \in D : t \in d\}|$ the number of documents which contain the term t. Applying this to a user-item interaction dataset, we obtain

$$IUF(i, U) = \log \frac{|U|}{|\{u \in U : u \Leftrightarrow i\}|}, \tag{3.2}$$

where i is an item, U the set of users, and $|\{u \in U : u \Leftrightarrow i\}|$ the number of users which have interacted with item i (the operator $\Leftrightarrow$ is used for the relation "interacted with"). For very popular items, the weight becomes a decaying factor based on the items' popularity, whereas less popular items get increased weights instead.

In our experiments, described in Section 3.5, we use the weighted versions of the cosine similarity, the Euclidean distance, and the Pearson correlation coefficient. To simplify technicalities, some notations are introduced. The set of users is denoted by $\mathcal{U}$, the set of items by $\mathcal{I}$, and the set of ratings by $\mathcal{R}$ respectively. The elements of $\mathcal{R}$ are ratings r_{ui} made by a user $u \in \mathcal{U}$ for an item $i \in \mathcal{I}$. Ratings may take values from some discrete set $\mathcal{S} \subseteq \mathbb{R}$. $\mathcal{I}_u$ is used for the subset of all items that have been rated by user u and $\mathcal{I}_{uv} = \mathcal{I}_u \cap \mathcal{I}_v$ for the subset of all items that have been co-rated by users u and v.

All ratings of a user u are summarized to an extended rating vector $\boldsymbol{x}_u = (x_{ui})_{i \in \mathcal{I}}$ of dimension $|\mathcal{I}|$ with elements

$$x_{ui} = \begin{cases} r_{ui} & i \in \mathcal{I}_u \\ \varepsilon & i \notin \mathcal{I}_u \end{cases}$$

where $\varepsilon \in \mathbb{R} \setminus \mathcal{S}$ is a pre-specified value outside $\mathcal{S}$ denoting the null- or void-rating. $\varepsilon + x = x + \varepsilon = 0$ and $\varepsilon \cdot x = x \cdot \varepsilon = 0$ are defined for all $x \in \mathbb{R}$.

Suppose that $\boldsymbol{w} \in \mathcal{R}^{|\mathcal{I}|}$ is a weight vector associating a weight w_i to each item $i \in \mathcal{I}$. To formulate the weighted (dis)similarity measures, the following shortcut notations is used

$$\langle \boldsymbol{x}_u, \boldsymbol{x}_v \rangle_{\boldsymbol{w}} = \sum_{i \in \mathcal{I}_{uv}} w_i \cdot r_{ui} \cdot r_{vi} \tag{3.3}$$

$$\|\boldsymbol{x}_u\|_{\boldsymbol{w}} = \sqrt{\langle \boldsymbol{x}_u, \boldsymbol{x}_u \rangle_{\boldsymbol{w}}} = \sqrt{\sum_{i \in \mathcal{I}_u} w_i \cdot r_{ui}^2} \tag{3.4}$$

Since $\varepsilon \cdot x = 0$, only co-occurring ratings contribute to the weighted inner product of two extended rating vectors.

The weighted cosine similarity (cos), the weighted Euclidean distance (euc), and the weighted Pearson correlation coefficient (pcc), respectively, between two users u and v is defined by

$$\cos_{\boldsymbol{w}}(u,v) = \frac{\langle \boldsymbol{x}_u, \boldsymbol{x}_v \rangle_{\boldsymbol{w}}}{\|\boldsymbol{x}_u\|_{\boldsymbol{w}} \cdot \|\boldsymbol{x}_v\|_{\boldsymbol{w}}} \tag{3.5}$$

$$\mathrm{euc}_{\boldsymbol{w}}(u,v) = \|\boldsymbol{x}_u - \boldsymbol{x}_v\|_{\boldsymbol{w}} \tag{3.6}$$

$$\mathrm{pcc}_{\boldsymbol{w}}(u,v) = \frac{\langle \boldsymbol{x}_u - \bar{\boldsymbol{r}}_u, \boldsymbol{x}_v - \bar{\boldsymbol{r}}_v \rangle_{\boldsymbol{w}}}{\|\boldsymbol{x}_u - \bar{\boldsymbol{r}}_u\|_{\boldsymbol{w}} \cdot \|\boldsymbol{x}_v - \bar{\boldsymbol{r}}_v\|_{\boldsymbol{w}}}, \tag{3.7}$$

where $\bar{\boldsymbol{r}}_u = (\bar{r}_u, \cdots, \bar{r}_u)$ denotes the $|\mathcal{I}|$-dimensional vector with all elements being the average rating $\bar{r}_u$ of user u, that is

$$\bar{r}_u = \frac{1}{|\mathcal{I}_u|} \sum_{i \in \mathcal{I}_u} r_{ui}. \tag{3.8}$$

The vector consisting of $|\mathcal{I}|$ components each of which has value $\bar{r}_u$ is denoted by $\boldsymbol{r}_u = (\bar{r}_u, \cdots, \bar{r}_u)$. Choosing $w_i = 1$ for all items $i \in \mathcal{I}$, the unweighted standard dis(similarity) measures is recovered.

In the context of Eq. (3.6) to Eq. (3.7), the IUF and LUI weights can be expressed as

$$w_i^{\mathrm{IUF}} = \ln \frac{|\mathcal{U}|}{|\mathcal{U}_i|} \tag{3.9}$$

$$w_i^{\mathrm{LUI}} = 1 - \frac{|\mathcal{U}_i|}{|\mathcal{R}|} \tag{3.10}$$

for all items $i \in \mathcal{I}$.

Using these weighting schemes, our approach attempts to identify when during a user's usage of a system weighting schemes could be employed in order to heighten the recommendation experience. Our assumption is that popular items help the user to start appreciating a system, but in the long run could be detrimental to the overall utility of a system.

3.5 Experiments

The experiments investigate how the proposed weighting schemes affect the prediction accuracy for different types of users (cold start, post cold start, power users), i.e. when a weighting scheme should be employed for best results.

The experiments were performed on the Movielens and Moviepilot datasets described in Section 3.3.

3.5.1 Experimental Setup

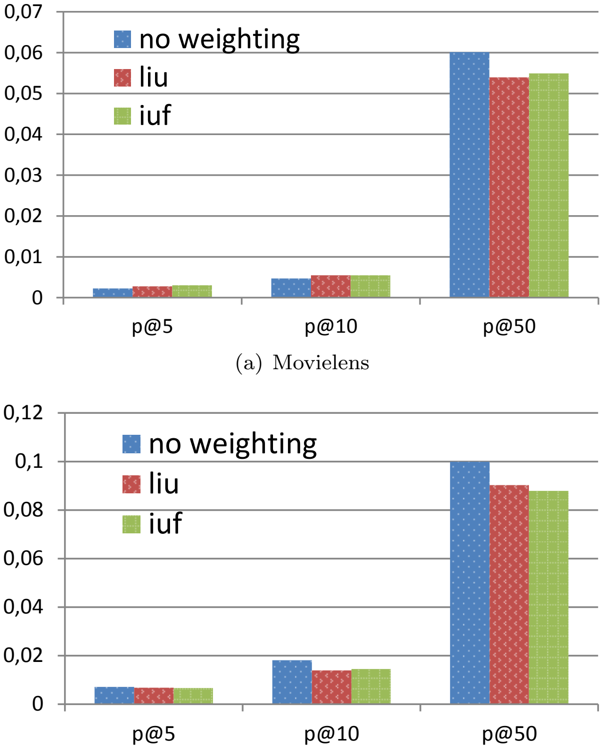
Two types of experimental setups were used, one where the weighting schemes were not used, and one where they were. Each experimental setup was performed on the similarity measures presented in Section 3.4.

For each of the datasets, several training and validation runs were performed. Positive ratings, i.e. true positives, are defined as values higher than each user's rating average + 0.5 of the user's standard rating deviation. These were removed from the training model and extracted to a validation set. The validation sets were used to calculate precision values.

In the first type of experiment, three exhaustive recommendations were conducted on the full datasets. For all users falling within the criteria (i.e. those having true positive ratings), $P@\frac{N}{2}$ where N is the number of ratings was calculated, i.e. (i) recommendations for all users with at least 10 ratings - evaluated with the Precision@5 (P@5) metric, (ii) recommendations for all users with at least 20 ratings - evaluated with Precision@10 (P@10), and finally (iii) recommendations for all users who had rated at least 100 items - evaluated with Precision@50 (P@50). $P@\frac{N}{2}$ was used in order to synthesize a more realistic scenario, i.e. calculating $P@50$ for a user with only 5 items in the evaluation set would be of little use as the user would simply have too few items.

For the cold start, post cold start and power user cases, subsets of the Movielens and Moviepilot datasets were created. Each of the subsets contained 10% of the ratings from the full datasets, selected randomly. Similarly to the experiments on the full dataset, all users in the subset falling within our criteria (those having true positive values) were evaluated. Additionally, in order to be able to identify whether an improvement of recommendation quality during the cold start, post cold start or later in a user's time line was possible, users were divided into groups. Each group consisted of users who had rated a similar amount of movies. The first group contained users with 1 to 4 ratings, the second with 5 to 10 ratings, the third with 10 to 20 ratings, etc. up to users with 140 to 150 ratings.

All experiments were conducted on both datasets, using all three similarity measures and the unweighted as well as both weighted approaches.



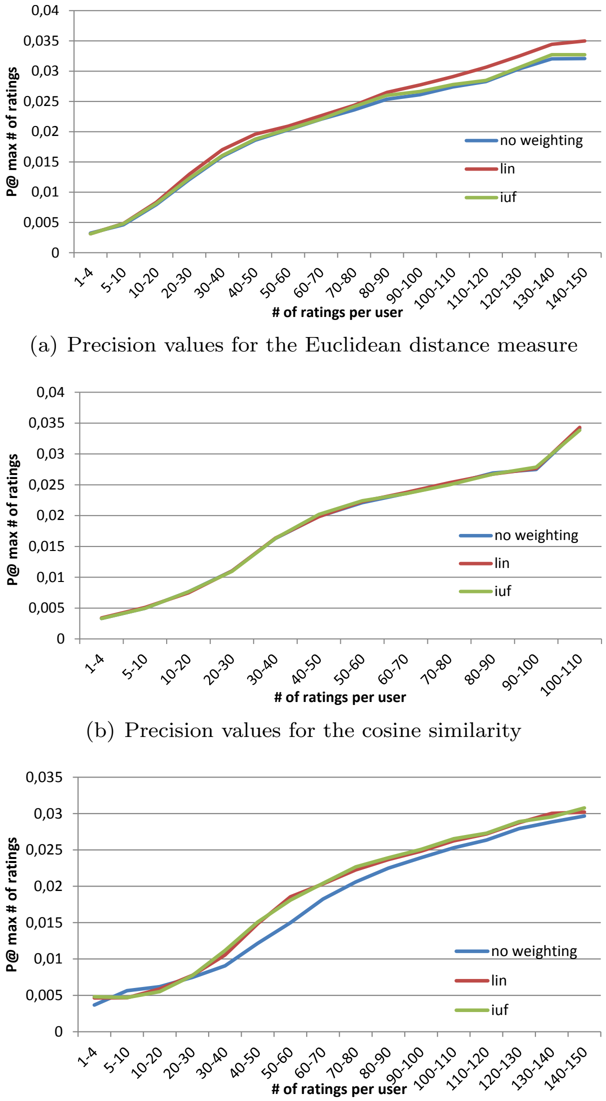
(a) Movielens

(b) Moviepilot

Figure 3.5: The precision at 5, 10 and 50 obtained with the unweighted and both weighted approaches on the full Movielens and Moviepilot datasets.

3.5.2 Results

The results of the first set of experiments, not using weighting schemes, are shown in Fig. 3.5. The Precision@N values show that when using the weighting functions, the resulting Precision@N is slightly higher for low values of N than for the unweighted approach for the Moviepilot dataset ($N = 5$). For the Moviepilot dataset, the unweighted approach seems to have the upper hand. However, as N increases, the improvement decreases and at a relatively large n ($N = 50$) the weighted approaches perform worse than the non weighted one. In the Movielens case, the unweighted approach always outperforms the weighted ones, irrelevant of N's value. This seems to be in agreement with the findings by Herlocker et al. [Her+99]. Results for the Euclidean and cosine measures show very similar trends.

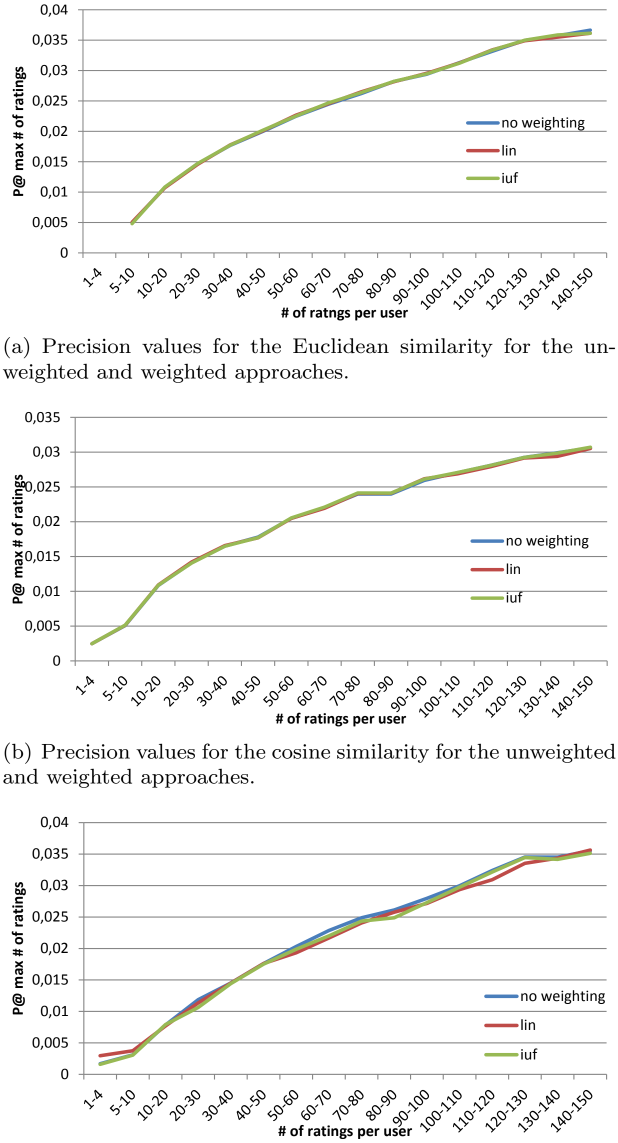


(a) Precision values for the Euclidean distance measure

(b) Precision values for the cosine similarity

(c) Precision values for the Pearson correlation

Figure 3.6: The P@ maximum number of ratings per user in each group for the unweighted and weighted approaches for the Movielens dataset and all three similarity measures.



(a) Precision values for the Euclidean similarity for the unweighted and weighted approaches.

(b) Precision values for the cosine similarity for the unweighted and weighted approaches.

(c) Precision values for the Pearson correlation for the unweighted and weighted approaches.

Figure 3.7: The P@ maximum number of ratings per user in each group for the unweighted and weighted approaches for the Moviepilot dataset and all three similarity measures.

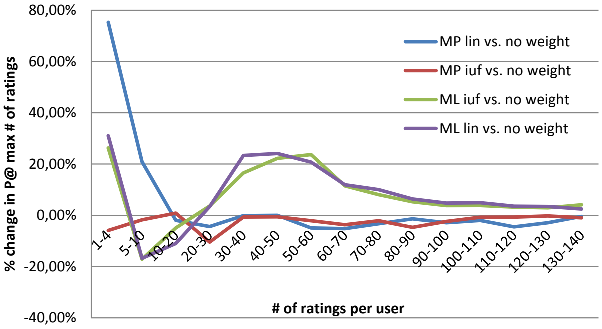


Figure 3.8: The changes of precision values for the weighted and unweighted approaches using the Pearson correlation for both datasets.

The results of the second set of experiments shown in Fig. 3.6 and Fig. 3.7 show a slight improvement when using the linear weighting approach (∼ 1%) in the case of the Euclidean distance measure for the Movielens dataset. This appears for users with 40 to 60 ratings and then again for users with more than 80 ratings. In the case of the cosine measure, all three approaches perform almost identically across the set set of precision values and users. Fig. 3.6(c) shows result of the weighted Pearson correlation approach, in this case, the figure shows a distinction between the weighted and unweighted approaches, especially in the case of users who have between 20 and 100 items (what is called the post cold start phase in this context). In this span, the weighted approaches outperform their unweighted counterpart by (at best) more than 20%.

For the Moviepilot dataset, shown in Fig. 3.7(c), results differ from those obtained with Movielens dataset. The weighting approaches seem to have little to no effect on the recommendations, resulting in a similar curve for all three models. The weighting approaches seem even seem to have detrimental effects on the Pearson correlation in this setting.

The precision changes on all datasets and weighting approaches for the Pearson correlation measure are summarized in Fig. 3.8.

3.6 Conclusion and Future Work

This chapter has studied the effects of two weighting approaches on three common similarity measures using two different movie recommendation datasets.

The motivation for the weighting schemes was given by analyzing common dataset used for recommender systems research, i.e. Last.fm, Moviepilot, Movielens. All of the datasets exhibit large popularity biases.

The test tests performed in the scope of the chapter attempted to identify whether weighting schemes can be applied in different phases of a user's usage of a recommendation website, e.g in order to identify whether or not weighting schemes can be beneficial for the purpose of overcoming problems related to cold start as well as profiling users in order to generate more accurate profiles not based on the most popular items.

Observations that the weighting schemes seem to have little effect on datasets with a wide rating scale and high concentration of ratings on popular items were made. For instance, in the case of the Moviepilot dataset, which has a rating scale stretching from 0 to 10 in steps of 0.5, and where 80% (refer to Fig. 3.3 and Fig. 3.4) of all ratings by users with few ratings are performed on movies from the 100 most popular movies.

Furthermore, the experiments imply that especially the cosine similarity measure is very insignificantly affected by any weighting measure and produces results identical no matter if weighting is applied or not. These observations hold irrelevant of the profile of the users (i.e. no matter whether the user is a new user with few ratings or a power user with many ratings).

However, there seems to be relatively much to gain in terms of precision during the post cold start phase for datasets similar to the Movielens dataset when using the Pearson correlation measure. At best, the improvement of precision values in this case is above 20%, as shown in Fig. 3.8.

There is however one aspect that is not evaluated in the context of this chapter, the quality of the recommendations as experienced by the users. Providing recommendations with higher diversity should prospectively be of a higher utility for the user, this type of evaluation can, however, not capture subjective qualities such as diversity.

Another of the drawbacks of these experiments is that due to the inherent popularity bias, items which could be regarded as good recommendations by the end users are not regarded as such in the evaluation, e.g. users will have not seen unpopular items, thus the evaluation favors known items with potentially low utility. These are aspects which Chapter 4 focuses on instead. In addition to the offline evaluation conducted in the scope of this chapter, the recommendations are evaluated through direct interaction with users in order to find whether offline accuracy measures correspond to the quality of the recommendations as perceived by the users.

Chapter 4

Evaluating Perception

This chapter covers the concept of *perceived quality* of recommender systems and issues related to it, e.g. *user satisfaction*, *correlation of offline accuracy measures to users' perceived quality* of a system.

First, in Section 4.1 an overview of evaluation concepts is given, i.e. user-centric evaluation and the correlation of perceived qualities. This is followed by an overview of the related work in the topic in Section 4.2, and neighborhood models in Section 4.3. The neighborhood models section is relevant due to the evaluation of the perception-related qualities explored in the this chapter, i.e. in order to evaluate two clearly differing recommender algorithms, an inverted version of the kNN algorithm was created, the *k furthest neighbors* (kFN) algorithm. kFN recommends more diverse items, which is one of the concepts evaluated in Section 4.4.

4.1 Introduction

User-centric evaluation, as opposed to traditional *information*- or *data*-centric evaluation is performed through direct involvement of a system's users in order to establish a *qualitative* estimate of a system's quality as perceived by the end users themselves.

	u_1	u_2	u_3	u_4	u_5
i_1	1	1	0	0	1
i_2	1	0	1	1	1
i_3	0	0	0	1	0
i_4	1	0	1	0	1
i_5	0	0	1	1	0
i_6	0	0	0	1	0

Table 4.1: A user-item matrix divided into a training set (above the dashed line) and a test set (below the dashed line).

The traditional evaluation approach to recommender systems is based on offline tests where a portion of existing data is withheld from the recommendation algorithm during the training phase [Her+04]. This withheld data, the so-called test set, is then used to measure the predictive accuracy of the algorithm, either in a *top-n*, or *rating prediction* scenario.

To illustrate one of the challenges related to offline evaluation in the context of user-centric evaluation, consider this top-n recommendation scenario: We have a user-item interaction matrix, as shown in Table 4.1. The table shows a matrix of 5 users and 6 items and their interactions, e.g. a 1 represents an interaction (rating, purchase, etc.), a 0 the lack of such. The training/test split is illustrated by the the dashed line. In this case, an offline evaluation will only recognize item i_5 as a true positive recommendation for user u_3 and items i_5 and i_6 for user u_4. Users u_1, u_2 and u_5 will not have any true positive recommendations since they have not interacted with any of the items. The evaluation does not consider that the items might actually be liked by the user, if recommended in a real-world situation. Similarly, the fact that the u_3 has interacted with i_5 does not need to imply that the item is a good recommendation.

Offline evaluation estimates the users' taste by analyzing their histories of item interactions, e.g. items they have *rated*, *purchased*, or *consumed* in any other way. However, as discussed in Chapter 3, popularity bias, as well as aspects such as *diversity*, *serendipity*, etc. could potentially affect the evaluation negatively. Furthermore, it is not certain that increasingly higher offline evaluation accuracy values translate into better perceived quality by users [McN+02].

In order to overcome this deficiency, online evaluation attempts to capture the quality of the recommendation as perceived by the users by analyzing their interaction patterns with the system as well explicitly asking questions. Analyzing only towards offline evaluation accuracy brings the risk of tuning an algorithm to recommend items which could potentially not be liked by the user in a real-world scenario.

Online evaluation commonly involves a user study. Users can be made aware or encouraged to participate, or participate unknowingly. In real-life systems, the concepts of *A/B testing* are readily used to estimate different algorithms' qualities [Koh12]. A/B testing, as presented in Section 2.3, involves assigning a subset of a system's users to the algorithm under evaluation. In studies of real life systems, users are usually not made aware of their participation in tests [Koh12]. The interactions of the users are then analyzed and compared to a baseline. An example of A/B testing is the Plista contest [Pli12] where the teams participating in the contest are assigned a random subset of the service's users. The users, i.e. readers of online newspapers, are not told they are evaluating the contest. In this scenario, the quality of a recommendation algorithm is deemed by the click-through rate (CTR), the higher the CTR, the better the algorithm performs. In this type of scenario, the analysis does not provide any non-quantitative representation of the system.

More elaborate user studies, including questionnaires and other explicitly collected information serve as an alternative to A/B testing. This type of studies commonly involve asking the users questions throughout, or after, their interaction with the system. In studies like this, the participants are naturally aware of their participation in the study. In order to be able to analyze the results quantitatively, the users are asked to agree or disagree with a question in the form a of a statement. The scale of (dis)agreement is represented as a *Likert* scale [SS01], a basic example of a Likert scale is shown in Fig. 4.1.

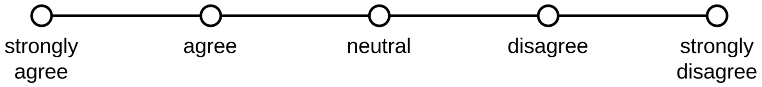


Figure 4.1: A Likert scale where users are asked to agree or disagree with a specific question.

In the scope of this chapter, the Likert scale is used to estimate qualitative measures such as diversity, novelty and obviousness of different recommendation algorithms.

Online, user-centric, evaluation is a means to measure the level of *subjectively perceived quality* by the users of a system. There is a large body of work conducted on how to perform and evaluate user studies and surveys based on statistical significance tests. However, little focus is put on the correlation of subjective and objective evaluation aspects, e.g. which perception-oriented concepts are perceived as similar to each other.

Recent research topics in recommender systems have been the evaluation of recommender systems with special focus on non-quantifiable concepts such as diversity, serendipity, novelty and unexpectedness [Cre+11a; Her+04; KWK11]. Due to the inherent difficulties in benchmarking non-quantifiable aspects, these concepts are evaluated through user studies [Kni12].

There is no default, quality-related, set of questions to ask when performing a recommender systems user study, instead questions are based on the type of quality that is sought for; whether relating to the concepts mentioned above or to rather technical qualities, e.g. time of recommendation, number of items recommended, etc. This type of user studies need to be meticulously planned and executed. If poorly executed, there is a risk of changing the users' opinions, e.g. through suggestive questions, or excessive work load or time involved in answering the questions. Work load and time-related issues can be mitigated by creating an incentive for the users to fulfill the survey, e.g. raffling off vouchers, cash, etc. If no incentive is given, the time involved in answering the survey creates a decaying effect on the fraction of uses who complete the study. When the users are given an incentive, there is a risk that some users will answer the questions quickly (at random) in order to be eligible for the award [SS01]. In order to mitigate these effect, the number of questions and work load should be kept relatively low.

In this chapter, an analysis of prediction accuracy and the perceived quality is performed in order to gain insight into how offline and online metrics correlate to each other. For this purpose, a diversity-oriented recommendation algorithm, k-furthest neighbor, is developed and evaluated. Additionally we evaluate how perception-oriented concepts such as novelty, serendipity, etc. correlate to each other from the users' perspective. In order to evaluate the hypothesis, that accuracy metrics do not correctly represent the perceived quality from a user's perspective, we performed a user study with more than 130 participants and analyzed the results. The results point to that a recommender algorithm which performs considerably lower in terms of precision in an offline evaluation scenario performs very similarly in terms of perceived quality. Our evaluation shows that offline evaluation using traditional metrics such as precision and recall does not correctly capture the quality as perceived by the users. The study performed in the scope of this chapter shows that even even considerably lower precision levels (60% of the baseline) do not result in a change in perceived quality from the users. Additionally, by correlating the answers given to qualitative questions (e.g. novelty, obviousness) we were able to create a mapping between how different qualitative measures are perceived in relation to each other.

4.2 Related Work

Concepts relating to perception are tightly integrated with the fields of Human Computer Interaction (HCI) and psychology. The work conducted in the scope of this dissertation is directed towards the former.

Perception of quality is a broad topic spanning several sub-topics in information retrieval-related research fields such as *e-business* where e.g. Lai [Lai06] developed methods and instruments for measuring the perceived quality of electronic services among employees of several international electronic services, *e-learning* where e.g. Sumner et al. [Sum+03] identified educator's expectations and requirements on educational systems, *information quality* where [Goh+12] compared perceptions of the users' engagement in a mobile game, etc.

In the context of recommender systems, perception of quality is not only dependent on the accuracy of the recommendation. The presentation of the results is considered to have a considerable impact, as has been discussed in the various research events related to this issue [KSTB10; THP12; WBE11]. The user study performed in the scope of this chapter tries to mitigate the effects of this by using the same interaction and presentation components irrespectively of which algorithm delivers the recommendation.

Different ways of presenting recommendations can also result in different perceptions based on cultural settings. Chen and Pu show this in [CP08] where a user study (n = 120) on participants from a "western culture" and participants from an "oriental culture" (i.e. Switzerland and China respectively) was performed in order to evaluate, among other things, the perceived quality of an organizational recommendation system. The study showed that even though cultural differences do not affect the complete spectrum of perception-related concepts, the perceived quality in one out of two presentation formats did differ significantly between the cultures. Similar concepts in information interaction were studied by Barber and Badre [BB98], showing that some design elements in websites are perceived differently across cultures.

The correlation of algorithmic accuracy and the utility of recommendations, in terms of quality, usefulness and other subjective measures has been a growing topic over the last years. McNee et al. [McN+02] presented the results of a user study with more than 120 participants where several algorithms were compared in the setting of a recommender of citations for scientific paper authors. The results from this study showed that even though several recommendation algorithms performed very well in an offline evaluation setting, there exists a discrepancy to the results obtained in the online study; authors were less satisfied with the recommendations in the online setting. The authors noted that this discrepancy could be related to their recommendation setting, i.e. paper authors commonly self-cite and the algorithms in the study were explicitly prohibited to recommend citations from the authors themselves. This created a setting where known and relevant papers were excluded due to their authorship.

Similar work has been performed by e.g. Cremonesi et al. [Cre+11b] where the authors compared the quality of recommendation algorithms in seven different systems by means of a user study ($n = 210$). The three principal findings of the authors where that (*i*) non-personalized recommendation algorithms provided for high user satisfaction, although with low utility, (*ii*) content-based algorithms performed on par with, or better than, collaborative filtering-based recommendation algorithms, and (*iii*) traditional accuracy metrics (recall and fallout) did not approximate the perceived quality very well.

A complete chapter of the Recommender Systems Handbook [Ric+11] is devoted to issues related to interaction with and perception of recommendations. Here, Pu et al. present a set of eleven design guidelines [Pu+11] describing matters related to data presentation, design, explanations etc. Many of the guidelines are based on user studies, A/B testing and other subjective measurements.

As mentioned in this chapter's introduction, in order to evaluate the perceived quality of differing algorithms, an inverted version of the k-nearest neighbor algorithm was developed. The ambition of the algorithm is to recommend items which are not recommended by more common counterparts, e.g. delivering more diverse recommendations. Many existing approaches to increasing diversity in recommender systems do so at the expense of lower accuracy levels and/or increased algorithmic complexity. The motivation behind the kFN algorithm is to gain diversity without the cost of higher algorithmic complexity or lower recommendation quality.

4.2.1 Accuracy vs. Diversity

One of the key reasons why lack of diversity, remains a problem, relates to the observation that algorithms modified to increase diversity have as a byproduct lower recommendation accuracy.

The importance of diversity are broadly outlined in [SM01]. Here, the authors contrast diversity to *similarity*, stating them as countering aims of recommender systems. The paper proposes and evaluates some case-based reasoning (CBR) approaches to recommendation that emphasize diversity. The CBR approach can however be difficult to scale due its complexity. Further, if the task being considered is "recommend the items a user will like" the dichotomy set up in this work of similarity versus diversity can be seen as false. The goal is neither to find items that are similar to each other or spread over the item space, rather to *satisfy a user*.

This idea of the tradeoffs introduced by diversity not actually corresponding to tradeoffs at all is further explored in [Zho+10]. In this work the authors use a *hybrid* of multiple algorithms in order to increase predictive accuracy as well as diversity in a single system. While the output of the system is excellent, it comes with a cost. Again, the effective complexity of such a system is quite high, as it effectively is a combination of the underlying algorithms of the hybrid model. The system does however achieve its goals and is a feasible and effective solution where the complexity is a less important factor (e.g. domains with smaller datasets or no need of swift model updates).

Diversity is a multi-faceted concept which can be considered in several ways. In the remainder of this chapter we take diversity in the context of fixed snapshots of the dataset used, Lathia et al. [Lat+10] instead consider the diversity of user ratings over time. The model of diversity in time-ordered sequence presented by the authors highlights relevant user-behavioral patterns when constructing a recommender system. The work reveals several insights into the effects of user behavior on recommended item diversity, such as an inverse relationship between profile size (i.e. items rated by a user) and the diversity of items recommended to a user.

4.2.2 User-Centricity and Satisfaction

When considering metrics for evaluation of a recommender system, keeping the user central and focusing on user satisfaction is vital. The involvement of actual users complicates methods and evaluation, and raises a number of issues.

One of the earlier works that consider users with regard to trust and privacy in a recommender system context was presented by Lam et al. [LFR06]. In their work, the authors formalized several issues concerning trust in recommender systems especially that of privacy in sharing preferences. Perhaps most relevant to our work, the paper considered how users trust in a recommender system affects their ratings bias.

One means to increase a user's trust in a system is to optimize the utility, or usefulness, of the recommendation generated by a system. McNee et al. [MRK06] concluded that the reliance on predictive accuracy as the most important measure of a recommender system's quality is detrimental to the overall utility of the system. This implies that using a more rich and diverse set of metrics, especially focused on the user is a means to creating a successful (in terms of the user's perception) recommender.

This presents another problem though. Giving up reliance on standard automatic measures opens the problem on what evaluation should be done, and why? The *ResQue* framework is one such solution [PCH11]. This framework presents a unified and extremely rigorous approach to user-driven evaluation, though it is time intensive for participants and may be overly costly for focused hypothesis testing.

4.2.3 Correlation

When considering metrics for evaluation of recommender systems, focusing on the user and on user satisfaction is vital. However, a number of issues are raised when people are brought into the equation, making the evaluation process more complicated.

Recently, a large amount of research has been focusing on user-centric aspects of recommender system evaluation, e.g. [Bol+10; Cos+03; KWK11; MRK06] to mention just a few. Knijnenburg et al. [KWK11] presented a pragmatic approach to user-centric evaluation of recommender systems, formalizing some of the aspects involved. Bollen et al. [Bol+10] evaluated how the number of available choices affects the perceived quality of a recommendation from the users' perspective and found that a smaller set of alternatives can heighten the users' experiences of the system as a larger number of items to choose from adds an increased level of difficulty to choose one item.

McNee et al. [MRK06] approached the concept from a different perspective, evaluating how better performance in terms of accuracy metrics could be detrimental to the overall perceived quality of a system, like many related works they recommend recommender systems researchers and developers to employ user-centric evaluation techniques.

Additional research directions undertaken focus on how user interfaces can be used to increase (or decrease) the perceived quality of recommender systems, e.g. Hu and Pu [HP11].

4.3 Neighborhood Models

As stated in the beginning of this chapter, a new recommendation algorithm was developed to provide more diverse recommendations. The algorithm builds on the k-nearest neighbor algorithm presented here.

Commonly memory-based collaborative filtering models utilize the k-nearest neighbor approach to identify candidate items [BHK98]. This approach uses a neighborhood of similar users to identify and recommend items to users. Neighborhoods are created by, for each user, finding users within a certain *similarity*. Each user's neighborhood contains the k users' who's similarity is highest. An abstract outline of the algorithm is shown in Algorithm 1. Recommendations are created by iterating over all the items rated by the user's neighbors which have not been seen by the user, and estimating the preference of each item by averaging all neighbors' rating values for each item multiplied with each neighbor's similarity value. Ultimately, the highly rated items of the most similar neighbors are recommended. The similarity function for neighbor identification can be freely selected.

Algorithm 1: The k-nearest neighbor algorithm

```
Input: set $\mathcal{U}$ of users
Input: number $k$ of neighbors
Output: $k$ neighbors for each user $u \in \mathcal{U}$
foreach $u \in \mathcal{U}$ do
    foreach $u' \in \mathcal{U} \setminus \{u\}$ do
        $s_{u,u'}$ = similarity($u$,$u'$);
    select $k$ neighbors $u' \neq u$ for $u$ with largest $s_{u,u'}$
```

Common similarity measures used for neighborhood creation are cosine similarity and Pearson product-moment correlation coefficient, when working with data containing ratings. These similarity metrics were introduced in Chapter 3, in this chapter only Pearson is used for similarity measurement.

The Person correlation is calculated by comparing the rating values of two users' co-rated items. We recall the formulation from Eq. (3.7), in this context the weighting factor is disregarded ($\boldsymbol{w} = 1$). We write out the sums and norms obtaining

$$\mathcal{P}(u,v) = \frac{\sum\limits_{i \in \mathcal{I}_{uv}} \left(r_{ui} - \bar{r}_u\right)\left(r_{vi} - \bar{r}_v\right)}{\sqrt{\sum\limits_{i \in \mathcal{I}_{uv}} \left(r_{ui} - \bar{r}_u\right)^2}\sqrt{\sum\limits_{i \in \mathcal{I}_{uv}} \left(r_{vi} - \bar{r}_v\right)^2}} \tag{4.1}$$

where u and v are the two users, I_{uv} the intersection of user u's and v's rated items, and r the rating score, e.g. 4 on a scale from 1 to 5.

For binary co-occurrence data, common similarity measures include the Jaccard and Tanimoto coefficients [HKR02; Her+04].

4.3.1 Nearest & Furthest Neighbors

Nearest neighborhood methods have been frequently used in information retrieval and information theory for many years [CH67], with satisfactory results. However, due to an inherent popularity bias in consumer-item interaction data [MZ09], it is possible that nearest neighbor models recommend items which are merely popular and not explicitly personalized for the user [Raf+09; SJA12]. Further, Bourke et al. found that the method of selecting nearest neighbors has little effect on the perceived quality of the recommender system itself [BMS11].

Inverting the k-nearest neighbor algorithm into a k-furthest neighbor algorithm could serve as a means to mitigate the popularity bias, without adding more complexity (compared to a k-nearest neighbor algorithm) to a recommender system.

The motivation behind the k-furthest neighbor approach is simple. Due to the inherent bias towards popular items in the consumption patterns of people (refer to Chapter 3), and the positive rating bias of popular items [MZ09; SJA12], basic similarity measures can falsely create similarities solely based on popular items, instead of on actual taste [SJA12]. In order to mitigate this effect, the furthest neighbor approach it used. The concept is based on an inverted similarity model, which finds items *disliked* by those *least similar* to a given user. This has the benefit of not being sensitive to bias induced by highly rated popular (i.e. obvious or non-novel) items. The approach first identifies the most dissimilar users to each user, to create *dissimilar* neighborhoods. In every neighborhood it then identifies the most disliked items and recommends these. This double negation creates a more diverse, yet personalized set of recommendations.

4.3.2 Inverting Similarity Metrics

In order to create neighborhoods of the least similar users, a dissimilarity measure must be defined. Dissimilarity cannot simply be expressed by negating or inverting traditional similarity measures, as this would potentially create neighborhoods of completely unrelated, or disjoint, users. In order for a dissimilarity measure to work accurately, we propose the introduction of the following two constraints: (i) two users who share no co-rated items can be considered dissimilar by metrics such as the cosine similarity and the Pearson correlation. However, this dissimilarity is not based on an actual disagreement in taste, rather just on the two users not having an intersection in rated items. A similarity measure like this could result in extremely large neighborhoods of completely disjoint users. In order to circumvent these "false" dissimilarities, we propose a *minimum co-rated items* constraint. This constraint means that in order for two users to be considered dissimilar, they must both have co- interacted with a number of items. The constraint could be extended to treat popular items differently (e.g. lower weights for popular items). This, or similar, constraints are commonly used in kNN-based recommendation approaches as well in order to heighten the level of similarity between prospective neighbors.
The second constraint, (ii) the *opposed ratings* constraint, is introduced to secure an actual dissimilarity and is data-specific and relates to a rating scale. Consider this example:

Suppose that R is a set of ratings r_{ui} submitted by users $u \in U$ for items $i \in I$. Ratings may take values from the discrete set $\{1, 2, 3, 4, 5\}$ of rating scores. Typically, ratings are known only for few user-item pairs.
If u_a's set of rated items is (i_a,i_b,i_c,i_d) rated $(1, 3, 4, 1)$ respectively and u_b's set of rated items is (i_a,i_b,i_c,i_e) rated $(5, 3, 2, 2)$ respectively, then the intersection of items (i_a, i_b, i_c) creates a basis for a dissimilarity calculation. (i_a, i_c) are rated on the opposite end of the rating scale, i_b is however rated in the "middle" of the rating scale and has no antipode. This rating, as well as those for i_d and i_e, are thus not taken into consideration when calculating a dissimilarity.

Given this opposed ratings constraint, the minimum co-rated items constraint does not consider items rated with the middle value of the ratings scale, as the rating cannot be inverted. This rating-based constraint should be modeled according to the rating scale used in the dataset or system. For a more precise dissimilarity, the matching of opposing ratings could be performed on a per-user level. This would be done by first identifying the personal rating scale of each user and subsequently matching this onto other users' personalized rating scales (e.g. some users may only use ratings from each end of the rating scale, others may predominantly use ratings from the middle, while others use the full rating scale meaning the matching of 2 to 4 might not be a true match for all users).

Applying this to the Pearson correlation as presented in Eq. (4.1), the similarity would be calculated as:

$$\acute{\mathcal{P}}(u,v) = \begin{cases} \mathcal{P}(u,v') & : \text{ if } |\mathcal{I}^*_{uv'}| \geq m \\ 0 & : \text{ otherwise} \end{cases}$$

where m is the number of minimum co-rated items, v' contains the inverted ratings of user v according to Eq. (4.2), and $\mathcal{I}^*_{uv'}$ is the list of co-rated items, excluding items rated with the middle score as specified by the opposed ratings constraint, i.e. (i_a, i_c) in the example above.

Each of user v's rating r_{vi} is inverted according to the following process

$$\acute{r}_{vi} = r_{max} - r_{vi} + r_{min} \tag{4.2}$$

where r_{max} is the highest possible score in the rating scale, r_{vi} is user v's rating on item i and r_{min} is the lowest possible score in the rating scale. In the example above u_b's ratings would be $r_{u_b,i_a} = 5-5+1 = 1$ and $r_{u_b,i_c} = 5-2+1 = 4$ turning the rating values 5 and 2 to 1 and 4 respectively.

A similar approach could be applied to many other recommendation scenarios, e.g. in a Singular Value Decomposition-based recommender [Sar+00b], the ratings of the candidate user could be inverted prior to training the recommender followed by recommending the items which, by the recommender, are deemed as least suitable for the user.

4.4 Experiments & Evaluation

The experiments and evaluation conducted in the scope of this chapter serve two purposes, (i) to evaluate how well offline evaluation corresponds to users' actual experiences in a recommendation scenario, and (ii) to find out whether common concepts sought for in online evaluations have correlations between each other. Both purposes were evaluated in the same setting on the same set of users.

4.4.1 Evaluation

To evaluate the quality of the furthest neighbor model, the Movielens 10 million ratings dataset was used. Two sets of evaluation experiments were conducted:

(1) a *traditional* setup using a portion of the ratings for training the model, and a portion used for validation, and (2) a *user-centric* evaluation method based on an online study evaluating the *perceived usefulness* of recommended items and users' overall satisfaction with the system.

4.4.2 Experimental Protocol

Both the nearest neighbor and the furthest neighbor recommenders were implemented using the Apache Mahout[1] framework. Additionally, a random recommender, not taking the users' ratings into consideration when finding candidate items was also implemented using the same framework.

To create the neighborhoods, the Pearson Correlation was used for the nearest neighbor approach. The furthest neighbor approach was created by inverting the Pearson correlation and applying the minimum co-rated and opposed ratings constraints as described in the previous section. The minimum co-rated items constraint was set to 5 based on small empirical experiments, as well as not to exclude users with few ratings. It should be noted that the higher this number is, the more the calculated dissimilarity accurately reflects the actual difference in taste, however it also minimizes the number of possible neighbors. In the nearest neighbor scenario, the minimum co-rated constraint was left to the default value set in the Mahout recommender framework, 2. However, as the nearest neighbor approach requires two users to have *similar* taste, even a lower number should reflect some level of similarity.

The neighborhood size for both approaches was set to 30. Even though a larger neighborhood (up to approximately 100 neighbors [HKR02]) can result in better performance in terms of accuracy metrics, the neighborhood size was kept low to minimize memory usage and keep the recommendation process fast for the survey.

An additional baseline recommender, a randomized recommender not taking the users' ratings into consideration, was used as well. The baseline algorithm, a normal k-nearest neighbor algorithm, was chosen to show the duality of both approaches – thus showing the gain in diversity compared to a traditional recommendation setting. The random recommender was used to show that the effects of the furthest neighbor algorithm were not random.

[1] `http://mahout.apache.org/`

4.4.3 Dataset

The Movielens 10 million ratings dataset[2], contains in excess of 10 million ratings by roughly 70 thousand users assigned to almost 11 thousand movies. The dataset additionally contains the titles and production years of movies. Additional features of the dataset where introduced in Chapter 2.

Using the titles, additional information such as actors, directors, plot, etc., was collected from the Internet Movie Database. This data was used to display additional information about the movies in the survey described in Section 4.4.5.

4.4.4 Traditional Evaluation

A traditional *offline* evaluation approach was used to assess the recommendation quality of the furthest neighbor approach. For this purpose, the above mentioned Movielens dataset was split into training and validation sets which were subsequently used to train and validate the furthest neighbor as well as the nearest neighbor model for comparative purposes.

For each user with at least $2 \times N$ ratings, precision and recall accuracy at N were evaluated for $N = \{5, 10, 100, 200\}$, similarly to the offline experiments conducted in Chapter 3. In each of the cases 20% of the ratings of each user in the dataset were included. For the validation, only movies having been rated above each user's average rating plus 0.5 of the user's standard deviation of rating scores were considered. Thus making sure only truly positively rated items were treated as true positive recommendations. The remaining ratings were included in the training set.

4.4.5 User-Centric Evaluation: A Movie Recommendation Study

In order to make a real-life estimate of the perceived quality of the furthest neighbor approach, a user study was performed. The study consisted of two steps, first participants were asked to rate a minimum of 10 movies from a page showing 100 randomly selected movies out of the 500 most rated movies in the Movielens dataset, shown in Fig. 2. Due to the large number of movies in the dataset (more than $10,000$), we chose randomly among the 500 most popular

[2] `http://www.grouplens.org/system/files/ml-10m-README.html`

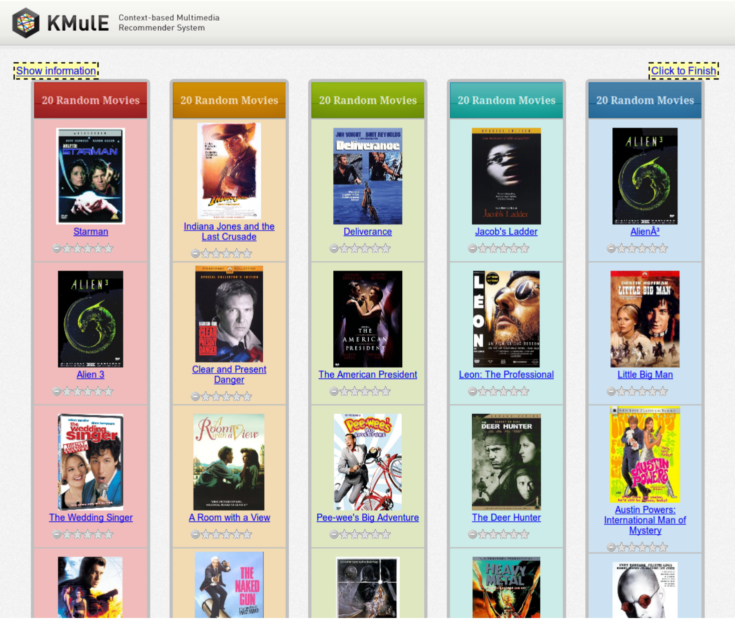
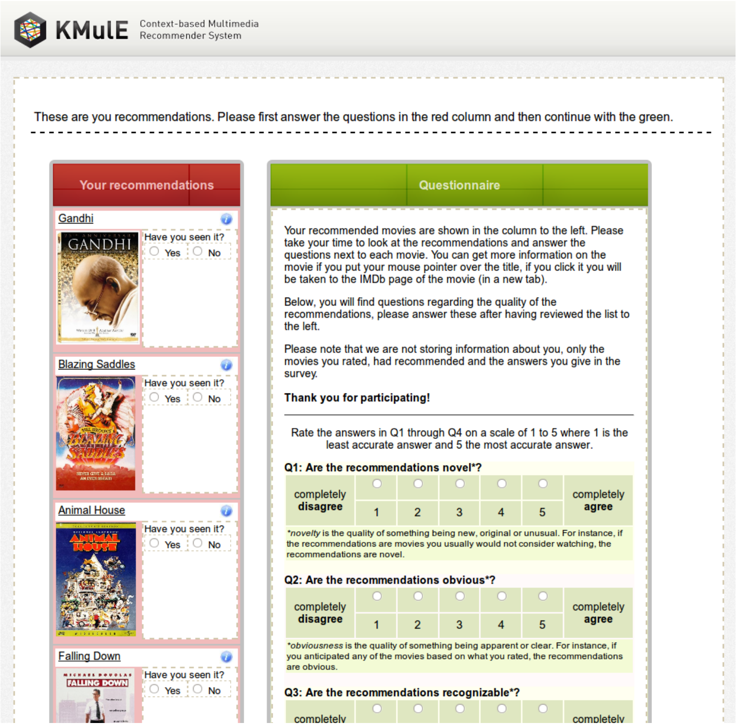


Figure 4.2: The movie rating page where participants were asked to rate movies. The page contains a random selection of 100 of the 500 most popular movies in the Movielens dataset.

movies in order to show the participants movies which they would most likely be familiar with[3].

Having rated at least 10 movies, recommendations based on each participant's ratings were generated and a page containing the top 10 recommendations and a set of questions was shown, see Fig. 4.3. The recommender engine providing the survey with recommendations was similar to the one used in offline evaluation as described in Section 4.4.4. The differences being the complete Movielens dataset was used for training as this scenario did not include any offline evaluation. The time to generate recommendations for one user, using either of the neighborhood-based algorithms, was under one minute for users having rated a large portion of the 100 movies shown in Fig. 4.2; this response time was considerably lower for those only having rated the minimum allowed number of movies (10). The random recommender did not need any noticeable time for

[3] In an earlier version of the survey, which was tested on a small number of users, random movies from the whole set of movies were shown instead. The vast majority of the reactions collected from these early testers were that most of the movies were completely unknown to them, thus the choice to present a random set from the 500 most popular movies was made instead.



These are you recommendations. Please first answer the questions in the red column and then continue with the green.

Your recommendations

Gandhi

Have you seen it?
○ Yes ○ No

Blazing Saddles

Have you seen it?
○ Yes ○ No

Animal House

Have you seen it?
○ Yes ○ No

Falling Down

Have you seen it?
○ Yes ○ No

Questionnaire

Your recommended movies are shown in the column to the left. Please take your time to look at the recommendations and answer the questions next to each movie. You can get more information on the movie if you put your mouse pointer over the title, if you click it you will be taken to the IMDb page of the movie (in a new tab).

Below, you will find questions regarding the quality of the recommendations, please answer these after having reviewed the list to the left.

Please note that we are not storing information about you, only the movies you rated, had recommended and the answers you give in the survey.

Thank you for participating!

Rate the answers in Q1 through Q4 on a scale of 1 to 5 where 1 is the least accurate answer and 5 the most accurate answer.

Q1: Are the recommendations novel*?

completely **disagree**	○	○	○	○	○	completely **agree**
	1	2	3	4	5	

**novelty* is the quality of something being new, original or unusual. For instance, if the recommendations are movies you usually would not consider watching, the recommendations are novel.

Q2: Are the recommendations obvious*?

completely **disagree**	○	○	○	○	○	completely **agree**
	1	2	3	4	5	

**obviousness* is the quality of something being apparent or clear. For instance, if you anticipated any of the movies based on what you rated, the recommendations are obvious.

Q3: Are the recommendations recognizable*?

completely	○	○	○	○	○	completely

Figure 4.3: The questions presented to participants in the survey after having rated a minimum of 10 ratings.

recommending. To create the illusion of generating personalized recommendations, one of the neighborhood-based algorithms was trained (but not used) in parallel to serve as a timer.

The selection of algorithm to be used for each participant was proportional to a randomized variable. For 40% of the participants, the recommendations were to be based on the traditional k-nearest neighbors approach. Another 40% of the participants were to be given recommendations based on the k-furthest neighbors approach, and finally 20% of the participants were to be presented with randomized recommendations, not taking their initial ratings into consideration.

4.4.5.1 Questionnaire

For each of the 10 recommended movies (shown in the leftmost column in Fig. 4.3), participants were asked to answer a set of questions. Questions were chosen to reflect standard quality measuring aspects in recommender systems (e.g. usefulness, recognizability, customer retention) as well as reflecting the current state of the art in recommender system quality measurement (e.g. serendipity, novelty) [Kni+12]. The first set of questions (relating to each of the 10 recommended movies) was

Have you seen the movie?

if Yes: Please rate it (5 star rating)

if No:

1. Are you familiar with it? (y/n)
2. Would you watch it? (y/n)

Additionally, participants were asked to answer a set of 8 question regarding the complete set of recommended items. The questions were:

1. Are the recommendations novel?
2. Are the recommendations obvious?
3. Are the recommendations recognizable?
4. Are the recommendations serendipitous?
5. Are the recommendations useful?
6. Pick the movie you consider the best recommendation
7. Pick the movie you consider the worst recommendation
8. Would you use this recommender again?

Participants were asked to answer questions 1 through 5 stating the level of agreement, from strongly disagree (1) to strongly agree (5). A short description was given for the terms *novel*, *obvious*, *recognizable* and *serendipitous* in order to mitigate erroneous answers based on misunderstanding of the question.

Additionally a field for comments was placed on the bottom of the survey.

Submission of the answers was only possible after all questions had been answered.

The questionnaire was designed to create a truthful picture of users' perceived usefulness of the recommendations, considering aspects such as choice overload [Bol+10], construction of questions [Kni+12] and similar perception-related concepts [Her+04; SS01].

The link to the survey was circulated on social media sites (e.g. Facebook, Twitter, Google+) during the last two weeks of March 2012. People from several professions (computer science professionals, lawyers, business management professionals) were asked to circulate the link in their networks in order to gain participants from several communities, thus attempting to minimize any biasing effects which could surface in answers from a homogeneous community. The data was collected in early April 2012, by then a total of 132 participants had completed the survey.

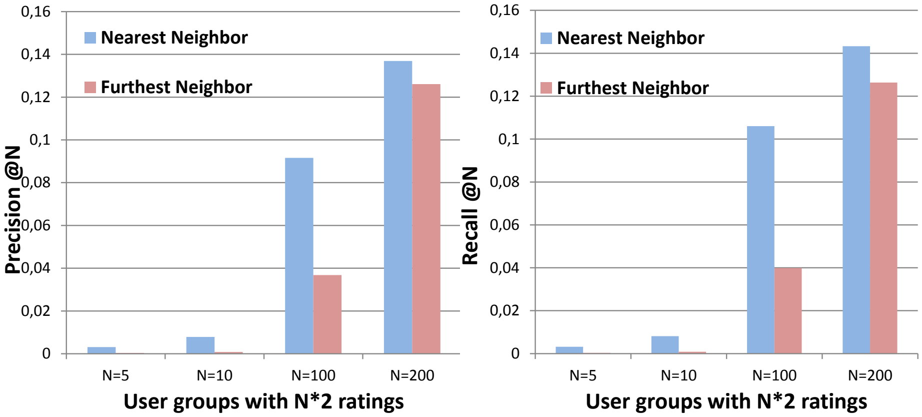
Out of the 132 participants in the study, 47 (36%) were presented with recommendations based on the nearest neighbor model, 43 (33%) based on the furthest neighbor model and 42 (32%) based on the random recommender (due to the randomized seeding of the algorithm selection, the percentages are somewhat different from the intended 40%, 40%, 20% distribution).

Knijnenburg et al. claim that "at least 20 users" per condition should be adequate for being able to mine statistically sound data from a user study [KWK11], indicating the amount of participants in our study is sufficient.

No demographic data except for location (reverse lookup via IP address) was collected. The participants of the survey came from a total of 19 countries on 4 continents. Participants from Germany, Ireland and Sweden were most prominent, in descending order.

4.5 Results & Discussion

In the following section we present and discuss the results for traditional (Section 4.5.1) and user-centric (Section 4.5.2) evaluation separately.



(a) Precision@N

(b) Recall@N

Figure 4.4: The Precision@N and Recall@N values for users with $2 \times N$ ratings for the furthest and nearest neighbor models for the traditional offline evaluation approach.

4.5.1 Traditional Evaluation Results

Fig. 4.4(a) and Fig. 4.4(b) summarize the prediction performance of the nearest and furthest neighbor algorithms in terms of precision and recall. For small values of N, nearest neighbor outperforms furthest by a factor of 8.4 in terms of precision(3.14×10^{-3} and 3.75×10^{-4} for $N = 5$ respectively). For larger values the difference in performance grows smaller, with the nearest neighbor approach outperforming the furthest neighbor approach by 8.6% at $N = 200$. The precision values for all four evaluated Ns are presented in Fig. 4.4(a). Performance in terms of recall (Fig. 4.4(b)) is almost identical to precision. The results of the random recommender are not shown as they were several orders of magnitude lower than both neighborhood-based recommenders.

The nearest neighbor approach outperforms the furthest one in traditional information retrieval metrics. However, the recommenders seem to be almost entirely disjoint in terms of recommended items, as shown in Table 4.2, i.e. users get completely different items recommended from each of the neighborhood-based recommender algorithms.

The almost complete orthogonality of recommended items indicates that both approaches are complementary, and if optimally combined into one, the resulting precision value would be the sum of the precision of both approaches. A recommender engine which would be able to create an ensemble recommender

	$N = 5$	$N = 10$	$N = 100$	$N = 200$
items	42	75	$7,925$	$27,295$
%	0.0013	0.0012	0.74	3.2

Table 4.2: The absolute number of intersecting items for the nearest and furthest neighbor recommender and the percentage of intersecting items from the whole set of recommended items (i.e. $N \times |U|$).

by estimating the true positive recommendations in each algorithm prior to recommendation would in this case serve as an optimal neighborhood algorithm. Ensembles are however commonly built by combining the intersecting items from both sets of recommendations, which becomes infeasible in this scenario due to the diverse recommendations provided by the kFN algorithm. Thus an alternative ensemble creation method would need to be developed to create the optimal merger of these two algorithms.

4.5.2 User-Centric Results

As stated in Section 4.4.5, participants were presented with recommendations from either a nearest neighbor recommender, a furthest neighbor recommender, or a random recommender. The results of all three approaches are presented throughout this section. All t-test values presented are either for two-tailed t-tests (when only two distributions are compared), or ANOVA testing using two-tailed t-tests with Bonferroni correction ($n = 2$ for two comparisons) [Abd07].

The results of the survey are either related to each of the movies which had been recommended, or to the whole list of recommended movies. Fig. 4.5 shows the results from the movie-related questions.

Considering whether the participants had seen the recommended movies or not, the ratio of seen vs. unseen was highest for the kNN approach, followed by the kFN, with the random recommender having the lowest ratio ($p < 0.1$), as shown in Fig. 4.5(a). Looking at this in the context of the ratings given to seen movies (Fig. 4.5(d)) per algorithm, we can conclude that the kFN algorithm indeed shows more unknown movies, whilst remaining an almost identical average rating score compared to kNN algorithm. This is interpreted as an indication of the kFN recommender being able to attain a comparable level of user satisfaction, with a higher level of diversity. The random recommender recommends more unseen movies, with the cost of a lower overall average rating.

Similar observations can be made for whether or not the participants were familiar (Fig. 4.5(b)) with the recommended movies ($p < 0.05$), and whether they would consider watching (Fig. 4.5(c)) the movie (showing a trend towards differences between both neighborhood approaches and random at $p < 0.1$).

These observation, in conjunction with the the significance of tests (kNN/kFN vs. random $p < 0.05$, kFN vs. kNN $p > 0.1$ failing to reject the assumed null hypothesis of there not being a difference between the rating quality of both neighborhood-based recommenders) indicate that the overall quality of the kFN recommender seems to be on par with its kNN counterpart.

Algorithm	rating	novel	obvious	known	serendipitous	useful
kNN	3.64	3.83	2.27	2.69	2.71	2.69
kFN	3.65	3.95	1.79	2.07	2.65	2.63
Random	3.07	4.17	1.64	1.81	2.48	2.24

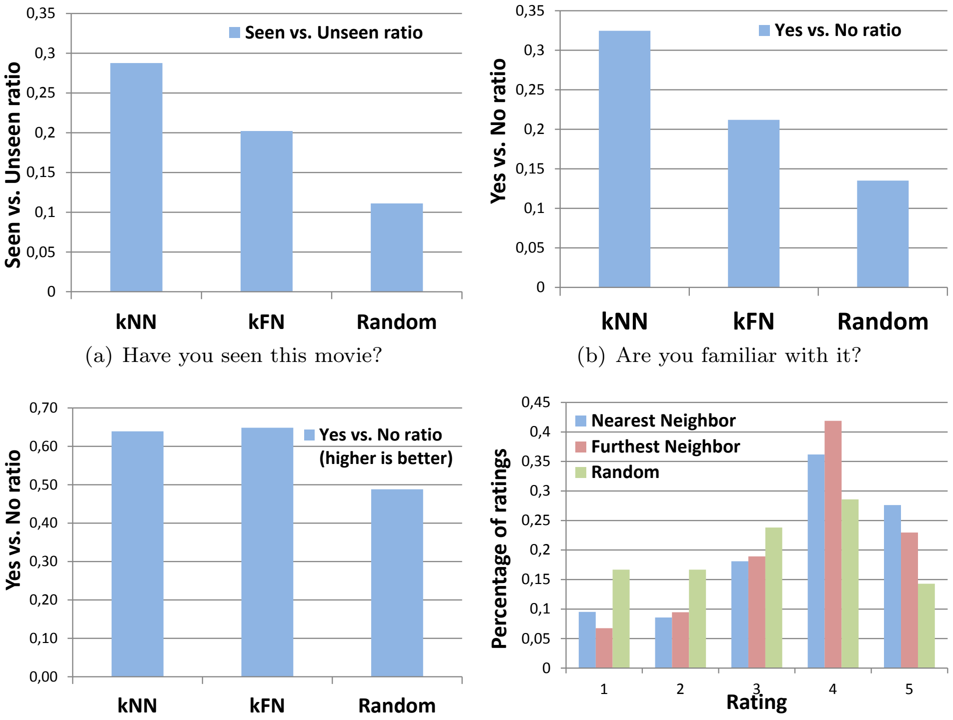
Table 4.3: The average rating given by users for movies they had seen which were recommended by the different algorithms as well as the average of the agreement levels for novelty, obviousness, recognizability, serendipity and usefulness respectively.

Looking at the results of the questions related to the full set of recommended movies (the leftmost column in Fig. 4.3), shown in Fig. 4.6, the trend continues. For instance, Fig. 4.6(a) shows that the perceived novelty of kFN matches that of kNN. Novelty, by its nature implies new, or unheard of movies explaining the relatively high score of the random recommender. For obviousness, the distribution is almost opposite to that of novelty. The furthest neighbor model seems to be slightly less obvious than the nearest neighbor counterpart. Obviousness in this context is interpreted as something negative as obvious items could be found by the user without the help of a recommender system.

Generally, the differences between the neighborhood models and the random recommender show (at a minimum) trends towards being different, except for when it comes to serendipity (Fig. 4.6(c)) and novelty (Fig. 4.6(a)). We believe this is based on two factors: (1) the non-trivial meaning of the words (especially concerning serendipity for non-native English speakers[4], and (2) the inherent difficulty of rating. something which is not known (which both novelty and serendipity at least in part cover). Analogously, the assumed null hypothesis of there not being a difference between kNN and kFN fails to be rejected within significant levels of p.

Regarding the level of agreement on the usefulness (Fig. 4.6(d)) across all the algorithms, Table 4.3 shows both neighborhood models performing very sim-

[4] In 2004 serendipity was voted the third most difficult English word to translate [Avé05].



(a) Have you seen this movie?

(b) Are you familiar with it?

(c) Would you watch it?

(d) Ratings of watched movies.

Figure 4.5: The ratio of yes versus no answers to the questions "Have you seen this movie?" (Fig. 4.5(a)) and "Are you familiar with it?" (Fig. 4.5(b)) both of which showing the diversity and non-obviousness of the recommended items. "Would you watch it?" (Fig. 4.5(c)), and the average ratings of recommended movies previously seen by the users (Fig. 4.5(d)) showing the general quality of the recommendations.

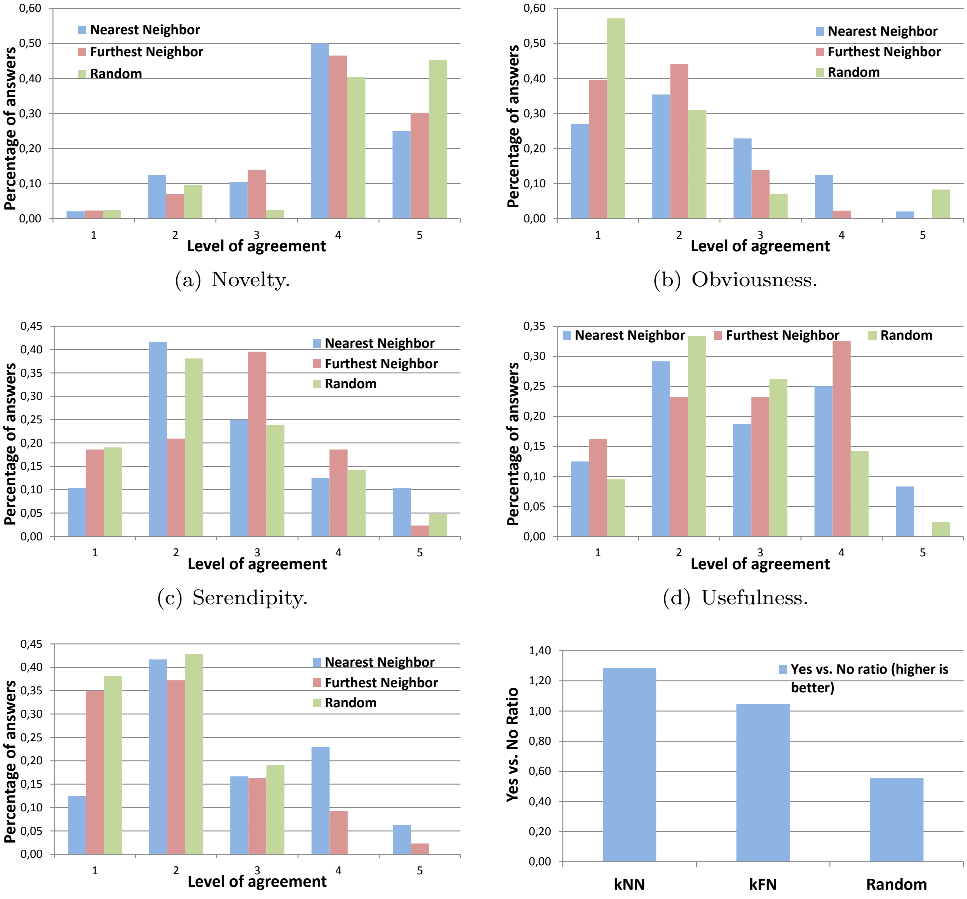
ilarly, t-test comparisons of the distributions show a p value close to one for comparison between neighborhood models and $p = 0.16$ for the comparisons between random and each of the neighborhood models.

However, recognizability is interpreted differently depending on the algorithm (two-tail t-tests give $p < 0.05$ between nearest and furthest, $p < 0.01$ between nearest and random, and a non-significant $p = 0.19$ between furthest and random). It seems the furthest neighbor recommender recommends less known items than the nearest neighbor does, which confirms the results shown in Fig. 4.5(b). We believe this provides, at least in part, an explanation to the lower precision values in offline evaluation, as less known items are less likely to have been seen by users, and are thus less likely to appear in the validation set. This also confirms some of the popularity-based concerns in Chapter 3.

Finally, on the question of whether the participants would use a recommender system like the one evaluated, there is a slight bias towards the nearest neighbor recommender, i.e. Fig. 4.6(f). However, this goes in line with users' natural bias towards known items as well trust-related aspects, e.g. before recommending unknown items a system should establish trust with the user [GH06; Her+04]. Establishing trust requires repetitive use of a system, which was not possible in the scope of the conducted study.

We believe that given the reported results, precision accuracy levels alone cannot be used to accurately evaluate an approach which is intended to recommend diverse items, e.g. items from the long tail. The results of the user study seem to confirm that even though the nearest neighbor approach outperforms the furthest neighbor approach in terms of traditional accuracy metrics, the general perceived usefulness of the recommenders by the users does not seem to differ substantially. Even though some of the reported t-test p-values show significance within a large error margin ($p > 0.1$), there is a definite trend pointing to similar perceived usefulness of both neighborhood approaches when considering all reported results in context.

Due to the fact that the nearest and furthest neighbor recommender are practically orthogonal in terms of recommended items, as shown in Table 4.2, this type of evaluation can have a higher positive effect than it would have had for two algorithms which recommend a similar set of items. In contrast, if comparing two recommendation algorithms that create less disjoint recommendation sets, i.e. by replacing the furthest neighbor approach with an algorithm recommending more "standard" items, it is reasonable to assume that the results of the offline evaluation would position both algorithms closer to each other in terms of predictive accuracy. In the online evaluation scenario, it is not entirely clear which algorithm would perform better, this is however related to aspects like how diverse the alternative algorithm would be, how affected by popularity bias



(a) Novelty.

(b) Obviousness.

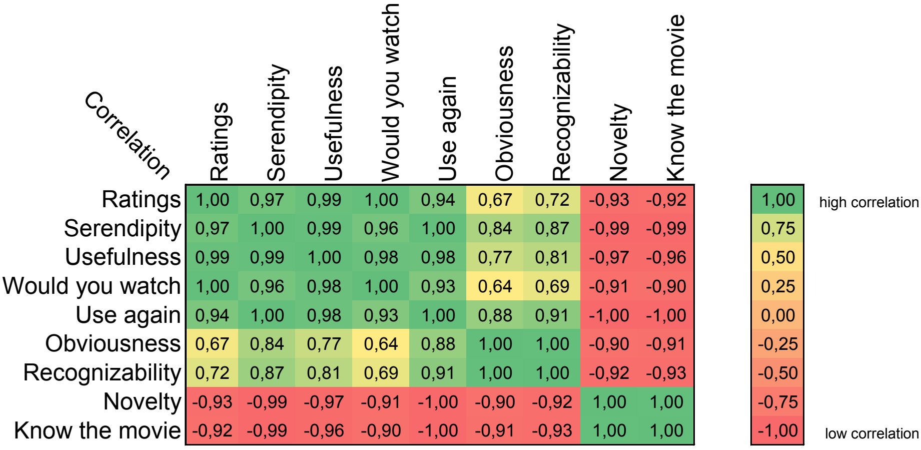
(c) Serendipity.

(d) Usefulness.

(e) Recognizability.

(f) Would you use the system again?

Figure 4.6: The distributions of agreement levels of the non-movie related answers (i.e. those shown in the right column of Fig. 4.3) for each algorithm evaluated in the user study. In Fig. 4.6(a) to Fig. 4.6(e) the values $\{1, 2, 3, 4, 5\}$ correspond to {*most* ***disagree***, ..., *most* ***agree***} respectively. In Fig. 4.6(f) the bars correspond to the ratio of yes vs. no answers.



Correlation	Ratings	Serendipity	Usefulness	Would you watch	Use again	Obviousness	Recognizability	Novelty	Know the movie
Ratings	1,00	0,97	0,99	1,00	0,94	0,67	0,72	-0,93	-0,92
Serendipity	0,97	1,00	0,99	0,96	1,00	0,84	0,87	-0,99	-0,99
Usefulness	0,99	0,99	1,00	0,98	0,98	0,77	0,81	-0,97	-0,96
Would you watch	1,00	0,96	0,98	1,00	0,93	0,64	0,69	-0,91	-0,90
Use again	0,94	1,00	0,98	0,93	1,00	0,88	0,91	-1,00	-1,00
Obviousness	0,67	0,84	0,77	0,64	0,88	1,00	1,00	-0,90	-0,91
Recognizability	0,72	0,87	0,81	0,69	0,91	1,00	1,00	-0,92	-0,93
Novelty	-0,93	-0,99	-0,97	-0,91	-1,00	-0,90	-0,92	1,00	1,00
Know the movie	-0,92	-0,99	-0,96	-0,90	-1,00	-0,91	-0,93	1,00	1,00

Figure 4.7: The Pearson Product-Moment Correlation of the answers given to questions across all three recommendation algorithms. Each row/column corresponds to the correlation of the questions asked in the questionnaire.

aspects (e.g. Chapter 3) and other related factors. Based on the results obtained in these experiments, together with the current state-of-the-art in diversifying recommendations, it is reasonable to believe that a diverse recommender algorithm would outperform the baseline in terms of usefulness, perceived quality, etc.

4.5.3 Correlating User-Centric Concepts

In order to analyze the similarity of the questions, the answers from all users were averaged on a per-question per-algorithm basis creating a vector containing three values, the average value for the question across the three recommenders. The question vectors were then compared using the Pearson Product-Moment Correlation Coefficient (Section 4.3.2) also used to calculate similarities between users in Section 4.3 (pp. 55).

In this case, u and v refer to the concepts asked for in the used study (e.g. serendipity, ratings, novelty) and the algorithms (kNN, kFN, Random) respectively, r the average value of each of the concepts per algorithm, i.e. the average of the Likert scale answers given to e.g. serendipity, obviousness, etc.

The resulting similarities are shown in Fig. 4.7.

Fig. 4.7 summarizes the pairwise similarities between the questions. The most dissimilar questions towards other questions are those regarding *novelty* and *recognizability*. They are however very similar towards each other, which is perhaps expected as they both, essentially, answer the same questions; one for unrated movies, and the other for rated movies (recall the questionnaire layout in Fig. 4.3).

Questions regarding *serendipity*, *usefulness*, whether the users would *use the system again* and the *ratings* seem to belong to a cluster of questions being perceived similarly. The implication of this being that the questions seem to correspond to similar values, e.g. a serendipitous movie will often be rated highly, or if a user considers a system useful the probability of using it again is higher, etc.

As for the less similar questions, i.e. ratings and willingness to watch a certain movies vs. obviousness and recognizability, the similarities tell us these questions are not entirely unrelated, however considerably less than, for instance, ratings and usefulness. We believe this is due to aspects such as obviousness and recognizability belong to the same concepts, i.e. once again how obvious a set of recommendations is probably related to how many movies are recognized, as something which is unknown can still be obvious if the initial step (the ratings performed in Fig. 4.2) only contains unknown items, and vice versa.

The main observations are: high ratings correlate with questions considering serendipity, usefulness and retention (use again, watch the movie). Obviousness and recognizability tend be of little importance, but do not seem to discourage people from using the system again. High levels of unknown/novel movies tend to point to lower ratings and less consumption (watching the movie).

It should however be noted that all observations are averaged with respect to the algorithms, expanding the study to include more algorithms diverse recommendation three algorithms could alter the similarities.

4.6 Conclusion

As mentioned in Section 4.3.2, similar inverted similarities could potentially be applied to other recommendation algorithm. The choice of the k-nearest neighbor algorithm was based on it's illustrative qualities as well as on hardware limitations. Experiments with matrix factorization-based recommendation approaches required 5 to 7 minutes for retraining of the recommender. The participants in the study would need to wait for their recommendation for this period of time, which would likely have detrimental effects on the number of users who completed the study, as discussed in Section 4.1.

In this chapter and extensive analysis of the recommendation quality of both a nearest neighbor and a furthest neighbor algorithm has been presented. The quality of these algorithms has been evaluated through traditional information retrieval metrics as well as through a user study in order to gain insight on the perceived usefulness of the algorithms. In terms of precision and recall, the nearest neighbor algorithm outperforms the furthest neighbor approach. The lists of recommended items are however almost completely disjoint due to the nature of the furthest neighbor approach, which recommends more non-obvious and diverse items. By analyzing the feedback obtained from a user study with more than 130 participants it has been shown that a higher predictive accuracy of a recommender system, in this scenario, does not increase the perceived usefulness of the recommender system, from the users' perspective.

From the answers obtained in the study, we were able to recognize several aspects that point to the notion that *at least* similar satisfaction can be obtained with lower precision, e.g. proportionally the same amount of users claimed they would watch a movie recommended by the lower scoring furthest neighbor approach than a movie recommended by the nearest neighbor approach - Fig. 4.5(c). Similarly, the furthest neighbor recommender and nearest neighbor recommender received almost identical overall rating values, despite the differences in precision and recall, shown in Table 4.3.

In real-world applications, often only the users' ratings are available. With this in mind, it is reasonable to infer that the ratings may also hold information about the perceived quality of the recommender in terms of serendipity, usefulness, intention to watch movies, and intention to use the system again.

Additionally, a correlation of a set of perception-related aspects to each other was performed in order to lower the burden on the users when conducting user studies. Notably, the work presented found that there is considerable overlap between aspects such as usefulness, serendipity and return rate.

This model of evaluation of diverse and other non-obvious items needs to be further analyzed in order to gain further insight on when traditional information retrieval accuracy metrics cannot express the quality of a recommender system accurately. Some of the challenges related to this are further discussed in Chapter 6.

Chapter 5

Recommender System Optimization

This chapter describes the concept of *optimization of recommender system accuracy* in the context of a *maximum* obtainable level of offline performance of a recommender system, a so-called *magic barrier*. The magic barrier is characterized and evaluated in a real-world movie recommendation website. The motivation is that optimization beyond the magic barrier cannot be said to actually reflect an actually better performing algorithm. In the scope of this dissertation, the magic barrier presents the point where traditional offline evaluation should be extended by an online evaluation estimating the perceived values of the users.

The structure of the chapter is as follows. First, an introduction to the concepts behind the magic barrier is given in Section 5.1, following this, the work conducted in the chapter is positioned in the current relevant state-of-the-art relevant in Section 5.2. In Section 5.3 and Section 5.4 the characterization of the magic barrier is given, which in turn is followed by a case study in a real-world movie recommendation website in Section 5.5. Finally the chapter is concluded in Section 5.6.

5.1 Introduction

Recommender systems play an important role in most top-ranked commercial websites such as Amazon, Netflix, Last.fm or IMDb [RRS11]. The goal of these recommender systems is to increase revenue and present personalized user experiences by providing suggestions for previously unknown items that are potentially interesting for a user. With the growing amount of data on the Internet, the importance of recommender systems increases even more to guide users through the mass of data.

The key role of recommender systems resulted in a vast amount of research in this field, which yielded a plethora of different recommender algorithms [AT05; DK11; Lat+10]. To select an appropriate recommender algorithm among the many available, and adapt it to a given scenario or problem, the algorithms are usually examined by testing their performance using either artificial or real test data reflecting the problem. The best performing algorithm and parameters among a number of candidate algorithms is chosen. To be able to compare performance, several different measures and metrics were defined. RMSE is perhaps the most popular metric used to evaluate the prediction accuracy of a recommender algorithm [SG11]. As mentioned in Chapter 2, it was the central evaluation metric used in the Netflix Prize. For the RMSE as performance measure, a recommendation task is typically posed as that of learning a rating function that minimizes the RMSE on a given training set of user ratings. The generalization RMSE-performance of the learned rating function is then assessed on some independent test set of ratings, which is disjoint from the training set. One major drawback of measuring and comparing the performance of recommendation systems using only static test data (i.e. previously collected frozen in time data) is that the data lacks the dynamic aspects of user behavior. According to studies conducted by [APO09; Ama+09; Hil+95] user ratings can be inconsistent (or noisy) in the sense that a user may rate the same item differently at different points of time. Following these findings, Herlocker et al. [Her+04] and other researchers coined the term *magic barrier*.

More recently, RMSE and similar error measures have been scrutinized in the recommender systems research community as lower rating prediction errors do not necessary correspond to an actual improvement in terms of recommendation accuracy and perceived quality as noted by [Cre+11b] and shown in Chapter 4. However, partly due to the popularity of the Netflix Prize, RMSE still remains one of most widely-used recommendation evaluation metrics.

In the context of perception-oriented quality, minimizing the rating prediction error needs not correctly reflect the quality of a recommendation due to concepts discussed in Chapter 4. There are several other reasons for this, e.g. rating bias;

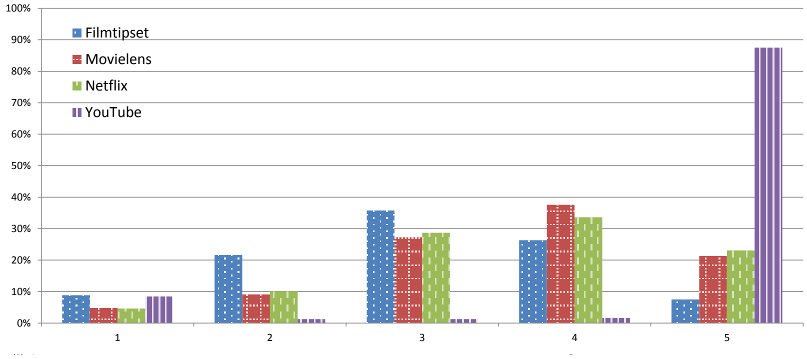
due to the users' a priori opinions on items (the intuition of whether someone will like an item or not, discussed in Chapter 3) there is a bias towards positive ratings. This popularity is common in the movie domain, additionally as shown in Fig. 5.1 there is a skewness towards positive ratings. Notably, the rating distribution in Moviepilot (Fig. 5.1(b)) indicates a somewhat differing rating scenario compared to the three other movie rating datasets, this is likely an effect of the rating scale used in the service, 0 to 10 stars in half star steps whereas the others use 1 to 5 stars in one star steps. This type of positive rating bias has been shown to affect evaluation of recommender systems [Lin+12; MZ09]. The rating distribution from YouTube is shown primarily for comparison, this rating distribution was one of the reasons the service changed the rating scale to a thumbs-up/thumbs-down instead of the initial 1 to 5 scale [Raj09] as it better reflected the users' perception of the content.

An additional reason for why rating error minimization does not necessarily lead to better performing recommendation algorithms is the level of *user-induced noise* in the ratings given by a system's users. User-induced noise is the irregularities in the collected data generated by users due to any (external or internal) factors surrounding the time of the rating creation.

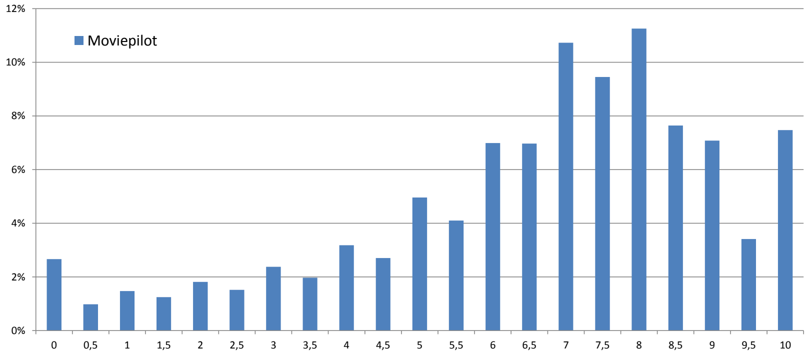
Similarly to perception-oriented aspects, an "optimal" error measure ($\epsilon = 0$ where ϵ refers to the error) does not necessarily reflect the actual true opinion of a user. Additionally, quantifying taste, whether on a one-to-five or any other scale, creates a problematic and non-intuitive setting for users as taste is neither static nor objective [Her+04]. Some research instead proposes an item-by-item comparative ranking approach in order to better model the specific tastes of users [ART06; DSM10; DSS12].

Rating prediction error minimization optimization is based on the notion that there is a "true rating" which reflects the actual taste [Klu+12; Sai+12b; Sai+12f]. This true rating is assumed to be the one in the dataset, e.g. a rating that was at one point given to an item by a user. A system which is capable of predicting this true rating is considered optimal [APO09; Her+04]. However, due to noise, the true rating needs not to be one specific value. Instead, it can be expressed as a value within a certain span.

These observations, in the context of ratings, were initially discussed by Hill et al. [Hil+95], however no approach to counter the effects of the user-induced noise was presented. More recently, Amatriain et al. [APO09; Ama+09] have approached the problem of noisy ratings in the context of movie recommendation, attempting to either de-noise the data, or identify which ratings contain more "accurate" representations of the users' taste. However, no means of estimating the level of noise versus the optimal rating prediction levels was shown.



(a) The rating distribution in Filmtipset, Movielens, Netflix and YouTube [Raj09] (prior YouTube switching to a thumbs up/thumbs down rating scale).



(b) The rating distribution in the Moviepilot dataset

Figure 5.1: The rating distributions in YouTube (according to [Raj09]) and datasets from Filmtipset, Movielens10M, Netflix and Moviepilot. Note the distinct difference between YouTube and the datasets from Filmtipset, Movielens and Netflix. Moviepilot uses a different rating scale, thus it cannot be presented in the same graph, still the rating distribution seems to follow those found in the other movie rating datasets.

Traditionally, data analysis approaches which acknowledge “natural noise” as an inherent part of the data have focused on attempts to rid the data of noise in order to gain a clearer picture of what the data represents [Ama+09]. However, as discussed in Chapter 4, data generated by subjective users might not be possible to rid of noise in order to generate a fixed, clean, value for each noisy value due to the inconsistencies in users’ actions. When considering this, the assumption that the perfect error measure is nil is challenged. In this case, a perfect error measure ϵ should be at a level within the span. Let us illustrate this with a simplified example:

> Consider a recommender system with the rating prediction error (RMSE) of $\epsilon = 0.15$. The dataset which the systems uses to generate rating predictions has an estimated level of noise at $n = 0.11$ (normalized to RMSE). In this scenario, the minimal error feasible to attain is $\epsilon = n$, any value $\epsilon \leq n$ is likely due to overfitting on the testset.
> In this case, it should only be reasonable to optimize the system rating prediction accuracy till it reaches a value close to n. Any optimization beyond that point should be regarded pointless as the level of noise could result in a detrimental effect on the actual rating prediction accuracy.

It seems reasonable that user-induced (or so-called “natural” [Ama+09]) noise, as well as other types of noise, exists in rating datasets. This observation is supported in the scientific literature presented in the next section. Due to the various noise types, e.g. user interface induced, contextual, behavioral [Her+04], it appears more feasible to estimate the level of noise than to actually eliminate it.

This chapter formalizes a mathematical characterization of the *maximum level of optimization* based on root-mean-squared error minimization. This characterization builds on the presumption that noise is a natural ingredient in any dataset. The characterization is supported by a large-scale, real-world user study with a commercial movie recommendation website. In the scope of this chapter, a magic barrier estimate for the system is found based on the data collected in the user study.

While investigations on the magic barrier are important for future recommendation research, only first evaluations on the inconsistency of ratings have been conducted so far. In particular, this allows to identify recommender algorithms that overfit the test set by chance or by peeking at the test set.[1].

[1] This occurs when information about the test set is used for constructing a recommender

The magic barrier marks the point at which the performance and accuracy of an algorithm cannot be enhanced due to noise in the data. Every improvement in accuracy might denote an over-fitting and not a better performance. Thus, comparing and measuring the expected future performance of algorithms based on static data might not work.

The main contributions of this chapter are:

- We present a mathematical characterization of the magic barrier. Based on the principle of empirical risk minimization form statistical learning theory, we show that the magic barrier reduces to the RMSE of the optimal (but unknown) rating function. Then we characterize the magic barrier in terms of the expected inconsistencies incurred in the ratings [Vap95]. Since the magic barrier cannot be computed directly, we derive a procedure to estimate it.

- We present and discuss a case study with Moviepilot, a commercial recommender system for movies. The case study aims at investigating user inconsistencies in a real-world setting. In addition, we estimated the magic barrier of Moviepilot to assess the quality of Moviepilot's recommendation engine and to propose a limit on how much Moviepilot's recommendations can be improved.

Based on our findings, we propose that a real-world recommender system should regularly interact with users by polling their opinions about items they have rated previously in order to audit their own performance and, where appropriate, to take measures to improve their system.

5.2 Related Work

As mentioned in the previous section, one of the first works mentioning user-induced noise in movie ratings was provided by Hill et al. [Hil+95], where the authors created an email-based movie recommendation service. The service asked its users to rate movies from a list of 500 pre-selected movies before creating recommendations. The authors mention "The Upper Limit" as a bound

algorithm. For example, when we devise a new recommendation algorithm based on some similarity measure. We train our new algorithm using the cosine similarity and assess its performance on a test set. Upon which, the same algorithm is trained using the Pearson correlation and measure its performance on the same test set. Finally, we report the result of the algorithm and the similarity measure with the best prediction performance on the test set rather than choosing the model with the best prediction performance on the training set. Such approaches lead to overly optimistic results.

on prediction performance based on the idea that a person's ratings are noisy, or otherwise inconsistent. Based on statistical theory, the authors claim that it will *never be possible to perfectly predict the users' ratings*, instead they cite "the square root of the observed test-retest reliability correlation" as the optimal level of prediction due to the levels of noise in user-generated data. No attempt at estimating the level of noise was however performed in the scope of this work.

The first mention of the *magic barrier*, the term currently used to state the practical upper bound on rating prediction accuracy (or lower bound on rating prediction error), was made by Herlocker et al. in their seminal paper on recommender system evaluation [Her+04]. In this work, the authors speculate whether recommender systems are hitting this potential magic barrier, e.g. a point where "natural variability may prevent us from getting much more accurate". Additionally, Herlocker et al. speculate whether minuscule rating prediction errors actually translate to a perceived improvement from the users or whether the increasingly smaller accuracy improvements do not alter the perceived quality, i.e. see Chapter 4.

Given the ever increasing amount of recommender systems research publications, projects, etc., the amount of literature covering the magic barrier of these systems is surprisingly small. Although there is considerable amount of previous work performed on the topic of data quality, user-generated noise, and its effect on information retrieval-related solutions, there is no common terminology used in the plethora of related works.

In the context of movie ratings, the datasets used for noise-related studies have almost exclusively been either the Netflix Prize dataset, or datasets from the Movielens movie recommendation website. Cosley et al [Cos+03] conducted an early study on the Movielens website where a selection of users where asked to provide re-ratings to previously rated movies. In their experiments, the authors chose four approaches; in the first case, users were presented with movies without displaying any ratings. In the second case, users were shown movies and a suggested rating identical to the one already given. In the third selection, movies were presented with ratings one star higher then the original rating, and in the last case the suggested ratings were one star lower than the original ones. In this experiment, the authors found that the users who participated in the study where either no rating, or the original rating was shown with the movies, re-rated movies with their original rating in 60% of the cases for not shown movies, and almost 70% for shown movies. In the cases where the rating had been altered, a significant amount of users chose to re-rate the movie with the new predicted rating.

O'Mahony et al. [OHS06] took a different approach to noise and instead tried to classify it as either "natural noise" or "malicious noise" where the former is the more “standard” noise model based on erroneous, hasty or otherwise imperfect ratings. The latter noise type is induced into the system in order to alter it’s functions, e.g. for purposes of making a specific item (a book or a movie) more popular or raising its rating in order to attract more customers/viewers.

Among the first works using the terminology introduced by Herlocker et al. includes two papers by Amatriain et al. where the authors in the earlier of the two attempted to characterize the noise in ratings based in reliability [APO09]. For their experiments, the authors collected three set of ratings for 100 movies from the Netflix Prize dataset. The ratings were collected with 24 hours and 15 days difference. From their collected data, the authors were able to identify a range for the magic barrier, i.e. the lowest possible RMSE value. The identified range (0.557 to 0.8157) is the point beyond which optimizing a recommender system, using their dataset, does not indicate a better recommendation. In a follow-up to this work [Ama+09], the authors used a similar approach, this time in order to remove the noise in the ratings. In order to do this, the authors again collected re-rating data in order to identify noisy ratings in a subset of the Netflix Prize dataset.

Some of the most recent work relating to the levels of noise in rating datasets and its effect on recommendation, presented by Kluver et al [Klu+12] approach this problem from a different direction. Instead of measuring the level of noise in the dataset, or attempting to formalize a magic barrier, the authors measure how much *preference information* is contained in a rating. The authors base their approach on the assumption that “. . . humans can form stable preferences for all items in the item domain. However, . . . these preferences are only partially articulated, and that processing is still required when mapping these partial preferences to a rating.” This assumption, based on preference forming an measurement [Fis91], together with Shannon’s information entropy [Sha01] creates the basis for *preference bits*, which indicate how much actual information is concealed in a rating. Preference bits are found through repeated re-ratings by users on the same items. Each re-rating can then be used to estimate the amount of preference bits in a rating.

Some of the inconsistencies in users’ rating behavior can be mitigated by temporal aspects, as Lathia et al. show [Lat+10]. This mitigation does however not compensate for all inconsistencies, which Amatriain et al. [Ama+09] showed by having different time spans between re-ratings.

The problems of noisy user behavior is connected to the type of evaluation used. Pu et al. [PCH11] present a *user-centric* (as opposed to *data-centric*) evaluation framework which measures the quality of recommendations in terms of *usability*, *usefulness*, *interaction quality* and *user satisfaction*, which allows for optimization of recommender systems based on direct user interaction instead of *offline* accuracy metrics such as RMSE, NDCG, etc. User-centric evaluation does however come with a cost in terms of time, additionally it requires a set of users to be available for the evaluation process, as described in Chapter 4. Given these drawbacks, most recommender system evaluation still uses traditional information retrieval measures and methods, even though these might not always reflect the actual quality of the recommendation [SG11] due to the aforementioned inconsistencies in users' rating behavior.

5.3 The Empirical Risk Minimization Principle

We pose the recommendation task as that of a function regression problem based on the empirical risk minimization principle from statistical learning theory [Vap95]. This setting provides the theoretical foundation to derive a lower bound, the magic barrier, on the root-mean-square error that can be attained by an optimal recommender system.

5.3.1 The Traditional Setting of a Recommendation Task

We begin by describing the traditional setting of a recommendation task as presented in [DK11].

Suppose that $\mathcal{R}$ is a set of ratings r_{ui} submitted by users $u \in \mathcal{U}$ for items $i \in \mathcal{I}$. Ratings may take values from some discrete set $\mathcal{S} \subseteq \mathbb{R}$ of rating scores. Typically, ratings are known only for few user-item pairs. The recommendation task consists of suggesting new items that will be rated high by users.

It is common practice to pose the recommendation task as that of learning a rating function

$$f : \mathcal{U} \times \mathcal{I} \to \mathcal{S}, \quad (u, i) \mapsto f(u, i)$$

on the basis of a set of training examples from $\mathcal{R}$. Given a user u, the learned rating function f is then used to recommend those items i that have largest scores $f(u, i)$. The accuracy of a rating function f is evaluated on a test set, which is a subset of $\mathcal{R}$ disjoint from the training set.

A popular and widely used measure for evaluating the accuracy of f on a set $\mathcal{R}$ of ratings is the root-mean-square error (RMSE) criterion presented in Section 2.3.1. In the scope of this chapter, we pose it as

$$E(f|\mathcal{R}) = \sqrt{\frac{1}{|\mathcal{R}|} \sum_{(u,i)\in\mathcal{R}} \big(f(u,i) - r_{ui}\big)^2}, \tag{5.1}$$

where the sum runs over all user-item pairs (u, i) for which $r_{ui} \in \mathcal{R}$.[2]

5.3.2 Recommendation as Risk Minimization

Learning a rating function by minimizing the RMSE criterion can be justified by the inductive principle of empirical risk minimization from statistical learning theory [Vap95]. Within this setting we describe the problem of learning a rating function as follows:
We assume that

- user-item pairs (u, i) are drawn from an unknown probability distribution $p(u, i)$,
- rating scores $r \in \mathcal{S}$ are provided for each user-item pair (u, i) according to an unknown conditional probability distribution $p(r|u, i)$,
- $\mathcal{F}$ is a class of rating functions.

The probability $p(u, i)$ describes how likely it is that user u rates item i. The conditional probability $p(r|u, i)$ describes the probability that a given user u rates a given item i with rating score r. The class $\mathcal{F}$ of functions describes the set from which we choose (learn) our rating function f for recommending items. An example for $\mathcal{F}$ is the class of nearest neighbor-based methods.

The goal of learning a rating function is to find a function $f \in \mathcal{F}$ that minimizes the expected risk function

$$R(f) = \sum_{(u,i,r)} p(u,i,s)\big(f(u,i) - r\big)^2, \tag{5.2}$$

where the sum runs over all possible triples $(u, i, r) \in \mathcal{U} \times \mathcal{I} \times \mathcal{S}$ and $p(u, i, r) = p(u, i)p(r|u, i)$ is the joint probability.

[2] For the sake of brevity, we abuse notation and write $(u, i) \in \mathcal{R}$ for user-item pairs (u, i) for which $r_{ui} \in \mathcal{R}$.

The problem of learning an optimal rating function is that the distribution $p(u, i, s)$ is unknown. Therefore, we can not compute the optimal rating function

$$f_* = \arg\min_{f \in \mathcal{F}} R(f).$$

directly. Instead, we approximate f_* by minimizing the empirical risk

$$\widehat{R}(f|\mathcal{X}) = \frac{1}{|\mathcal{X}|} \sum_{r_{ui} \in \mathcal{X}} \big(f(u,i) - r_{ui}\big)^2,$$

where $\mathcal{X} \subseteq \mathcal{R}$ is a training set consisting of ratings r_{ui} given by user u for item i. Observe that minimizing the empirical risk is equivalent to minimizing the RMSE criterion.

A theoretical justification of minimizing the RMSE criterion (or the empirical risk) arises from the following result of statistical learning theory [Vap95]: under the assumption that the user ratings from $\mathcal{R}$ are independent and identically distributed, the empirical risk is an unbiased estimate of the expected risk.[3]

5.4 The Magic Barrier

This section derives a magic barrier (lower bound) on the RMSE that can be attained by an optimal recommender system. We show that the magic barrier is the standard deviation of the inconsistencies (noise) inherent in user ratings. To this end, we first present a noise model and then derive the magic barrier.

5.4.1 A Statistical Model for Users' Inconsistencies

As shown in user studies [APO09; Ama+09; Hil+95], users' rating tend to be inconsistent. Inconsistencies in the ratings could be due to, for example, change of taste over time, personal conditions, inconsistent rating strategies, and/or social influences, just to mention a few.

For the sake of convenience, we regard inconsistencies in user ratings as noise. The following fictitious scenario illustrates the basic idea behind our noise model: Consider a movie recommender with n movies and a rating scale from zero to five stars, where zero stars refers to a rating score reserved for unknown movies

[3] The set of users and items are both finite. In order to apply the law of large numbers, we may think of $\mathcal{R}$ as being a set of ratings obtained by randomly selecting triples (u, i, s) according to their joint distribution.

only. Users are regularly asked to rate m randomly selected movies. After a sufficiently long period of time, each user has rated each movie several times. The ratings may vary over time due to several reasons ([Lat+10]) such as change of taste, current emotional state, group-dynamic effects, and other external as well as internal influences.

Keeping the above scenario in mind, the *expected rating* of a user u on movie i is defined by the expectation

$$\mathbb{E}[R_{ui}] = \mu_{ui},$$

where R_{ui} is a random variable on the user-item pair (u, i) and takes on zero to five stars as values. Then a rating r_{ui} is composed of the expected rating μ_{ui} and some error term ε_{ui} for the noise incurred by user u when rating item i. We occasionally refer to the error ε_{ui} as user-item noise. Thus, user ratings arise from a statistical model of the form

$$r_{ui} = \mu_{ui} + \varepsilon_{ui}, \tag{5.3}$$

where the random error ε_{ui} has expectation $\mathbb{E}[\varepsilon_{ui}] = 0$. This assumption, is however rather naive in a recommendation scenario.

5.4.2 Deriving the Magic Barrier

Suppose that f_* is the true (but unknown) rating function that knows all expected ratings μ_{ui} of each user u about any item i, that is

$$f_*(u, i) = \mu_{ui} \tag{5.4}$$

for all users $u \in \mathcal{U}$ and items $i \in \mathcal{I}$. Then the optimal rating function f_* minimizes the expected risk function Eq. (5.2). Substituting Eq. (5.3) and Eq. (5.4) into the expected risk function Eq. (5.2) and using $p(u, i, s) = p(u, i)p(s|u, i)$ gives

$$R(f_*) = \sum_{(u,i)} p(u, i)\mathbb{E}\left[\varepsilon_{ui}^2\right] = \sum_{(u,i)} p(u, i)\mathbb{V}\left[\varepsilon_{ui}\right], \tag{5.5}$$

where the sum runs over all possible user-item pairs $(u, i) \in \mathcal{U} \times \mathcal{I}$ and $\mathbb{V}[\varepsilon_{ui}]$ denotes the variance of the user-item noise ε_{ui}. Eq. (5.5) shows that the expected risk of an optimal rating function f_* is the mean variance of the user-item noise terms.

Expressed in terms of the RMSE criterion, the magic barrier $B_{\mathcal{U}\times\mathcal{I}}$ of a recommender system with users $\mathcal{U}$ and items $\mathcal{I}$ is then defined by

$$B_{\mathcal{U}\times\mathcal{I}} = \sqrt{\sum_{(u,i)} p(u, i)\mathbb{V}\left[\varepsilon_{ui}\right]}.$$

The magic barrier is the RMSE of an optimal rating function f_*. We see that even an optimal rating function has a non-zero RMSE *unless all users are consistent with their ratings.*

Observe that an optimal rating function needs not to be a member of our chosen function class $\mathcal{F}$ from which we select (learn) our actual rating function f. Thus the RMSE of f can be decomposed into the magic barrier $B_{\mathcal{U}\times\mathcal{I}}$ and an error E_f due to model complexity of f giving

$$E_{RMSE}(f) = B_{\mathcal{U}\times\mathcal{I}} + E_f > B_{\mathcal{U}\times\mathcal{I}}.$$

5.4.3 Estimating the Magic Barrier

As for the expected risk, we are usually unable to directly determine the magic barrier $B_{\mathcal{U}\times\mathcal{I}}$. Instead we estimate the magic barrier according to the procedure outlined in Algorithm 1.

Algorithm 1 Procedure for estimating the magic barrier.

Procedure: Let $\mathcal{X} \subseteq \mathcal{U} \times \mathcal{I}$ be a randomly generated subset of user-item pairs.

1. For each user-item pair $(u, i) \in \mathcal{X}$ do
    a) Sample m ratings $r^1_{ui}, \ldots, r^m_{ui}$ on a regular basis
    b) Estimate the expectation μ_{ui} by the sample mean
    $$\widehat{\mu}_{ui} = \frac{1}{m} \sum_{t=1}^{m} r^t_{ui}$$
    c) Estimate the variance of the ratings
    $$\widehat{\varepsilon}^2_{ui} = \frac{1}{m} \sum_{t=1}^{m} \left(\widehat{\mu}_{ui} - r^t_{ui}\right)^2$$
2. Estimate the magic barrier by taking the average
$$\widehat{B}_{\mathcal{X}} = \sqrt{\frac{1}{|\mathcal{X}|} \sum_{(u,i)\in\mathcal{X}} \widehat{\varepsilon}^2_{ui}}. \tag{5.6}$$

We postulate that all rating functions $f \in \mathcal{F}$ with an empirical risk of the form

$$\widehat{R}(f|\mathcal{X}) \leq \widehat{B}_{\mathcal{X}}^2$$

are likely to overfit on the set $\mathcal{X}$ and consider further improvements on the RMSE below $\widehat{B}_{\mathcal{X}}$ as *meaningless*.

5.5 Case Study Using a Commercial Movie Recommender

Our experimental case study serves to validate the noise model and to investigate the relationship between the estimated magic barrier and the prediction accuracy of moviepilot. Due to limited resources, we conducted a moderately scaled user study in a real-world setting.

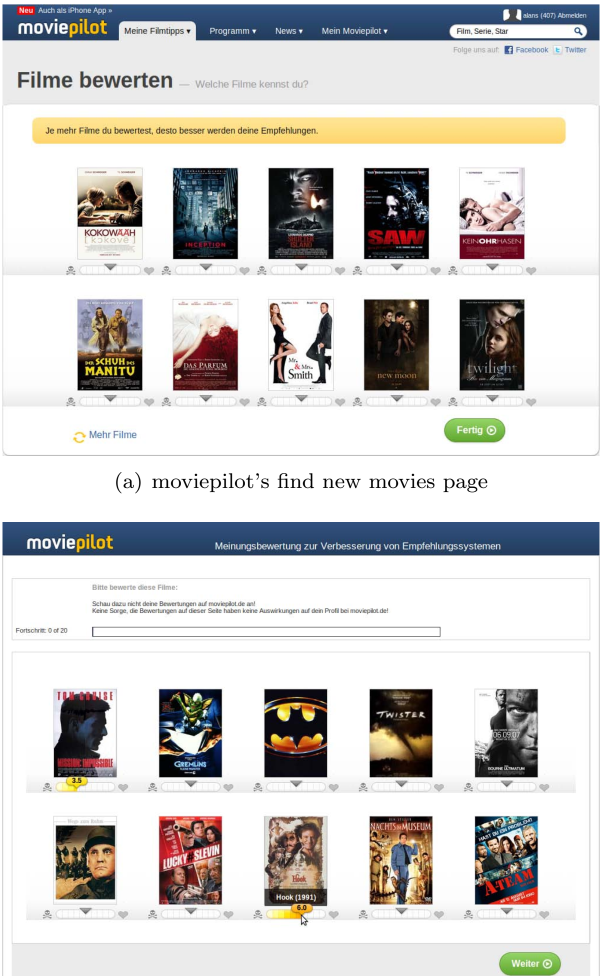
5.5.1 moviepilot

moviepilot is a commercial movie recommender system having more than one million users, $55,000$ movies, and over 10 million ratings. Movies are rated on a 0 to 10 scale with step size 0.5 (0 corresponding to a rating score of 0, not an unknown rating). The two most common ways to rate movies are either through the "discover new movies" page, shown in Fig. 5.2(a) or through the "100 movies of your lifetime" page. The former presents a combination of new, popular and recommended movies whereas the latter one presents, like the title suggests, the 100 previously unrated movies deemed most probable to be liked by the user.

The recommendation engine uses a neighborhood-based collaborative filtering approach, with a similarity measure inspired by the cosine similarity, and is retrained regularly, so as to always be able to recommend movies based on an up-to-date model of users' rating histories.

5.5.2 Data

To estimate the magic barrier, we created a Web-based study for collecting users' *opinions* on movies. An opinion is a score in the same rating scale as standard user ratings. The difference between the two is that ratings are stored



Filme bewerten — Welche Filme kennst du?

Je mehr Filme du bewertest, desto besser werden deine Empfehlungen.

(a) moviepilot's find new movies page

Bitte bewerte diese Filme:

Schau dazu nicht deine Bewertungen auf moviepilot.de an!
Keine Sorge, die Bewertungen auf dieser Seite haben keine Auswirkungen auf dein Profil bei moviepilot.de!

Fortschritt: 0 of 20

(b) Opinion interface

Figure 5.2: The Moviepilot rating interface and the opinion interface used in the user study. The opinion interface mimics the look-and-feel of Moviepilot in order to give users a feeling of familiarity lowering the level UI-induced noise.

in the user profile and used for predictions, whereas opinions do not show up in users' profiles, are only stored in the survey and do, subsequently, not affect the recommendations users are given.

The opinion collection study was implemented as a Web application, Fig. 5.2(b), mirroring the look-and-feel of Moviepilot's rating interface as closely as possible. By emulating the rating interface of Moviepilot for opinion polling, we aimed at mitigating any potential distortion of the data due to different interface elements. A comparison of both is shown in Fig. 5.2.

Users were notified about the study by announcements in newsletters and posts in the community forums. The announcements provided information on timing, duration, process of the opinion collection, the collector, the URL, etc. Users were asked to not to peek at their old ratings when taking part in the study. They were also informed that submitted opinions would not be stored in their profiles.

Whilst taking part in the study, users were presented with a number of movies randomly drawn from the complete set of their rated movies. Each user could submit at most one opinion on each movie. A user could skip any number of movies without providing any opinion. After at least a 20 opinions had been given, the active user could complete the study.

The study ran from mid April 2011 to early May 2011. We recorded only opinions of users that provided opinions on at least 20 movies. A total of 306 users provided $6,299$ opinions about $2,329$ movies.

5.5.3 Experimental Setup

We estimated the magic barrier according to the procedure described in Algorithm 1. For this, we used the ratings and opinions of those user-item pairs for which the $6,299$ opinions had been recorded.[4] This setup corresponds to sampling two ratings for each of the $6,299$ user-item pairs. The estimate of the magic barrier is the average of the squared sample noise over all $6,299$ user-item pairs.

[4] Though opinions differ from ratings conceptually, we treat them equally when estimating the magic barrier.

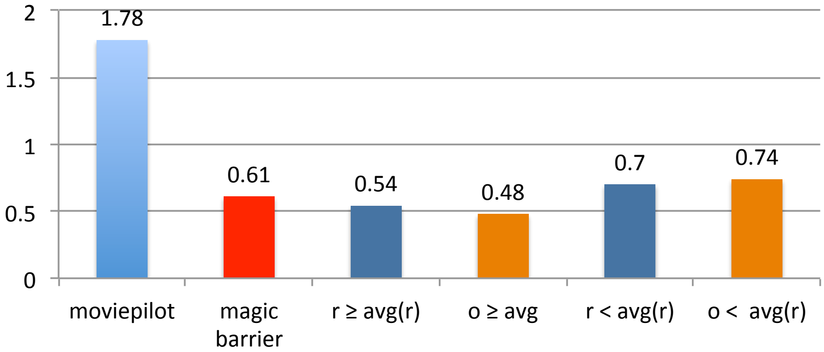


Figure 5.3: RMSE of Moviepilot and estimated magic barrier, where *all* refers to the estimation computed over all $6,299$ user-item pairs, $r \geq avg(r)$ and $o \geq avg(r)$, resp. refer to the magic barrier restricted to all user-item pairs with ratings and opinions, respectively, *above or equal to* the user-specific average over all user ratings. Similarly, $r < avg(r)$ and $o < avg(r)$, respectively, refer to the magic barrier restricted to all user-item pairs with ratings and opinions, resp., *below* the user-specific average over all user ratings and opinions.

5.5.4 Results and Discussion

Fig. 5.3 summarizes the outcome of the opinion polling study, it shows the RMSE of Moviepilot's recommendation engine and an estimated magic barrier taken over all $6,299$ user-item pairs. The plot also shows the estimated magic barrier restricted to the following subsets of the $6,299$ user-item pairs: (1) user-item pairs with above average rating, (2) user-item pairs with above average opinion, (3) user-item pairs with below average rating, and (4) user-item pairs with below average opinion. The user-specific averages were taken over all ratings given by the respective user.

The first observation to be made is that the estimated magic barrier of Moviepilot is circa 0.61, which is slightly more than one step in Moviepilot's rating scale (± 0.61). In contrast, the RMSE of Moviepilot's recommendation engine is about 1.8 which is between three and four rating steps. Under the assumption that the estimated magic barrier is a good estimate of the unknown magic barrier, improvements of a recommender method close to or below the estimated magic barrier are meaningless. Under the same assumptions, there is room for improving the prediction accuracy of Moviepilot. These assumptions, however, have to be taken with care due to the limited amount of data for estimating the expected rating of each user-item pair.

The second observation to be made is that our estimate of the magic barrier is lower when restricted to user-item pairs with ratings/opinions above average ratings than for below average ratings. We hypothesize that users tend to be more consistent with their ratings/opinions for movies that they have rated above average. This finding complements the observations of Amatriain et al. [APO09], i.e. that user ratings seem to be more consistent at the extreme ends of the rating scale.

The results obtained in this as well as in other studies and the theoretical treatment on magic barriers give a strong case for collecting opinions (or re-ratings) in order to

1. estimate the magic barrier for performance evaluation, and
2. improve recommendations based on a set of ratings for each user-item pair rather than on a single rating.

A good estimate of the magic barrier is useful for assessing the quality of a recommendation method and for revealing room for improvements. Recommenders with a prediction accuracy close to the estimated magic barrier can be regarded as 'optimal'. Further improvements of such recommenders are meaningless.

It is reasonable to assume that the generic magic barrier evaluated in the scope of this chapter could be additionally lowered provided the sampled ratings were treated contextually, i.e. each sample was given within a certain context. Identifying these contexts and estimating the magic barrier within each context would likely result in a set of different magic barriers, each for one specific context. However, as context-awareness falls outside of the scope of this chapter, and this dissertation, no attempt at contextualizing the magic barrier have been attempted in this work.

5.6 Conclusion and Future Work

The magic barrier is the RMSE of an optimal rating function, and as such, it provides a lower bound for the RMSE level an arbitrary rating function can attain. In terms of noise incurred when users rate items, the magic barrier is the square root of the expected variance of the user-item noise. When using this characterization, it becomes a straightforward task to derive a procedure for estimating the magic barrier.

In an experimental case study using Moviepilot, a commercial movie recommendation system, we investigated the inconsistencies in users' ratings and used these to estimate the magic barrier for assessing the actual prediction accuracy of the recommendation system. The results obtained confirm that users are inconsistent in their ratings, specifically more inconsistent the lower the rating, and more consistent for higher (above average) ratings. The estimate of the magic barrier presented in this chapter reveals that there still is some room for improvement of Moviepilot's recommendation algorithm.

The magic barrier is obtained by comparison the several ratings on the same user-item pairs, on the basis of our findings, we suggest that regularly polling ratings for previously rated items (the same user-item pairs). These ratings can subsequently be used to audit the performance of the recommendation engine and may, where appropriate, lead to measures which can be taken to improve an existing system's performance.

To obtain statically sound results, performing a large-scale user study is imperative. In order to regularly poll opinions or ratings of previously rated items, certain issues should be addressed, e.g.

- How to implement a user-friendly interface for polling opinions/ratings without having a deterrent effect on users and unbiased results at the same time?
- How to present items and sample opinions/ratings to obtain a good estimate of the magic barrier?
- How should samples be collected over time?
- How should old samples be treated, i.e. is there a limit for when a rating should expire?

In the context of this chapter, the issues dealt with in Chapter 4 should be brought to light when a system reaches its magic barrier. As optimization beyond this point is likely to be useless, other means of evaluation need to be employed, i.e. user-centric evaluation.

Additionally, as mentioned in Section 5.5.4, one potential direction for future work is the contextualization of the magic barrier. Given the state-of-the-art in context-aware recommender systems, it is not unlikely that even a trivial contextualization (e.g. time-based) would lower the magic barrier considerably.

Chapter 6

Conclusions

This chapter summarizes and concludes the work conducted in the scope of this dissertation, presents avenues for potential future research in related directions and concludes the dissertation with some final remarks on the topics covered by this work.

6.1 Conclusion

Recommender systems are becoming more and more accurate, reaching a point where the standard evaluation methods do not necessarily correctly estimate the actual quality of the employed recommendation algorithms. There are several reasons for this; many evaluation methods where not developed with the recommendation use case in mind and are thus not able to capture the accuracy sought for, others measure objective, quantitative, values which do not always correctly correlate to the perceived quality from the users' perspectives. Other methods estimate the quality based on assumptions that the underlying data used for the training of recommender systems is consistent, at least to a point where optimization based on this data is feasible, without any constraints.

This dissertation aimed to analyze current evaluation methods and develop new methods and guidelines for accurate recommendation quality estimation. The methods presented in the context of this thesis were motivated by these two major observations:

1. There is a discrepancy in the way recommender systems are evaluated and in the way they are used.

2. Current evaluation metrics do not always correctly reflect the perceived quality from the users' perspectives.

Many of the evaluation concepts still hold, there are however multiple drawbacks when only evaluating recommendations based on measures, methods and metrics not specifically developed for the purpose of evaluating recommender systems.

In order to fully accurately estimate the quality of personalized recommendations, above a certain quality threshold, recommendations must be evaluated in the context of recommendation, i.e. by using evaluation methods specifically developed for the purpose of estimating recommendation accuracy. Otherwise, what is measured as a boost in quality could actually be experienced as a reduction in the quality, as seen by the end users of recommender systems.

6.2 Summary of Contributions

This dissertation has advanced the state-of-the-art in recommender system evaluation and diversification through a number of contributions.

First, we introduced the reader to recommender systems and presented the main challenges and research questions related to the topic of evaluating recommender systems in Chapter 1. The chapter additionally described some of the potential application scenarios of the work presented in this dissertation. It also highlighted some of the historical aspects of recommender systems that have led to the current challenges.

Following the introduction, Chapter 2 presented an exhaustive overview of current resources for recommender systems research, including common datasets and their attributes, as well as the most common evaluation methods and evaluation measures. The chapter then continued to discuss the *utility of recommendation*, including aspects such as perceived quality and diversity, and how current evaluation protocols very often tend to overlook this concept completely.

Chapter 3 then continued on to present an overview of how aspects related to popularity bias in user interaction datasets, commonly used in recommender systems research and development, affect recommendation. Specifically, the chapter analyzed these aspects in different stages of a user's interaction with a recommendation system. The main contribution of the chapter was to show that users require different recommendations based on their accumulated level of interaction with the system. Specifically, during the *post cold start phase* where users have accumulated a certain number of item rating, the measured recommendation quality can be boosted by up to 20% by using popularity bias mitigating techniques, i.e. so-called weighting schemes. This chapter also created a motivation for the work which followed, as the measured recommendation quality could not guarantee an improvement in the experienced quality.

Following popularity-related concepts, Chapter 4 covered a set of topics related to recommendation quality and recommendation diversity as perceived by the end users of a recommender system. In this chapter, a new recommendation algorithm was created for the purpose of delivering more diverse, novel, and non-obvious recommendations. This algorithm was evaluated both through traditional methods and measures as well as through a qualitative study in order to capture the quality of the recommendations as experienced by the users. The results from the study showed that even though the diverse algorithm performed considerably lower in the traditional evaluation setting when compared to a standard baseline algorithm, the perceived quality was on a level comparable, if not better, to the baseline algorithm's perceived quality. In addition to this, the chapter covered the topic of correlation of specific user-centric, qualitative, aspects of evaluation in order to identify a potential perception overlap when using these attributes in qualitative user studies. Furthermore, these attributes where additionally correlated to standard quantitative evaluation methods, e.g. ratings, in order to prospectively lower the burden on users when performing qualitative evaluation.

The final research chapter, Chapter 5, then approached inconsistencies in rating datasets in order to identify whether there existed a prospective maximum level of optimization for recommendation using this type of datasets. For this purpose a mathematical characterization of the *magic barrier* was proposed based on empirical risk minimization techniques. Even though the concept of the magic barrier has been known and researched for almost two decades, this chapter presented the first mathematical characterization of the concept. Additionally, the theoretical model was evaluated through a large-scale real-world user study deployed on a commercial movie recommendation website. The results indicate that even though the deployed recommender systems performs well, there still is room for some improvement. The chapter concluded that the magic barrier could be used as the point where traditional offline evaluation should stop, and user-centric online evaluation should begin.

6.3 Future Work

There are many potential directions for future work in areas related to evaluation and optimization of recommender systems. Below, a selection of them are listed and motivated.

User-centric Popularity-sensitive Evaluation

As shown in Chapter 3, analyzing the datasets to identify certain popularity-induced aspects prior to evaluation or recommendation does present non-negligible effects to the accuracy of the recommendations. There is however a threshold for when mitigating popularity becomes detrimental, e.g. in order to establish trust among new users, a system should recommend items which are recognized by the users. Tying the popularity-related aspects to user-centric evaluation in the context of trust and recommendation quality presents an avenue of research where the utility of popularity, trust and quality all represent sought for aspects in a recommender systems. In order to understand whether the results shown in Chapter 3 also correspond to the perceived usefulness of the recommendations, as perceived by the users is an additional potential research direction.

Diversity-oriented Evaluation

Chapter 4 showed that even though a more diverse recommendation algorithm received lower accuracy scores than a basic baseline in a traditional evaluation setting, this was not the case in a user-centric perspective, where it was on par with the baseline. This creates a potential avenue for research on diversity-oriented evaluation, since the traditional evaluation metrics are not able to express the accuracy of a recommender system in terms of diversity. Prospective topics include a quantification approach to diversity where diversity can be analyzed from an accuracy and satisfiability perspective. There already exist several approaches related to this, none of them have however been able to capture the usefulness and accuracy of a recommended item in a simply comparable measure or metric.

Diversity-based Recommendation Ensembles

As noted in Chapter 4, an optimal ensemble of both the nearest and further neighbor approaches could potentially increase the quality of the baseline by as much as 60% when creating an ensemble of the k-nearest neighbor and the k-furthest neighbor algorithms. However, recommendation algorithm ensembles are traditionally based on the intersections of several approaches, i.e. by taking the intersection of the recommended items from several recommendation algorithms. In the context of the furthest and nearest neighbor approaches, this strategy would most likely fail due to the orthogonality of the lists of recommendations from the two algorithms. Since there were minimal amounts of intersecting items between these lists, alternative ensemble-building methods need to be researched, preferably where the intersection of the recommended items is not a factor. Instead, these alternative ensemble methods should be able to estimate the usefulness of the list of recommended items from each algorithm in order to create one unified, diversity-oriented recommender algorithm which retains the predictive accuracy of the baseline whilst gaining the diversity of the antipode algorithm.

Context-aware Correlation of Qualitative and Quantitative Measures

Chapter 4 attempted to find whether there were correlating aspects between specific user-centric concepts, such as novelty and serendipity, and correlated these to traditional measures such as rating performance. However, as user-centric concepts most likely are affected by the users' current situations, it is likely that the correlation of perception-oriented aspects and traditional metrics will differ in different contexts, e.g. novelty needs not always be a positive aspect, similarly a high rating can have different meaning in different contexts. This research direction seems to not have been explored at all in the context of recommender systems and their evaluation.

Contextualizing the Magic Barrier

The mathematical characterization of the magic barrier presented in Chapter 5 tied the barrier to a rating prediction error, i.e. RMSE. However, as mentioned above, the context the rating was given in can affect the rating value, e.g. the mood of the user can result in a higher or lower rating than in a different context. This opens up a research direction on the topic of contextualizing the magic barrier. There is a vast amount of related work on context-aware recommender systems, however, only a small portion of it deals with context-aware evaluation

Generalization of the Magic Barrier

As mentioned above, the characterization of the magic barrier presented in Chapter 5 tied the barrier to a specific rating prediction error measure, i.e. RMSE. The magic barrier is however a concept that can be extended to cover other rating error and rating accuracy metrics (e.g. precision, recall) as well. The current formalization does however not go beyond RMSE. A potential extension, or generalization, of the magic barrier to extend to other types of metrics presents another potential avenue for future work. As the magic barrier is a rather newly researched topic, there is little prior related work in this direction, the potential impact of a general magic barrier characterization are, however, likely high.

Bridging the Gap Between User-centric Evaluation and the Magic Barrier

The conclusion of Chapter 5 mentions that as optimization and evaluation beyond the magic barrier is likely to be of little use, other means of evaluation should be employed, e.g. user-centric evaluation. It is reasonable to assume that there are correlations between the perceived quality of a recommender system and the underlaying model's magic barrier. This presents a potential research direction bridging the gap between qualitative and quantitative evaluation of recommender systems.

6.4 Closing Remarks

Recommender systems help people in a multitude of ways in their everyday lives, whether recommending a travel route, a purchase or a friend on a social network. In many of these cases the recommendations seem so natural that the user receiving the recommendation is not actively aware of the recommendation itself. If we imagine navigating the current information landscape without the assistance of recommender systems, the task quickly becomes infeasible for any single user. With the current growth of created information, recommender systems will need to become more personalized, more accurate, and more finely tuned to the user and the task at hand. This is not possible without new approaches to recommender system evaluation. Future methods for evaluating recommender systems need to take into consideration other aspects than just the rating or recommendation accuracy. These methods will however not emerge overnight, they will likely come into being over time, as the recommendation landscape changes. In the meantime, the techniques presented throughout this dissertation provide the basis for learning more about the users and use cases of recommender systems, paving the way to more elaborate, efficient and accurate recommendations.

Bibliography

[Abd07] H. Abdi. “Bonferroni and Sidak corrections for multiple comparisons”. In: *Encyclopedia of Measurement and Statistics*. Ed. by N. J. Salkind. Thousand Oaks, CA: Sage, 2007.

[AT11] Panagiotis Adamopoulos and Alexander Tuzhilin. “On Unexpectedness in Recommender Systems: Or How to Expect the Unexpected”. In: *Proceedings of the Workshop on Novelty and Diversity in Recommender Systems*. DiveRS ’11. Chicago, IL, USA, 2011, pp. 11–18.

[AK09] Gediminas Adomavicius and YoungOk Kwon. “Towards More Diverse Recommendations: Item Re-Ranking Methods for Recommender Systems”. In: *Proceedings of the 19th Workshop on Information Technologies and Systems*. WITS ’09. Phoenix, IN, USA, 2009.

[AT05] Gediminas Adomavicius and Alexander Tuzhilin. “Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions”. In: *IEEE Trans. on Knowl. and Data Eng.* 17.6 (June 2005), pp. 734–749. ISSN: 1041-4347. DOI: 10.1109/TKDE.2005.99. URL: http://dx.doi.org/10.1109/TKDE.2005.99.

[ART06] Rakesh Agrawal, Ralf Rantzau, and Evimaria Terzi. “Context-sensitive ranking”. In: *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*. SIGMOD ’06. Chicago, IL, USA: ACM, 2006, pp. 383–394. ISBN: 1-59593-434-0. DOI: 10.1145/1142473.1142517. URL: http://doi.acm.org/10.1145/1142473.1142517.

[Ama12] Xavier Amatriain. “Building industrial-scale real-world recommender systems”. In: *Proceedings of the sixth ACM conference on Recommender systems*. RecSys ’12. Dublin, Ireland: ACM, 2012, pp. 7–8. ISBN: 978-1-4503-1270-7. DOI: `10.1145/2365952.2365958`. URL: `http://doi.acm.org/10.1145/2365952.2365958`.

[AB12] Xavier Amatriain and Justin Basilico. *Netflix Recommendations: Beyond the 5 stars (Part 1) – The Netflix Tech Blog*. `http://techblog.netflix.com/2012/04/netflix-recommendations-beyond-5-stars.html` (retrieved May 12, 2012). 2012. URL: `http://techblog.netflix.com/2012/04/netflix-recommendations-beyond-5-stars.html`.

[APO09] Xavier Amatriain, Josep M. Pujol, and Nuria Oliver. “I Like It... I Like It Not: Evaluating User Ratings Noise in Recommender Systems”. In: *Proceedings of the 17th International Conference on User Modeling, Adaptation, and Personalization: formerly UM and AH*. UMAP ’09. Trento, Italy: Springer-Verlag, 2009, pp. 247–258. ISBN: 978-3-642-02246-3. DOI: `10.1007/978-3-642-02247-0_24`. URL: `http://dx.doi.org/10.1007/978-3-642-02247-0_24`.

[Ama+09] Xavier Amatriain et al. “Rate it again: increasing recommendation accuracy by user re-rating”. In: *Proceedings of the third ACM conference on Recommender systems*. RecSys ’09. New York, NY, USA: ACM, 2009, pp. 173–180. ISBN: 978-1-60558-435-5. DOI: `10.1145/1639714.1639744`. URL: `http://doi.acm.org/10.1145/1639714.1639744`.

[And04] Chris Anderson. “The Long Tail”. In: *Wired Magazine* 12.10 (2004).

[And06] Chris Anderson. *The Long Tail: Why the Future of Business Is Selling Less of More*. Hyperion, 2006. ISBN: 1401302378.

[Avé05] Yves Avérous. “Untranslateable Words - Serendipity”. In: *Journal of the Northern Californa Translators Association: Online Edition* (2005). URL: `http://translorial.com/2005/09/01/untranslatable-words-serendipity/`.

[BYRN99] Ricardo A. Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1999. ISBN: 020139829X.

[BS97] Marko Balabanović and Yoav Shoham. “Fab: content-based, collaborative recommendation”. In: *Commun. ACM* 40.3 (Mar. 1997), pp. 66–72. ISSN: 0001-0782. DOI: `10.1145/245108.245124`. URL: `http://doi.acm.org/10.1145/245108.245124`.

[BR07] Linas Baltrunas and Francesco Ricci. "Dynamic item weighting and selection for collaborative filtering". In: *Proceedings of the ECML-PKDD 2007 Workshop on Web mining 2.0.* 2007. URL: http://www.inf.unibz.it/~ricci/papers/ecml2007.pdf.

[BB98] W. Barber and A. Badre. "Culturability: The Merging of Culture and Usability". In: *4th Conference on Human Factors & the Web.* 4th Conference on Human Factors & the Web, Baskin Ridge NJ, USA, 1998. URL: http://research.microsoft.com/en-us/um/people/marycz/hfweb98/barber/.

[BHC98] Chumki Basu, Haym Hirsh, and William Cohen. "Recommendation as classification: using social and content-based information in recommendation". In: *Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence.* AAAI '98/IAAI '98. Madison, Wisconsin, United States: American Association for Artificial Intelligence, 1998, pp. 714–720. ISBN: 0-262-51098-7. URL: http://dl.acm.org/citation.cfm?id=295240.295795.

[BL07] James Bennett and Stan Lanning. "The Netflix Prize". In: *In Proceedings of the 2007 KDD Cup and Workshop in conjunction with KDD.* New York, NY, USA: ACM, 2007.

[Ben+07] James Bennett et al. "KDD Cup and workshop 2007". In: *SIGKDD Explor. Newsl.* 9.2 (Dec. 2007), pp. 51–52. ISSN: 1931-0145. DOI: 10.1145/1345448.1345459. URL: http://doi.acm.org/10.1145/1345448.1345459.

[BL] T. Berners-Lee. *The WorldWideWeb browser.* http://www.w3.org/People/Berners-Lee/WorldWideWeb - March, 2010. URL: http://www.w3.org/People/Berners-Lee/WorldWideWeb.

[BNJ03] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. "Latent dirichlet allocation". In: *J. Mach. Learn. Res.* 3 (Mar. 2003), pp. 993–1022. ISSN: 1532-4435. URL: http://dl.acm.org/citation.cfm?id=944919.944937.

[Bol+10] Dirk Bollen et al. "Understanding choice overload in recommender systems". In: *Proceedings of the fourth ACM conference on Recommender systems.* RecSys '10. New York, NY, USA: ACM, 2010, pp. 63–70. ISBN: 978-1-60558-906-0. DOI: 10.1145/1864708.1864724. URL: http://doi.acm.org/10.1145/1864708.1864724.

[BMS11] Steven Bourke, Kevin McCarthy, and Barry Smyth. "Power to the people: exploring neighbourhood formations in social recommender system". In: *Proceedings of the fifth ACM conference on Recommender systems.* RecSys '11. New York, NY, USA: ACM, 2011, pp. 337–340. ISBN: 978-1-4503-0683-6. DOI: 10.1145/2043932.2043997. URL: http://doi.acm.org/10.1145/2043932.2043997.

[BHK98] John S. Breese, David Heckerman, and Carl Kadie. “Empirical analysis of predictive algorithms for collaborative filtering”. In: *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*. UAI’98. Madison, Wisconsin: Morgan Kaufmann Publishers Inc., 1998, pp. 43–52. ISBN: 1-55860-555-X. URL: http://dl.acm.org/citation.cfm?id=2074094.2074100.

[BP98] Sergey Brin and Lawrence Page. “The anatomy of a large-scale hypertextual Web search engine”. In: *Comput. Netw. ISDN Syst.* 30.1-7 (Apr. 1998), pp. 107–117. ISSN: 0169-7552. DOI: 10.1016/S0169-7552(98)00110-X. URL: http://dx.doi.org/10.1016/S0169-7552(98)00110-X.

[BHS10] Erik Brynjolfsson, Yu (Jeffrey) Hu, and Michael D. Smith. “Long Tails vs. Superstars: The Effect of Information Technology on Product Variety and Sales Concentration Patterns”. In: *Information Systems Research* 21.4 (2010), pp. 736–747. DOI: 10.1287/isre.1100.0325. eprint: http://isr.journal.informs.org/content/21/4/736.full.pdf+html. URL: http://isr.journal.informs.org/content/21/4/736.abstract.

[CVW11] P. Castells, S. Vargas, and J. Wang. “Novelty and Diversity Metrics for Recommender Systems: Choice, Discovery and Relevance”. In: *International Workshop on Diversity in Document Retrieval (DDR 2011) at the 33rd European Conference on Information Retrieval (ECIR 2011)*. Dublin, Ireland, 2011. URL: http://ir.ii.uam.es/rim3/publications/ddr11.pdf.

[Cel10a] Ò. Celma. *Last.fm dataset 360K.* (retrieved June, 2012). 2010. URL: http://mtg.upf.edu/node/1671.

[Cel10b] Ò. Celma. *Music Recommendation and Discovery - The Long Tail, Long Fail, and Long Play in the Digital Music Space.* Springer, 2010. ISBN: 978-3-642-13286-5.

[Cel08] Ò. Celma. “Music Recommendation and Discovery in the Long Tail”. PhD thesis. Barcelona: Universitat Pompeu Fabra, 2008.

[CC08] Ò. Celma and P. Cano. “From hits to niches? or how popular artists can bias music recommendation and discovery”. In: *2nd Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition (ACM KDD)*. Las Vegas, USA, 2008. URL: http://mtg.upf.edu/files/publications/Celma-ACM-Netflix-KDD2008.pdf.

[CH08] Òscar Celma and Perfecto Herrera. “A new approach to evaluating novel recommendations”. In: *Proceedings of the 2008 ACM conference on Recommender systems*. RecSys ’08. Lausanne, Switzerland: ACM, 2008, pp. 179–186. ISBN: 978-1-60558-093-7. DOI: 10.1145/1454008.1454038. URL: http://doi.acm.org/10.1145/1454008.1454038.

[CL08] Òscar Celma and Paul Lamere. "If you like the beatles you might like...: a tutorial on music recommendation". In: *Proceedings of the 16th ACM international conference on Multimedia*. MM '08. Vancouver, British Columbia, Canada: ACM, 2008, pp. 1157–1158. ISBN: 978-1-60558-303-7. DOI: 10.1145/1459359.1459615. URL: http://doi.acm.org/10.1145/1459359.1459615.

[CL11] Oscar Celma and Paul Lamere. "Music recommendation and discovery revisited". In: *Proceedings of the fifth ACM conference on Recommender systems*. RecSys '11. Chicago, Illinois, USA: ACM, 2011, pp. 7–8. ISBN: 978-1-4503-0683-6. DOI: 10.1145/2043932.2043936. URL: http://doi.acm.org/10.1145/2043932.2043936.

[CP08] Li Chen and Pearl Pu. "A cross-cultural user evaluation of product recommender interfaces". In: *Proceedings of the 2008 ACM conference on Recommender systems*. RecSys '08. Lausanne, Switzerland: ACM, 2008, pp. 75–82. ISBN: 978-1-60558-093-7. DOI: 10.1145/1454008.1454022. URL: http://doi.acm.org/10.1145/1454008.1454022.

[Cop09] Michael V. Copeland. *Box office boffo for brainiacs: The Netflix Prize*. http://tech.fortune.cnn.com/2009/09/21/box-office-boffo-for-brainiacs-the-netflix-prize/, (retrieved March, 2012). 2009. URL: http://tech.fortune.cnn.com/2009/09/21/box-office-boffo-for-brainiacs-the-netflix-prize/.

[Cos+03] Dan Cosley et al. "Is seeing believing?: how recommender system interfaces affect users' opinions". In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. CHI '03. New York, NY, USA: ACM, 2003, pp. 585–592. ISBN: 1-58113-630-7. DOI: 10.1145/642611.642713. URL: http://doi.acm.org/10.1145/642611.642713.

[CR11] Alberto Costa and Fabio Roda. "Recommender systems by means of information retrieval". In: *Proceedings of the International Conference on Web Intelligence, Mining and Semantics*. WIMS '11. Sogndal, Norway: ACM, 2011, 57:1–57:5. ISBN: 978-1-4503-0148-0. DOI: 10.1145/1988688.1988755. URL: http://doi.acm.org/10.1145/1988688.1988755.

[CH67] T. Cover and P. Hart. "Nearest neighbor pattern classification". In: *Information Theory, IEEE Transactions on* 13.1 (1967), pp. 21 –27. ISSN: 0018-9448. DOI: 10.1109/TIT.1967.1053964.

[CKT10] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. "Performance of recommender algorithms on top-n recommendation tasks". In: *Proceedings of the fourth ACM conference on Recommender systems*. RecSys '10. Barcelona, Spain: ACM, 2010, pp. 39–46. ISBN:

978-1-60558-906-0. DOI: 10.1145/1864708.1864721. URL: http://doi.acm.org/10.1145/1864708.1864721.

[Cre+11a] Paolo Cremonesi et al. "Comparative evaluation of recommender system quality". In: *Proceedings of the 2011 annual conference extended abstracts on Human factors in computing systems*. CHI EA '11. New York, NY, USA: ACM, 2011, pp. 1927–1932. ISBN: 978-1-4503-0268-5. DOI: 10.1145/1979742.1979896. URL: http://doi.acm.org/10.1145/1979742.1979896.

[Cre+11b] Paolo Cremonesi et al. "Looking for "good" recommendations: a comparative evaluation of recommender systems". In: *Proceedings of the 13th IFIP TC 13 international conference on Human-computer interaction - Volume Part III*. INTERACT'11. Lisbon, Portugal: Springer-Verlag, 2011, pp. 152–168. ISBN: 978-3-642-23764-5. URL: http://dl.acm.org/citation.cfm?id=2042182.2042197.

[Czi89] Gary A Cziko. "Unpredictability and Indeterminism in Human Behavior: Arguments and Implications for Educational Research". In: *Educational Researcher* 18.3 (1989), pp. 17–25. DOI: 10.3102/0013189X018003017. eprint: http://edr.sagepub.com/content/18/3/17.full.pdf+html. URL: http://edr.sagepub.com/content/18/3/17.abstract.

[DK90] J. Demmel and W. Kahan. "Accurate Singular Values of Bidiagonal Matrices". In: *SIAM Journal on Scientific and Statistical Computing* 11.5 (1990), pp. 873–912. DOI: 10.1137/0911052. eprint: http://epubs.siam.org/doi/pdf/10.1137/0911052. URL: http://epubs.siam.org/doi/abs/10.1137/0911052.

[DSM10] Maunendra Sankar Desarkar, Sudeshna Sarkar, and Pabitra Mitra. "Aggregating preference graphs for collaborative rating prediction". In: *Proceedings of the fourth ACM conference on Recommender systems*. RecSys '10. Barcelona, Spain: ACM, 2010, pp. 21–28. ISBN: 978-1-60558-906-0. DOI: 10.1145/1864708.1864716. URL: http://doi.acm.org/10.1145/1864708.1864716.

[DSS12] Maunendra Sankar Desarkar, Roopam Saxena, and Sudeshna Sarkar. "Preference relation based matrix factorization for recommender systems". In: *Proceedings of the 20th international conference on User Modeling, Adaptation, and Personalization*. UMAP'12. Montreal, Canada: Springer-Verlag, 2012, pp. 63–75. ISBN: 978-3-642-31453-7. DOI: 10.1007/978-3-642-31454-4_6. URL: http://dx.doi.org/10.1007/978-3-642-31454-4_6.

[DK11] Christian Desrosiers and George Karypis. "A Comprehensive Survey of Neighborhood-based Recommendation Methods." In: *Recommender Systems Handbook*. Ed. by Francesco Ricci et al. Springer, 2011, pp. 107–144. ISBN: 978-0-387-85819-7. URL: http://dblp.uni-trier.de/db/reference/rsh/rsh2011.html#DesrosiersK11.

[Fil] Filmtipset. *Filmtipset.se i ett nötskal.* http://nyheter24.se/filmtipset/tailpages.cgi?page=Om_filmtipset (retrieved June, 2012). URL: http://nyheter24.se/filmtipset/tailpages.cgi?page=Om_filmtipset.

[Fis91] B. Fischhoff. "Value elicitation: Is there anything there?" In: *American psychologist* 46 (1991), pp. 835–847.

[FH07] Daniel M. Fleder and Kartik Hosanagar. "Recommender systems and their impact on sales diversity". In: *Proceedings of the 8th ACM conference on Electronic commerce.* EC '07. San Diego, California, USA: ACM, 2007, pp. 192–199. ISBN: 978-1-59593-653-0. DOI: 10.1145/1250910.1250939. URL: http://doi.acm.org/10.1145/1250910.1250939.

[GAD98] N. Glance, D. Arregui, and M. Dardenne. "Knowledge Pump: Supporting the Flow and Use of Knowledge." In: *Information Technology for Knowledge Management.* Ed. by U. Borghoff and R. Pareschi. Springer Verlag, 1998.

[Gla+01] Natalie Glance et al. "Collaborative document monitoring". In: *Proceedings of the 2001 International ACM SIGGROUP Conference on Supporting Group Work.* GROUP '01. Boulder, Colorado, USA: ACM, 2001, pp. 171–178. ISBN: 1-58113-294-8. DOI: 10.1145/500286.500313. URL: http://doi.acm.org/10.1145/500286.500313.

[Goe+10] Sharad Goel et al. "Anatomy of the long tail: ordinary people with extraordinary tastes". In: *Proceedings of the third ACM international conference on Web search and data mining.* WSDM '10. New York, New York, USA: ACM, 2010, pp. 201–210. ISBN: 978-1-60558-889-6. DOI: 10.1145/1718487.1718513. URL: http://doi.acm.org/10.1145/1718487.1718513.

[Goh+12] Dion Goh et al. "Investigating user perceptions of engagement and information quality in mobile human computation games". In: *Proceedings of the 12th ACM/IEEE-CS joint conference on Digital Libraries.* JCDL '12. Washington, DC, USA: ACM, 2012, pp. 391–392. ISBN: 978-1-4503-1154-0. DOI: 10.1145/2232817.2232906. URL: http://doi.acm.org/10.1145/2232817.2232906.

[GH06] *FilmTrust: movie recommendations using trust in web-based social networks.* Vol. 1. 2006, pp. 282–286. URL: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1593032.

[Gol+92] David Goldberg et al. “Using collaborative filtering to weave an information tapestry”. In: *Commun. ACM* 35.12 (Dec. 1992), pp. 61–70. ISSN: 0001-0782. DOI: 10.1145/138859.138867. URL: http://doi.acm.org/10.1145/138859.138867.

[Goo+99] Nathaniel Good et al. “Combining collaborative filtering with personal agents for better recommendations”. In: *Proceedings of the sixteenth national conference on Artificial intelligence and the eleventh Innovative applications of artificial intelligence conference innovative applications of artificial intelligence.* AAAI ’99/IAAI ’99. Orlando, Florida, United States: American Association for Artificial Intelligence, 1999, pp. 439–446. ISBN: 0-262-51106-1. URL: http://dl.acm.org/citation.cfm?id=315149.315352.

[HKR02] Jon Herlocker, Joseph A. Konstan, and John Riedl. “An Empirical Analysis of Design Choices in Neighborhood-Based Collaborative Filtering Algorithms”. In: *Inf. Retr.* 5.4 (Oct. 2002), pp. 287–310. ISSN: 1386-4564. DOI: 10.1023/A:1020443909834. URL: http://dx.doi.org/10.1023/A:1020443909834.

[Her+99] Jonathan L. Herlocker et al. “An algorithmic framework for performing collaborative filtering”. In: *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval.* SIGIR ’99. Berkeley, California, United States: ACM, 1999, pp. 230–237. ISBN: 1-58113-096-1. DOI: 10.1145/312624.312682. URL: http://doi.acm.org/10.1145/312624.312682.

[Her+04] Jonathan L. Herlocker et al. “Evaluating collaborative filtering recommender systems”. In: *ACM Trans. Inf. Syst.* 22.1 (Jan. 2004), pp. 5–53. ISSN: 1046-8188. DOI: 10.1145/963770.963772. URL: http://doi.acm.org/10.1145/963770.963772.

[Hil+95] Will Hill et al. “Recommending and evaluating choices in a virtual community of use”. In: *Proceedings of the SIGCHI conference on Human factors in computing systems.* CHI ’95. Denver, Colorado, United States: ACM Press/Addison-Wesley Publishing Co., 1995, pp. 194–201. ISBN: 0-201-84705-1. DOI: 10.1145/223904.223929. URL: http://dx.doi.org/10.1145/223904.223929.

[HP11] Rong Hu and Pearl Pu. “Enhancing recommendation diversity with organization interfaces”. In: *Proceedings of the 16th international conference on Intelligent user interfaces.* IUI ’11. Palo Alto, CA, USA: ACM, 2011, pp. 347–350. ISBN: 978-1-4503-0419-1. DOI: 10.1145/1943403.1943462. URL: http://doi.acm.org/10.1145/1943403.1943462.

[JW10] Tamas Jambor and Jun Wang. "Optimizing multiple objectives in collaborative filtering". In: *Proceedings of the fourth ACM conference on Recommender systems.* RecSys '10. Barcelona, Spain: ACM, 2010, pp. 55–62. ISBN: 978-1-60558-906-0. DOI: 10.1145/1864708.1864723. URL: http://doi.acm.org/10.1145/1864708.1864723.

[JCS04] Rong Jin, Joyce Y. Chai, and Luo Si. "An automatic weighting scheme for collaborative filtering". In: *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval.* SIGIR '04. Sheffield, United Kingdom: ACM, 2004, pp. 337–344. ISBN: 1-58113-881-4. DOI: 10.1145/1008992.1009051. URL: http://doi.acm.org/10.1145/1008992.1009051.

[Klu+12] Daniel Kluver et al. "How many bits per rating?" In: *Proceedings of the sixth ACM conference on Recommender systems.* RecSys '12. Dublin, Ireland: ACM, 2012, pp. 99–106. ISBN: 978-1-4503-1270-7. DOI: 10.1145/2365952.2365974. URL: http://doi.acm.org/10.1145/2365952.2365974.

[Kni12] Bart P. Knijnenburg. "Conducting user experiments in recommender systems". In: *Proceedings of the sixth ACM conference on Recommender systems.* RecSys '12. Dublin, Ireland: ACM, 2012, pp. 3–4. ISBN: 978-1-4503-1270-7. DOI: 10.1145/2365952.2365956. URL: http://doi.acm.org/10.1145/2365952.2365956.

[KSTB10] Bart P. Knijnenburg, Lars Schmidt-Thieme, and Dirk G.F.M. Bollen. "Workshop on user-centric evaluation of recommender systems and their interfaces". In: *Proceedings of the fourth ACM conference on Recommender systems.* RecSys '10. Barcelona, Spain: ACM, 2010, pp. 383–384. ISBN: 978-1-60558-906-0. DOI: 10.1145/1864708.1864800. URL: http://doi.acm.org/10.1145/1864708.1864800.

[KWK11] Bart P. Knijnenburg, Martijn C. Willemsen, and Alfred Kobsa. "A pragmatic procedure to support the user-centric evaluation of recommender systems". In: *Proceedings of the fifth ACM conference on Recommender systems.* RecSys '11. Chicago, Illinois, USA: ACM, 2011, pp. 321–324. ISBN: 978-1-4503-0683-6. DOI: 10.1145/2043932.2043993. URL: http://doi.acm.org/10.1145/2043932.2043993.

[Kni+12] B.P. Knijnenburg et al. "Explaining the user experience of recommender systems". In: *User Modeling and User-Adapted Interaction* 22 (2012), pp. 441–504.

[KDK11] Noam Koenigstein, Gideon Dror, and Yehuda Koren. “Yahoo! music recommendations: modeling music ratings with temporal dynamics and item taxonomy”. In: *Proceedings of the fifth ACM conference on Recommender systems.* RecSys ’11. Chicago, Illinois, USA: ACM, 2011, pp. 165–172. ISBN: 978-1-4503-0683-6. DOI: 10.1145/2043932.2043964. URL: http://doi.acm.org/10.1145/2043932.2043964.

[Koh12] Ron Kohavi. “Online controlled experiments: introduction, learnings, and humbling statistics”. In: *Proceedings of the sixth ACM conference on Recommender systems.* RecSys ’12. Dublin, Ireland: ACM, 2012, pp. 1–2. ISBN: 978-1-4503-1270-7. DOI: 10.1145/2365952.2365954. URL: http://doi.acm.org/10.1145/2365952.2365954.

[Kwo08] YoungOk Kwon. “Improving top-n recommendation techniques using rating variance”. In: *Proceedings of the 2008 ACM conference on Recommender systems.* RecSys ’08. Lausanne, Switzerland: ACM, 2008, pp. 307–310. ISBN: 978-1-60558-093-7. DOI: 10.1145/1454008.1454059. URL: http://doi.acm.org/10.1145/1454008.1454059.

[Lai06] Jung-Yu Lai. “Assessment of employees’ perceptions of service quality and satisfaction with e-business”. In: *Proceedings of the 2006 ACM SIGMIS CPR conference on computer personnel research: Forty four years of computer personnel research: achievements, challenges & the future.* SIGMIS CPR ’06. Claremont, California, USA: ACM, 2006, pp. 236–243. ISBN: 1-59593-349-2. DOI: 10.1145/1125170.1125228. URL: http://doi.acm.org/10.1145/1125170.1125228.

[LR04] Shyong K. Lam and John Riedl. “Shilling recommender systems for fun and profit”. In: *Proceedings of the 13th international conference on World Wide Web.* WWW ’04. New York, NY, USA: ACM, 2004, pp. 393–402. ISBN: 1-58113-844-X. DOI: 10.1145/988672.988726. URL: http://doi.acm.org/10.1145/988672.988726.

[LFR06] Shyong K. “Tony” Lam, Dan Frankowski, and John Riedl. “Do you trust your recommendations? an exploration of security and privacy issues in recommender systems”. In: *Proceedings of the 2006 international conference on Emerging Trends in Information and Communication Security.* ETRICS’06. Freiburg, Germany: Springer-Verlag, 2006, pp. 14–29. ISBN: 3-540-34640-6, 978-3-540-34640-1. DOI: 10.1007/11766155_2. URL: http://dx.doi.org/10.1007/11766155_2.

[Lat+10] Neal Lathia et al. “Temporal diversity in recommender systems”. In: *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval.* SIGIR ’10. Geneva, Switzerland: ACM, 2010, pp. 210–217. ISBN: 978-1-4503-

0153-4. DOI: 10.1145/1835449.1835486. URL: http://doi.acm.org/10.1145/1835449.1835486.

[LL11] Kibeom Lee and Kyogu Lee. "My head is your tail: applying link analysis on long-tailed music listening behavior for music recommendation". In: *Proceedings of the fifth ACM conference on Recommender systems*. RecSys '11. Chicago, Illinois, USA: ACM, 2011, pp. 213–220. ISBN: 978-1-4503-0683-6. DOI: 10.1145/2043932.2043971. URL: http://doi.acm.org/10.1145/2043932.2043971.

[LR07] Juha Leino and Kari-Jouko Räihä. "Case amazon: ratings and reviews as part of recommendations". In: *Proceedings of the 2007 ACM conference on Recommender systems*. RecSys '07. Minneapolis, MN, USA: ACM, 2007, pp. 137–140. ISBN: 978-1-59593-730–8. DOI: 10.1145/1297231.1297255. URL: http://doi.acm.org/10.1145/1297231.1297255.

[LB11] Mark Levy and Klaas Bosteels. "Music Recommendation and the Long Tail". In: *Proceedings of the Workshop on Music Recommendation and Discovery 2011*. WOMRAD '11. Chicago, IL, USA: CEUR-WS Vol. 793, 2011.

[Lin06] Greg Linden. *Geeking with Greg :: Early Amazon: Similarities*. http://glinden.blogspot.com/2006/03/early-amazon-similarities.html, (retrieved March, 2012). 2006. URL: http://glinden.blogspot.de/2006/03/early-amazon-similarities.html.

[Lin+12] Guang Ling et al. "Response Aware Model-Based Collaborative Filtering". In: *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence*. UAI. Catalina Island, CA, USA, 2012.

[Lud] Michael Ludwig. *MovieLens Data Sets*. http://www.grouplens.org/node/73, (retrieved June, 2012). URL: http://www.grouplens.org/node/73.

[Luo+08] Heng Luo et al. "A collaborative filtering framework based on both local user similarity and global user similarity". In: *Mach. Learn.* 72.3 (Sept. 2008), pp. 231–245. ISSN: 0885-6125. DOI: 10.1007/s10994-008-5068-4. URL: http://dx.doi.org/10.1007/s10994-008-5068-4.

[Lyc] Lycos. *Lycos - Company Overview*. http://info.lycos.com/about/company-overview (retrieved November, 2012). URL: http://info.lycos.com/about/company-overview.

[MZ09] Benjamin M. Marlin and Richard S. Zemel. "Collaborative prediction and ranking with non-random missing data". In: *Proceedings of the third ACM conference on Recommender systems.* RecSys '09. New York, New York, USA: ACM, 2009, pp. 5–12. ISBN: 978-1-60558-435-5. DOI: 10.1145/1639714.1639717. URL: http://doi.acm.org/10.1145/1639714.1639717.

[McC11] Brad McCarty. *Last.fm: 50 billion scrobbles and the return of the mix tape.* http://thenextweb.com/media/2011/04/16/last-fm-50-billion-scrobbles-and-the-return-of-the-mix-tape/3/ (retrieved June, 2012). 2011. URL: http://thenextweb.com/media/2011/04/16/last-fm-50-billion-scrobbles-and-the-return-of-the-mix-tape/3/.

[MRK06] Sean M. McNee, John Riedl, and Joseph A. Konstan. "Being accurate is not enough: how accuracy metrics have hurt recommender systems". In: *CHI '06 extended abstracts on Human factors in computing systems.* CHI EA '06. Montreal, Quebec, Canada: ACM, 2006, pp. 1097–1101. ISBN: 1-59593-298-4. DOI: 10.1145/1125451.1125659. URL: http://doi.acm.org/10.1145/1125451.1125659.

[McN+02] Sean M. McNee et al. "On the recommending of citations for research papers". In: *Proceedings of the 2002 ACM conference on Computer supported cooperative work.* CSCW '02. New Orleans, Louisiana, USA: ACM, 2002, pp. 116–125. ISBN: 1-58113-560-2. DOI: 10.1145/587078.587096. URL: http://doi.acm.org/10.1145/587078.587096.

[Mov] MovieLens. *MovieLens 10M/100k Data Set README.* http://www.grouplens.org/system/files/ml-10m-README.html (retrieved June, 2012). URL: http://www.grouplens.org/system/files/ml-10m-README.html.

[Nat] National Institute of Standard and Technology. http://trec.nist.gov/overview.html (retrieved June, 2012). URL: http://trec.nist.gov/overview.html.

[Nee] C. Needham. *IMDb History.* http://www.imdb.com/help/show_leaf?history - December, 2009. URL: http://www.imdb.com/help/show_leaf?history.

[Net06] Netflix Prize. *The Netflix Prize Rules.* http://www.netflixprize.com//rules (retrieved June, 2012). 2006. URL: http://www.netflixprize.com//rules.

[OZ51] Julio Ojeda-Zapata. "New Site Personalizes Movie Reviews". In: *St. Paul Pioneer Press* (9/15/1997).

[OHS06] Michael P. O'Mahony, Neil J. Hurley, and Guénolé C.M. Silvestre. "Detecting noise in recommender system databases". In: *Proceedings of the 11th international conference on Intelligent user interfaces.* IUI '06. Sydney, Australia: ACM, 2006, pp. 109–115. ISBN: 1-59593-287-9. DOI: 10.1145/1111449.1111477. URL: http://doi.acm.org/10.1145/1111449.1111477.

[Pag+99] Lawrence Page et al. *The PageRank Citation Ranking: Bringing Order to the Web.* Technical Report 1999-66. Previous number = SIDL-WP-1999-0120. Stanford InfoLab, 1999. URL: http://ilpubs.stanford.edu:8090/422/.

[PT08] Yoon-Joo Park and Alexander Tuzhilin. "The long tail of recommender systems and how to leverage it". In: *Proceedings of the 2008 ACM conference on Recommender systems.* RecSys '08. Lausanne, Switzerland: ACM, 2008, pp. 11–18. ISBN: 978-1-60558-093-7. DOI: 10.1145/1454008.1454012. URL: http://doi.acm.org/10.1145/1454008.1454012.

[PT09] István Pilászy and Domonkos Tikk. "Recommending new movies: even a few ratings are more valuable than metadata". In: *Proceedings of the third ACM conference on Recommender systems.* RecSys '09. New York, New York, USA: ACM, 2009, pp. 93–100. ISBN: 978-1-60558-435-5. DOI: 10.1145/1639714.1639731. URL: http://doi.acm.org/10.1145/1639714.1639731.

[Pli12] Plista. *The Contest: Plista Contest.* http://contest.plista.com/content/the_contest (retrieved June, 2012). 2012. URL: http://contest.plista.com/content/the_contest.

[PCH11] Pearl Pu, Li Chen, and Rong Hu. "A user-centric evaluation framework for recommender systems". In: *Proceedings of the fifth ACM conference on Recommender systems.* RecSys '11. Chicago, Illinois, USA: ACM, 2011, pp. 157–164. ISBN: 978-1-4503-0683-6. DOI: 10.1145/2043932.2043962. URL: http://doi.acm.org/10.1145/2043932.2043962.

[Pu+11] Pearl Pu et al. "Usability Guidelines for Product Recommenders Based on Example Critiquing Research". In: *Recommender Systems Handbook.* Ed. by Francesco Ricci et al. 10.1007/978-0-387-85820-3_16. Springer US, 2011, pp. 511–545. ISBN: 978-0-387-85820-3. URL: http://dx.doi.org/10.1007/978-0-387-85820-3_16.

[Raf+09] Rachael Rafter et al. "What Have the Neighbours Ever Done for Us? A Collaborative Filtering Perspective". In: *Proceedings of the 17th International Conference on User Modeling, Adaptation, and Personalization: formerly UM and AH.* UMAP '09. Trento, Italy: Springer-Verlag, 2009, pp. 355–360. ISBN: 978-3-642-02246-3. DOI:

10.1007/978-3-642-02247-0_36. URL: http://dx.doi.org/10.1007/978-3-642-02247-0_36.

[Raj09] Shiva Rajaraman. *YouTube Blog: Five Stars Dominate Ratings.* http://youtube-global.blogspot.de/2009/09/five-stars-dominate-ratings.html, (retrieved March, 2010). 2009. URL: http://youtube-global.blogspot.de/2009/09/five-stars-dominate-ratings.html.

[RV97] Paul Resnick and Hal R. Varian. "Recommender systems". In: *Commun. ACM* 40.3 (Mar. 1997), pp. 56–58. ISSN: 0001-0782. DOI: 10.1145/245108.245121. URL: http://doi.acm.org/10.1145/245108.245121.

[Res+94] Paul Resnick et al. "GroupLens: an open architecture for collaborative filtering of netnews". In: *Proceedings of the 1994 ACM conference on Computer supported cooperative work.* CSCW '94. Chapel Hill, North Carolina, United States: ACM, 1994, pp. 175–186. ISBN: 0-89791-689-1. DOI: 10.1145/192844.192905. URL: http://doi.acm.org/10.1145/192844.192905.

[RRS11] Francesco Ricci, Lior Rokach, and Bracha Shapira. "Introduction to Recommender Systems Handbook". In: *Recommender Systems Handbook.* Springer, 2011, pp. 1–35.

[Ric+11] Francesco Ricci et al., eds. *Recommender Systems Handbook.* Springer, 2011. ISBN: 978-0-387-85819-7.

[Sai10] Alan Said. "Identifying and utilizing contextual data in hybrid recommender systems". In: *Proceedings of the fourth ACM conference on Recommender systems.* RecSys '10. Barcelona, Spain: ACM, 2010, pp. 365–368. ISBN: 978-1-60558-906-0. DOI: 10.1145/1864708.1864792. URL: http://doi.acm.org/10.1145/1864708.1864792.

[SBDL11] Alan Said, Shlomo Berkovsky, and Ernesto W. De Luca. "Group Recommendation in Context". In: *Proceedings of the 2nd Challenge on Context-Aware Movie Recommendation.* CAMRa '11. Chicago, Illinois: ACM, 2011, pp. 2–4. ISBN: 978-1-4503-0825-0. DOI: 10.1145/2096112.2096113. URL: http://doi.acm.org/10.1145/2096112.2096113.

[SBDL13] Alan Said, Shlomo Berkovsky, and Ernesto W. De Luca. "Introduction to special section on CAMRa2010: Movie recommendation in context". In: *ACM Trans. Intell. Syst. Technol.* 4.1 (Feb. 2013), 13:1–13:9. ISSN: 2157-6904. DOI: 10.1145/2414425.2414438. URL: http://doi.acm.org/10.1145/2414425.2414438.

[SBDL10] Alan Said, Shlomo Berkovsky, and Ernesto W. De Luca. "Putting things in context: Challenge on Context-Aware Movie Recommendation". In: *Proceedings of the Workshop on Context-Aware Movie Recommendation.* CAMRa '10. Barcelona, Spain: ACM, 2010, pp. 2–6. ISBN: 978-1-4503-0258-6. DOI: `10.1145/1869652.1869665`. URL: `http://doi.acm.org/10.1145/1869652.1869665`.

[SDA10] Alan Said, Ernesto W. De Luca, and Sahin Albayrak. "How Social Relationships Affect User Similarities". In: *Proceedings of the IUI'10 Workshop on Social Recommender Systems (SRS'10).* SRS'10. Hong Kong, China, 2010.

[SFJ13] Alan Said, Benjamin Fields, and Brijnesh J. Jain. "User-Centric Evaluation of a K-Furthest Neighbor Collaborative Filtering Recommender Algorithm". In: *Proceedings of the ACM 2013 conference on Computer Supported Cooperative Work.* San Antonio, Texas, USA: ACM, 2013.

[SJA13] Alan Said, Brijnesh J. Jain, and Sahin Albayrak. "A 3D Approach to Recommender System Evaluation". In: *Proceedings of the ACM 2013 conference on Computer Supported Cooperative Work.* San Antonio, Texas, USA: ACM, 2013.

[SJA12] Alan Said, Brijnesh J. Jain, and Sahin Albayrak. "Analyzing weighting schemes in collaborative filtering: cold start, post cold start and power users". In: *Proceedings of the 27th Annual ACM Symposium on Applied Computing.* SAC '12. Trento, Italy: ACM, 2012, pp. 2035–2040. ISBN: 978-1-4503-0857-1. DOI: `10.1145/2245276.2232114`. URL: `http://doi.acm.org/10.1145/2245276.2232114`.

[SLA11a] Alan Said, Ernesto W. De Luca, and Sahin Albayrak. "Inferring Contextual User Profiles – Improving Recommender Performance". In: *Proceedings of the 3rd RecSys Workshop on Context-Aware Recommender Systems.* CARS '11. Chicago, IL, USA: CEUR-WS Vol. 791, 2011.

[SLA11b] Alan Said, Ernesto William De Luca, and Sahin Albayrak. "Using Social and Pseudo Social Networks for Improved Recommendation Quality". In: *Proceedings of the 9th Workshop on Intelligent Techniques for Web Personalization and Recommender Systems.* ITWP '11. Barcelona, Spain: CEUR-WS Vol. 756, 2011, pp. 45–48. URL: `http://ceur-ws.org/Vol-756/paper06.pdf`.

[STH12] Alan Said, Domonkos Tikk, and Andreas Hotho. "The challenge of recommender systems challenges". In: *Proceedings of the sixth ACM conference on Recommender systems.* RecSys '12. Dublin, Ireland: ACM, 2012, pp. 9–10. ISBN: 978-1-4503-1270-7. DOI: `10.1145/2365952.2365959`. URL: `http://doi.acm.org/10.1145/2365952.2365959`.

[Sai+11] Alan Said et al. "A Comparison of How Demographic Data Affects Recommendation". In: *Adjoint Proceedings of the 19th International Conference on User Modeling, Adaptation, and Personalization: formerly UM and AH.* UMAP '11. Girona, Spain, 2011.

[Sai+09] Alan Said et al. "A Hybrid PLSA Approach for Warmer Cold Start in Folksonomy Recommendation". In: *Proceedings of the RecSys'09 Workshop on Recommender Systems & The Social Web.* RSWeb '09. New York, NY, USA: CEUR-WS Vol. 532, 2009, pp. 87–90.

[Sai+12a] Alan Said et al. "Correlating perception-oriented aspects in user-centric recommender system evaluation". In: *Proceedings of the 4th Information Interaction in Context Symposium.* IIIX '12. Nijmegen, The Netherlands: ACM, 2012, pp. 294–297. ISBN: 978-1-4503-1282-0. DOI: 10.1145/2362724.2362778. URL: http://doi.acm.org/10.1145/2362724.2362778.

[Sai+12b] Alan Said et al. "Estimating the magic barrier of recommender systems: a user study". In: *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval.* SIGIR '12. Portland, Oregon, USA: ACM, 2012, pp. 1061–1062. ISBN: 978-1-4503-1472-5. DOI: 10.1145/2348283.2348469. URL: http://doi.acm.org/10.1145/2348283.2348469.

[Sai+10] Alan Said et al. "Exploiting Hierarchical Tags for Context-Awareness". In: *Proceedings of the third workshop on Exploiting semantic annotations in information retrieval.* ESAIR '10. Toronto, ON, Canada: ACM, 2010, pp. 35–36. ISBN: 978-1-4503-0372-9. DOI: 10.1145/1871962.1871984. URL: http://doi.acm.org/10.1145/1871962.1871984.

[Sai+12c] Alan Said et al. "Increasing Diversity Through Furthest Neighbor-Based Recommendation". In: *Proceedings of the WSDM'12 Workshop on Diversity in Document Retrieval (DDR'12).* Seattle, WA, USA, 2012.

[Sai+12d] Alan Said et al. "KMulE: A Framework for User-based Comparison of Recommender Algorithms". In: *Proceedings of the 2012 ACM international conference on Intelligent User Interfaces.* IUI '12. Lisbon, Portugal: ACM, 2012, pp. 323–324. ISBN: 978-1-4503-1048-2. DOI: 10.1145/2166966.2167034. URL: http://doi.acm.org/10.1145/2166966.2167034.

[Sai+12e] Alan Said et al. "Recommender Systems Evaluation: A 3D Benchmark". In: *Proceedings of the Workshop on Recommendation Utility Evaluation: Beyond RMSE (RUE 2012).* RUE'12. Dublin, Ireland: CEUR-WS Vol. 910, 2012, pp. 21–23.

[Sai+12f] Alan Said et al. "Users and Noise: The Magic Barrier of Recommender Systems". In: *User Modeling, Adaptation, and Personalization.* Ed. by Judith Masthoff et al. Vol. 7379. Lecture Notes in Computer Science. 10.1007/978-3-642-31454-4_20. Springer Berlin / Heidelberg, 2012, pp. 237–248. ISBN: 978-3-642-31453-7. DOI: `10.1007/978-3-642-31454-4_20`. URL: `http://dx.doi.org/10.1007/978-3-642-31454-4_20`.

[SB97] Gerard Salton and Chris Buckley. "Readings in information retrieval". In: ed. by Karen Sparck Jones and Peter Willett. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997. Chap. Improving retrieval performance by relevance feedback, pp. 355–364. ISBN: 1-55860-454-5. URL: `http://dl.acm.org/citation.cfm?id=275537.275712`.

[SM86] Gerard Salton and Michael J. McGill. *Introduction to Modern Information Retrieval.* New York, NY, USA: McGraw-Hill, Inc., 1986. ISBN: 0070544840.

[SC12] M. Sanderson and W.B. Croft. "The History of Information Retrieval Research". In: *Proceedings of the IEEE* 100.Special Centennial Issue (2012), pp. 1444 –1451. ISSN: 0018-9219. DOI: `10.1109/JPROC.2012.2189916`.

[Sar+00a] Badrul Sarwar et al. "Analysis of recommendation algorithms for e-commerce". In: *Proceedings of the 2nd ACM conference on Electronic commerce.* EC '00. Minneapolis, Minnesota, United States: ACM, 2000, pp. 158–167. ISBN: 1-58113-272-7. DOI: `10.1145/352871.352887`. URL: `http://doi.acm.org/10.1145/352871.352887`.

[Sar+01] Badrul Sarwar et al. "Item-based collaborative filtering recommendation algorithms". In: *Proceedings of the 10th international conference on World Wide Web.* WWW '01. Hong Kong, Hong Kong: ACM, 2001, pp. 285–295. ISBN: 1-58113-348-0. DOI: `10.1145/371920.372071`. URL: `http://doi.acm.org/10.1145/371920.372071`.

[Sar+00b] Badrul M. Sarwar et al. "Application of Dimensionality Reduction in Recommender System – A Case Study". In: *WebKDD Workshop at the ACM SIGKKD.* 2000.

[SSA12] Christian Scheel, Alan Said, and Sahin Albayrak. "Semantic Preference Retrieval for Querying Knowledge Bases". In: *Proceedings of the 1st Joint International Workshop on Entity-oriented and Semantic Search.* JIWES. Portland, OR, USA, 2012.

[SBZ11] Yanir Seroussi, Fabian Bohnert, and Ingrid Zukerman. "Personalised rating prediction for new users using latent factor models". In: *Proceedings of the 22nd ACM conference on Hypertext and hypermedia*. HT '11. Eindhoven, The Netherlands: ACM, 2011, pp. 47–56. ISBN: 978-1-4503-0256-2. DOI: 10.1145/1995966.1995976. URL: http://doi.acm.org/10.1145/1995966.1995976.

[SG11] Guy Shani and Asela Gunawardana. "Evaluating Recommendation Systems". In: *Recommender Systems Handbook*. Springer, 2011, pp. 257–297.

[Sha01] C. E. Shannon. "A mathematical theory of communication". In: *SIGMOBILE Mob. Comput. Commun. Rev.* 5.1 (Jan. 2001), pp. 3–55. ISSN: 1559-1662. DOI: 10.1145/584091.584093. URL: http://doi.acm.org/10.1145/584091.584093.

[SM01] Barry Smyth and Paul McClave. "Similarity vs. Diversity". In: *Case-Based Reasoning Research and Development*. Ed. by David Aha and Ian Watson. Vol. 2080. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer-Verlag, 2001, pp. 347–361. ISBN: 978-3-540-42358-4. URL: http://dx.doi.org/10.1007/3-540-44593-5_25.

[Ste11] Harald Steck. "Item popularity and recommendation accuracy". In: *Proceedings of the fifth ACM conference on Recommender systems*. RecSys '11. Chicago, Illinois, USA: ACM, 2011, pp. 125–132. ISBN: 978-1-4503-0683-6. DOI: 10.1145/2043932.2043957. URL: http://doi.acm.org/10.1145/2043932.2043957.

[Sum+03] Tamara Sumner et al. "Understanding educator perceptions of "quality" in digital libraries". In: *Proceedings of the 3rd ACM/IEEE-CS joint conference on Digital libraries*. JCDL '03. Houston, Texas: IEEE Computer Society, 2003, pp. 269–279. ISBN: 0-7695-1939-3. URL: http://dl.acm.org/citation.cfm?id=827140.827188.

[SS01] Kirsten Swearingen and Rashmi Sinha. *Beyond Algorithms: An HCI Perspective on Recommender Systems*. 2001. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.3.8339.

[SNM07] Panagiotis Symeonidis, Alexandros Nanopoulos, and Yannis Manolopoulos. "Feature-Weighted User Model for Recommender Systems". In: *Proceedings of the 11th international conference on User Modeling*. UM '07. Corfu, Greece: Springer-Verlag, 2007, pp. 97–106. ISBN: 978-3-540-73077-4. DOI: 10.1007/978-3-540-73078-1_13. URL: http://dx.doi.org/10.1007/978-3-540-73078-1_13.

[THP12] Nava Tintarev, Rong Hu, and Pearl Pu. “RecSys’12 workshop on interfaces for recommender systems (InterfaceRS’12)”. In: *Proceedings of the sixth ACM conference on Recommender systems*. RecSys ’12. Dublin, Ireland: ACM, 2012, pp. 355–356. ISBN: 978-1-4503-1270-7. DOI: 10.1145/2365952.2366044. URL: http://doi.acm.org/10.1145/2365952.2366044.

[Vap95] Vladimir N. Vapnik. *The nature of statistical learning theory*. New York, NY, USA: Springer-Verlag New York, Inc., 1995. ISBN: 0-387-94559-8.

[VC11] Saúl Vargas and Pablo Castells. “Rank and relevance in novelty and diversity metrics for recommender systems”. In: *Proceedings of the fifth ACM conference on Recommender systems*. RecSys ’11. Chicago, Illinois, USA: ACM, 2011, pp. 109–116. ISBN: 978-1-4503-0683-6. DOI: 10.1145/2043932.2043955. URL: http://doi.acm.org/10.1145/2043932.2043955.

[WUS09] Robert Wetzker, Winfried Umbrath, and Alan Said. “A hybrid approach to item recommendation in folksonomies”. In: *Proceedings of the WSDM ’09 Workshop on Exploiting Semantic Annotations in Information Retrieval*. ESAIR ’09. Barcelona, Spain: ACM, 2009, pp. 25–29. ISBN: 978-1-60558-430-0. DOI: 10.1145/1506250.1506255. URL: http://doi.acm.org/10.1145/1506250.1506255.

[WZB08] Robert Wetzker, Carsten Zimmermann, and Christian Bauckhage. “Analyzing Social Bookmarking Systems: A del.icio.us Cookbook”. In: *Proceedings of the Workshop on Mining Social Data*. MSoDa ’08. ECAI 2008. 2008, pp. 26–30.

[WBE11] Martijn Willemsen, Dirk Bollen, and Michael Ekstrand. “UCERSTI 2: second workshop on user-centric evaluation of recommender systems and their interfaces”. In: *Proceedings of the fifth ACM conference on Recommender systems*. RecSys ’11. Chicago, Illinois, USA: ACM, 2011, pp. 395–396. ISBN: 978-1-4503-0683-6. DOI: 10.1145/2043932.2044020. URL: http://doi.acm.org/10.1145/2043932.2044020.

[Yin+12] Hongzhi Yin et al. “Challenging the long tail recommendation”. In: *Proc. VLDB Endow.* 5.9 (May 2012), pp. 896–907. ISSN: 2150-8097. URL: http://dl.acm.org/citation.cfm?id=2311906.2311916.

[Yu+03] Kai Yu et al. “Feature Weighting and Instance Selection for Collaborative Filtering: An Information-Theoretic Approach”. In: *Knowl. Inf. Syst.* 5.2 (Apr. 2003), pp. 201–224. ISSN: 0219-1377. DOI: 10.1007/s10115-003-0089-6. URL: http://dx.doi.org/10.1007/s10115-003-0089-6.

[Zha09] Mi Zhang. "Enhancing diversity in Top-N recommendation". In: *Proceedings of the third ACM conference on Recommender systems.* RecSys '09. New York, New York, USA: ACM, 2009, pp. 397–400. ISBN: 978-1-60558-435-5. DOI: 10.1145/1639714.1639798. URL: http://doi.acm.org/10.1145/1639714.1639798.

[Zho+10] Tao Zhou et al. "Solving the apparent diversity-accuracy dilemma of recommender systems". In: *Proceedings of the National Acadamy of Sciences of the United States of America (PNAS)* 107.10 (2010), pp. 4511–4515. DOI: http://dx.doi.org/10.1073/pnas.1000488107.

[Zie+05] Cai-Nicolas Ziegler et al. "Improving recommendation lists through topic diversification". In: *Proceedings of the 14th international conference on World Wide Web.* WWW '05. Chiba, Japan: ACM, 2005, pp. 22–32. ISBN: 1-59593-046-9. DOI: 10.1145/1060745.1060754. URL: http://doi.acm.org/10.1145/1060745.1060754.

Glossary

Application Programming Interface a specification intended to be used as an interface between different software components.. 26, 101

CineMatch Netflix's rating prediction algorithm, prior to The Netflix Prize.. 14, 101

click-through rate the number of clicks on an item divided by the number of times the item is shown. The measure is commonly applied to ad campaigns.. 49, 101

collaborative filtering Is a concept used in recommender systems for filtering information by means of analyzing the collaboration of multiple data points. 17, 101

content-based Content-based techniques analyze the content of the item (document) in order to identify its features. 17, 101

diversity the quality expressing something is different or varied.. 22, 101

F-measure a weighted average of the precision and recall.. 19, 101

feature vector a vector representing the interactions of the user (or item), each position in the vector corresponds to an entity. The vector can be unary, binary, or rated.. 18, 101

folksonomies A folksonomy is a classification system. The classification is created by collaborative tags, or annotations, of content.. 26, 101

k-Nearest Neighbor is technique for classifying objects in a feature space based on some distance, similarity or correlation measure. 101

kNN k-Nearest Neighbor. 101

magic barrier A maximum obtainable level of performance a recommender system can attain.. 10, 101

nDCG normalized Discounted Cumulative Gain. 101

netizen the citizens of the Web, i.e. users of the Web.. 2, 101

normalized discounted cumulative gain a relevance scale in aresult set. nDCG is commonly used for evaluation of rating prediction.. 5, 101

novelty the quality expressing that something is new.. 22, 101

objective metrics Objective metrics, or measures, express the interaction events between a user and a system, e.g. ratings, purchases, bounce rate, etc.. 22, 101

online The online concept refers to any information available on the Internet. 1–4, 6, 7, 101

precision the fraction of retrieved documents that are relevant.. 3–5, 19, 101

ranking the act of predicting the highest ranked items according to a user's taste.. 3, 101

rating prediction the act of predicting a the rating that a user will given an item.. 3, 101

recall the fraction of the documents that are relevant that are successfully retrieved.. 3, 5, 19, 101

receiver operating characteristic a curve to plot the true positive rate vs. the false positive rate. Summing up the area under ROC curve can be used as a measure to express the quality of a recommender system.. 3, 101

recommender system is a programmable machine that receives input, stores and manipulates data, and provides output in a useful format. 101

RMSE Root-mean-square error. 101

Root-Mean-Square Error is a measure describing the differences of two populations. Commonly used to measure rating prediction accuracy.. 3, 101

Root-Mean-Square Error Root-Mean-Square Error.. 5, 101

serendipity the ability of making accidental fortunate discoveries.. 22, 101

subjective metrics Subjective metrics, or measures, express the subjective qualities of a system as perceived by individual users.. 23, 101

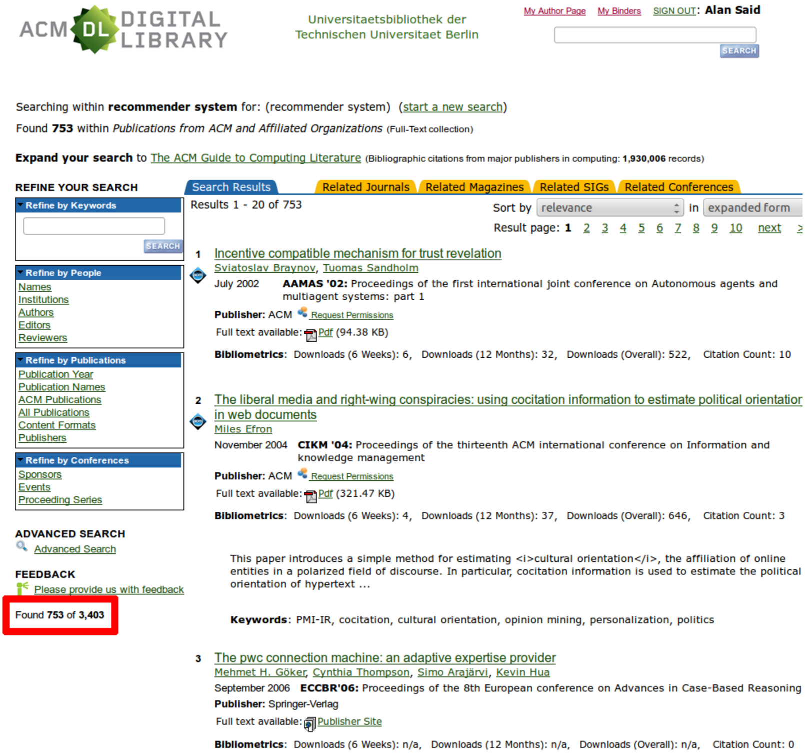
top-n see ranking.. 3–5, 101

Web short for The World Wide Web. 1, 101

Web 2.0 a term to describe highly dynamic and interactive websites differing from the first generation of static Web pages through an increased level of user-generated content.. 2, 101

Appendices

Images



Universitaetsbibliothek der
Technischen Universitaet Berlin

My Author Page My Binders SIGN OUT: **Alan Said**

SEARCH

Searching within **recommender system** for: (recommender system) (start a new search)

Found **753** within *Publications from ACM and Affiliated Organizations* (Full-Text collection)

Expand your search to The ACM Guide to Computing Literature (Bibliographic citations from major publishers in computing: **1,930,006** records)

REFINE YOUR SEARCH

Refine by Keywords

SEARCH

Refine by People

Names
Institutions
Authors
Editors
Reviewers

Refine by Publications

Publication Year
Publication Names
ACM Publications
All Publications
Content Formats
Publishers

Refine by Conferences

Sponsors
Events
Proceeding Series

ADVANCED SEARCH

Advanced Search

FEEDBACK

Please provide us with feedback

Found **753** of **3,403**

Search Results Related Journals Related Magazines Related SIGs Related Conferences

Results 1 - 20 of 753

Sort by relevance in expanded form

Result page: **1** 2 3 4 5 6 7 8 9 10 next >

1 Incentive compatible mechanism for trust revelation
Sviatoslav Braynov, Tuomas Sandholm
July 2002 **AAMAS '02:** Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 1
Publisher: ACM Request Permissions
Full text available: Pdf (94.38 KB)
Bibliometrics: Downloads (6 Weeks): 6, Downloads (12 Months): 32, Downloads (Overall): 522, Citation Count: 10

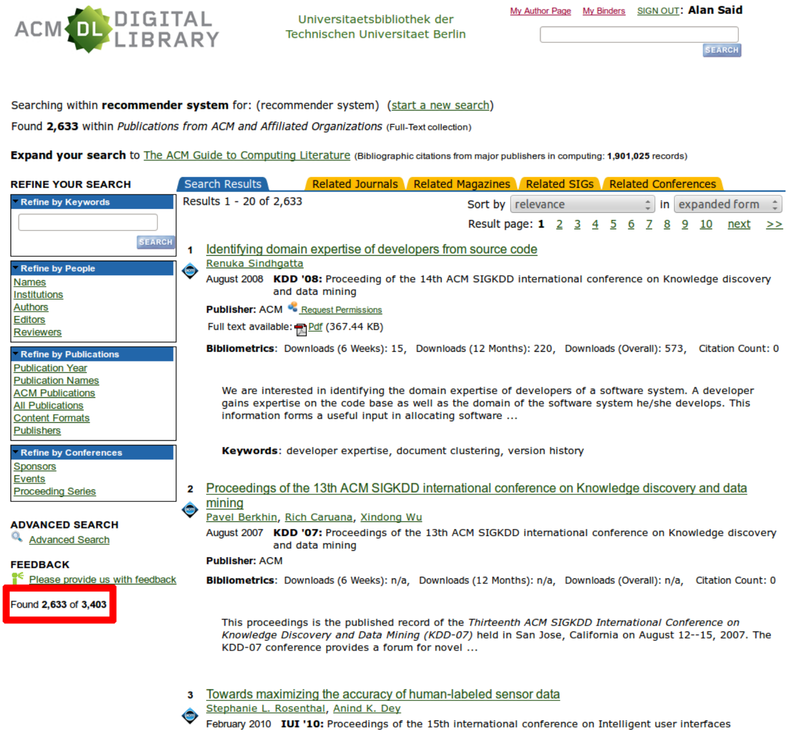
2 The liberal media and right-wing conspiracies: using cocitation information to estimate political orientatior in web documents
Miles Efron
November 2004 **CIKM '04:** Proceedings of the thirteenth ACM international conference on Information and knowledge management
Publisher: ACM Request Permissions
Full text available: Pdf (321.47 KB)
Bibliometrics: Downloads (6 Weeks): 4, Downloads (12 Months): 37, Downloads (Overall): 646, Citation Count: 3

This paper introduces a simple method for estimating <i>cultural orientation</i>, the affiliation of online entities in a polarized field of discourse. In particular, cocitation information is used to estimate the political orientation of hypertext ...

Keywords: PMI-IR, cocitation, cultural orientation, opinion mining, personalization, politics

3 The pwc connection machine: an adaptive expertise provider
Mehmet H. Göker, Cynthia Thompson, Simo Arajärvi, Kevin Hua
September 2006 **ECCBR'06:** Proceedings of the 8th European conference on Advances in Case-Based Reasoning
Publisher: Springer-Verlag
Full text available: Publisher Site
Bibliometrics: Downloads (6 Weeks): n/a, Downloads (12 Months): n/a, Downloads (Overall): n/a, Citation Count: 0

Figure 1: Number of papers containing the term "recommender system" in the ACM DL published prior to 2007. Retrieved June 14th 2012.



Searching within **recommender system** for: (recommender system) (start a new search)

Found **2,633** within *Publications from ACM and Affiliated Organizations* (Full-Text collection)

Expand your search to The ACM Guide to Computing Literature (Bibliographic citations from major publishers in computing: **1,901,025** records)

Found **2,633** of **3,403**

Results 1 - 20 of 2,633

1 Identifying domain expertise of developers from source code
Renuka Sindhgatta
August 2008 **KDD '08:** Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining
Publisher: ACM Request Permissions
Full text available: Pdf (367.44 KB)
Bibliometrics: Downloads (6 Weeks): 15, Downloads (12 Months): 220, Downloads (Overall): 573, Citation Count: 0

We are interested in identifying the domain expertise of developers of a software system. A developer gains expertise on the code base as well as the domain of the software system he/she develops. This information forms a useful input in allocating software ...

Keywords: developer expertise, document clustering, version history

2 Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining
Pavel Berkhin, Rich Caruana, Xindong Wu
August 2007 **KDD '07:** Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining
Publisher: ACM
Bibliometrics: Downloads (6 Weeks): n/a, Downloads (12 Months): n/a, Downloads (Overall): n/a, Citation Count: 0

This proceedings is the published record of the *Thirteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-07)* held in San Jose, California on August 12--15, 2007. The KDD-07 conference provides a forum for novel ...

3 Towards maximizing the accuracy of human-labeled sensor data
Stephanie L. Rosenthal, Anind K. Dey
February 2010 **IUI '10:** Proceedings of the 15th international conference on Intelligent user interfaces

Figure 2: Number of papers containing the term “recommender system” in the ACM DL published from 2007. Retrieved June 14th 2012.

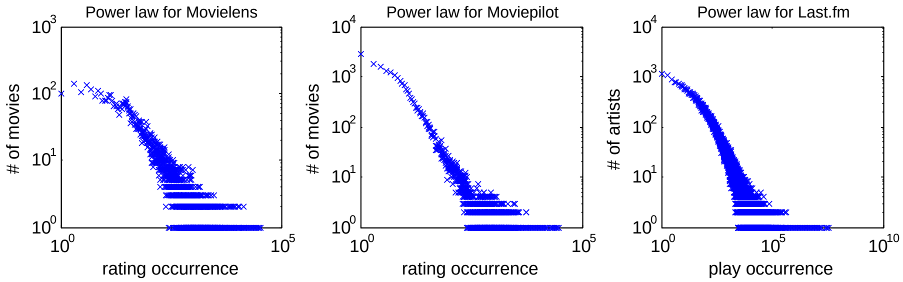


Figure 3: The power law of the Movielens, Moviepilot and Last.fm datasets respectively.

Websites

Website	**URL**
ACM DL	`http://dl.acm.org`
Alexa	`http://www.alexa.com`
Amazon	`http://www.amazon.com`
CAMRa 2010	`http://www.dai-labor.de/camra2010`
Filmtipset	`http://www.filmtipset.se`
GroupLens Research	`http://www.gouplens.org`
IMDb	`http://www.imdb.com`
iTunes	`http://www.itunes.com`
KDD Cup 2011	`http://kddcup.yahoo.com`
LastFM	`http://www.last.fm`
Lycos	`http://www.lycos.com`
Apache Mahout	`http://mahout.apache.org`
Movielens	`http://www.movielens.org`
moviepilot	`http://www.moviepilot.de`
Netflix	`http://www.netflix.com`
Netflix Prize	`http://www.netflixprize.com`
Spotify	`http://www.spotify.com`
The Free Dictionary	`http://www.thefreedictionary.com`
Today Translations	`http://www.todaytranslations.com`
Yahoo! Music	`http://music.yahoo.com`
YouTube	`http://www.youtube.com`

Table 1: Websites mentioned in this thesis.