Setting Goals and Choosing Metrics for Recommender System Evaluations

Gunnar Schröder
T-Systems Multimedia Solutions GmbH
Riesaer Straße 5
01129 Dresden, Germany
gunnar.schroeder@t-systems.com

Maik Thiele, Wolfgang Lehner
Dresden University of Technology
Faculty of Computer Science, Database Technology Group
01062 Dresden, Germany
{maik.thiele,wolfgang.lehner}@tu-dresden.de

ABSTRACT

Recommender systems have become an important personalization technique on the web and are widely used especially in e-commerce applications. However, operators of web shops and other platforms are challenged by the large variety of available algorithms and the multitude of their possible parameterizations. Since the quality of the recommendations that are given can have a significant business impact, the selection of a recommender system should be made based on well-founded evaluation data. The literature on recommender system evaluation offers a large variety of evaluation metrics but provides little guidance on how to choose among them. This paper focuses on the often neglected aspect of clearly defining the goal of an evaluation and how this goal relates to the selection of an appropriate metric. We discuss several well-known accuracy metrics and analyze how these reflect different evaluation goals. Furthermore we present some less well-known metrics as well as a variation of the area under the curve measure that are particularly suitable for the evaluation of recommender systems in e-commerce applications.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms

Theory

Keywords

recommender systems, e-commerce, evaluation, metrics, measure, area under the curve, auc, informedness, markedness, matthews correlation, precision, recall, roc

1. INTRODUCTION

There is a large variety of algorithms for recommender systems that were published in research and have been implemented by the industry. Almost every author claims that a particular algorithm or implementation is superior to another in a certain respect, which makes it difficult to choose among them. Currently a comprehensive and objective comparison of existing recommender systems is hard to find and the available results are sometimes contradictory for different data sets or metrics. How efficient and successful a specific recommender system is also depends on the specific purpose of a recommender system and the characteristics of the domain it is applied to. It is very unlikely that there is a single best solution for any domain and context. If we want to increase the effectiveness of recommendations we have to determine the best fit for a given scenario through thorough evaluation of available algorithms and parameterizations.

This is particularly important for the usage of recommender systems in e-commerce applications where the choice of algorithms can have a significant business impact. In a web shop a better recommender system can have a direct effect on the company's revenue since the recommendations can significantly influence the users' buying decisions [15].

The research that is presented in this paper is derived from an evaluation of various recommendation algorithms that we conducted for a large German e-commerce portal. In order to achieve meaningful evaluation results we developed an evaluation methodology and went through a wide range of literature on the evaluation of recommender systems and information retrieval systems and developed a framework for the evaluation of recommender systems.

In this paper we will focus on the specific evaluation demands of recommender systems in e-commerce applications. We discuss the importance of defining a sufficiently detailed goal and analyze which aspects of the recommender's user interface and the used preference data influence a reasonable choice of metrics. We give an overview of applicable accuracy metrics, explain how to choose among the large variety and highlight some metrics that are particularly well-suited to the evaluation of recommender systems. In order to discuss accuracy metrics in detail a discussion of non-accuracy measures has to be omitted due to space constraints.

2. RELATED WORK

Over time numerous quantitative metrics and qualitative techniques for offline and online evaluations of recommender systems have been published in research. We will start by giving a short overview of some of the most important publications that are relevant to this paper.

Herlocker et al. provide a comprehensive overview of the existing methods for evaluating collaborative filtering systems [9]. Although they focus on collaborative filtering methods, many of the presented evaluation approaches, metrics and techniques are applicable to other types of recommender

systems as well. They conducted an insightful study on the correlation of different metrics and concluded that the analyzed metrics can be subdivided in three major classes.
Olmo and Gaudioso build on this survey and derive a framework for recommender systems that divides recommender systems into a *filter* and a *guide* component [4]. Their aim is to separate the calculation of recommendations from their presentation. They propose to use metrics that focus on the fact whether the recommendations presented by the system are actually followed by the users of the recommender system and suggest to pay more attention to the presentation of recommendations as well as the respective objective of recommender systems.
Cremonesi and Lentini present an evaluation methodology for collaborative filtering recommender systems [2]. They use mean squared error, root mean squared error (RMSE) and mean absolute error (MAE) as well as the classification accuracy metrics precision, recall, f-measure and ROC graphs to compare two collaborative filtering algorithms using the MovieLens[1] data set and a further movie data set obtained from an IPTV service provider.
Konstan et al. summarize findings about the usage of automated recommender systems for information-seeking tasks [12]. They list many problems and challenges in evaluating recommender systems and emphasize the importance of standardized evaluation methodologies, metrics and test data sets for progress in research.
Kohavi et al. provide a survey and practical guide for conducting controlled experiments on the web [11]. In a follow-up paper Crook et al. describe common pitfalls they experienced and emphasize the importance of choosing an *overall evaluation criterion* that truly reflects business goals [3].
Jannach et al. provide a chapter on evaluating recommender systems in their book [10]. They review the current state of research and survey the approaches taken in published papers on the evaluation of recommender systems.
Shani and Gunawardana contributed a chapter on evaluating recommender systems to the handbook by Ricci et al. [14] and describe the most important aspects in conducting offline and online experiments as well as user studies. They outline basic approaches for all three types of evaluations, some important accuracy metrics and discuss various other properties of recommenders that should be considered when evaluating recommenders.
A further valuable source for metrics and measures is the literature on information retrieval. It is, by its very nature, concerned with the quality of search results and provides insight into evaluation methodologies and metrics (e.g.[5]).

The evaluation of recommender systems has emerged as an important topic as more and more algorithms and techniques for recommenders systems are presented. Many different evaluation metrics can be applied in evaluations, although some (e.g. RMSE, MAE and precision) are more frequently used than others. However, authors rarely justify their particular choice and little guidance is offered on how to choose a metric for a specific purpose. Some notable exceptions are the comparison of metrics given by Herlocker et al. [9] and the overview by Jannach et al. [10]. In this paper we try to provide the reader with a better understanding of the existing accuracy metrics and how to apply them in order to evaluate recommenders for specific goals.

[1] `http://www.grouplens.org/node/73`

3. SETTING THE EVALUATION GOAL

The first step of an evaluation should be to define its goal as precisely as possible. Although this seems rather obvious, we would like to emphasize this step since it is often not executed with sufficient care in evaluations in general. We can only choose one or several adequate metrics and interpret their results, if we have set the purpose of our evaluation beforehand. The results of popular evaluation metrics such as RMSE, precision or F_1-measure do not necessarily lead us to the best algorithm for our purpose, as we try to illustrate later on. Only if the set of metrics accurately reflects the specific objectives that are connected to the recommender system in our evaluation and the concrete usage scenario, can we be sure that the obtained results are useful [11, 3]. Moreover, the specific interface choice for the recommender system and the usage patterns that are to be expected should be taken into account. Such a precise evaluation goal could be: Find the recommendation algorithm and parameterization that leads to the highest overall turnover on a specific e-commerce web site, if four product recommendations are displayed as a vertical list below the currently displayed product.

3.1 Objectives for Recommender Usage

Having a clear conception of the objectives we want to achieve by designing, implementing or applying a recommender system is a worthwhile effort, even if we recognize that it may be too difficult or expensive to measure them accurately. In many cases we may discover that only the results of a large scale field study would accurately reflect our evaluation goal. Nevertheless a defined goal will help us in selecting the most appropriate metric or set of metrics that allows us to obtain the most precise measurement of the recommender's suitability for the usage scenario which can be achieved with a reasonable effort.
Common reasons for implementing a recommender system are the desire to improve user satisfaction and to increase the economic success of a platform. Although both goals are interrelated they may be competing in some scenarios. As an example, in e-commerce a recommender may either determine the top recommendations based on the best price-performance ratio for the customer but it may also show the products that are likely to lead to the highest revenue for the business. For this purpose commercial recommenders for web shops often consider a reward attribute for items that models how much the company profits from a sale of a certain item. This information can be used e.g. in combination with classification accuracy metrics (see Section 4.2).
A recommender system in a live environment usually has to be optimized with regard to various other objectives that are related to the technical performance and the system's life cycle such as responsiveness, scalability, peak load, reliability, ramp-up efforts, maintainability, extensibility and cost of ownership [10]. These aspects have a strong influence on the decision among a set of algorithms and implementations and put further constraints on a choice. Further aspects that extend beyond the accuracy of a recommender are coverage, novelty, serendipity, confidence, persuasiveness, trust, robustness, security and many more (cf. [9, 14]). Measuring these diverse qualities of a recommender system is a large field that is beyond the scope of the presented work. We will confine our analysis to accuracy metrics and attempt to analyze how different measures reflect varying objectives.

3.2 Analyzing the Recommender System and its Data

In order to further specify our evaluation goal we have to take a closer look at the recommender's task, its interface and the utilized data.

What is the recommender's task and how does the system interact with the user? In our opinion the most common usage scenarios for recommender systems are *prediction*, *ranking* and *classification* tasks which manifest themselves in different types of user interfaces.

In a *prediction* task the focus lies on predicting ratings for unrated items (e.g. movies, music or news articles) of a user and showing these predictions to him. For example, a movie rental company might show predicted ratings to users in order to aid them in their decision making.

In a *ranking* task the recommender tries to determine an order of items, often with the purpose of creating a top-k list of items. This task is particularly common in e-commerce applications, where a top-k or unlimited ordered list of recommended products is shown in a sidebar or on a dedicated page. But also web portals for news, content and multimedia make heavy use of top-k lists. A further interesting ranking task for a recommender is to order a limited set of search results according the user's predicted preference.

In a *classification* task the recommender determines a set with a limited (often fixed) number of recommended items with no particular order among them implied. E.g. articles or products that are predicted to be of interest to a user are highlighted or annotated on a web page. A top-k list of recommended items can be seen as a classification task as well, especially if the number of items is very small and the order is less prominent e.g. recommended items are arranged in a grid or scrollable in two directions in a circular fashion (cf. Amazon).

Recommender systems are further used to determine *similar items* or *similar users*. E.g. in web shops on a product detail page usually a list of similar products is shown or a news web site often lists articles similar to the one currently shown. Recommending similar users is of particular interest for social web applications such as Facebook. These tasks which focus on the items and users themselves instead of the user's ratings, can be seen as ranking or classification tasks depending on the chosen interface.

Realizing for which of these tasks the recommender system is used and what the user interface looks like is important for choosing the most appropriate metric. The number of displayed recommendations and their visual arrangement (To which extent does it convey a ranking?) should also be considered before choosing a metric. In many usage scenarios more than one of these tasks has to be fulfilled by a recommender system. Therefore it may be sensible to apply one accuracy metric for each of the usage scenarios to find a reasonable compromise or to even consider using different recommendation algorithms for each task.

What kind of user preference data is used by the recommender? The preferences of a user can be gathered either through *explicit* or *implicit* ratings. While an explicit rating is made as a deliberate statement by a user who has the intention to express his or her opinion, an implicit rating is deducted from actions of the user that had a different primary goal. If, for example, a user rates a movie on a web site, this is a direct expression of his or her opinion and preferences, so we consider it an explicit rating. Purchasing a product or clicking on a link to display an article usually has a different primary goal than expressing a positive rating, so these actions are considered an implicit rating.

The user ratings are usually collected using either a *numerical*, a *binary* or a *unary* rating scale. Although other preference models such as textual reviews are conceivable they are rarely used in today's recommender systems.

A *numerical* rating is represented by a number from either a discrete or a continuous rating scale, in most cases with a limited range. Typical examples of a discrete rating scale are ratings on a scale from zero to five stars or Likert response scales that are commonly used in questionnaires. To be precise, these rating scales are actually *ordinal* scales, a fact which is ignored by predictive accuracy metrics (cf. 4.1) that make intensive use of ratios. An example of a continuous rating scale could be a slider that is set by a user and translated to a real value.

A *binary* rating scale allows users to assign items to two different classes (like/dislike). YouTube, for example, allows users to rate movies with either thumb up or thumb down.

A *unary* rating, by contrast, allows users to assign items only to a single class, which is in most cases positive (e.g. like). A prominent example of an explicit unary rating is Facebook's "Like"-button. Implicit unary ratings can be purchased products in a web shop or clicked links on a news page.

The important difference between binary and unary ratings is that unary ratings offer no distinction between disliked and unrated items. With unary ratings we cannot tell whether a user actually dislikes an item or simply does not know the item or does not bother to rate it. In a large web shop, such as Amazon, a customer will only be aware of a small portion of the product catalog. Furthermore, other factors such as limited budget, limited time, external constraints, and products the user already owns determine whether a customer will actually buy a product he or she likes. Being aware of this difference and the implied biases is important when conducting an evaluation and interpreting the results of various metrics.

4. OVERVIEW OF EVALUATION METRICS

Evaluation metrics for recommender systems can be divided into four major classes [9, 4]: 1) Predictive accuracy metrics, 2) Classification accuracy metrics, 3) Rank accuracy metrics and 4) Non-accuracy metrics. Since we confine our analysis to accuracy metrics we will omit the discussion of this class and suggest to consult the literature (e.g. [9, 14]).

4.1 Predictive Accuracy Metrics

Predictive accuracy or *rating prediction* metrics embark on the question of how close the ratings estimated by a recommender are to the true user ratings. This type of measures is very popular for the evaluation of non-binary ratings. It is most appropriate for usage scenarios in which an accurate prediction of the ratings for all items is of high importance. The most important representatives of this class are *mean absolute error* (MAE), *mean squared error* (MSE), *root mean squared error* (RMSE) and *normalized mean absolute error* (NMAE) (cf. [9, 8]). MSE and RMSE use the squared deviations and thus emphasize larger errors in comparison to the MAE metric. MAE and RMSE describe the error in the same units as the computed values, while MSE yields

squared units. NMAE normalizes the MAE metric to the range of the respective rating scale in order to make results comparable among recommenders with varying rating scales. The RMSE metric has been used in the Netflix competition in order to determine the improvement in comparison to the Cinematch algorithm as well as the prize winner. This was a significant source of discussion over the course of the competition.

These predictive accuracy error metrics are frequently used for the evaluation of recommender systems since they are easy to compute and understand. They are well-studied and are also applied in many contexts other than recommender systems. However, they do not necessarily correspond directly to the most popular usage scenarios for recommender systems. There are few cases where users are in fact interested in the overall prediction accuracy. Recommender systems are more commonly used to display a limited list of top ranked items or the set of all items that have been rated above a certain threshold. Many recommendation algorithms are able to provide more accurate statements about a limited set of items that the user either likes or dislikes. The estimations for many other items are rather inaccurate but often also significantly less important to users.

This applies in particular to e-commerce applications where we are usually more concerned with suggesting some products to customers that they will like in comparison to estimating the most accurate ratings for the large amount of items that customers would never purchase.

4.2 Classification Accuracy Metrics

Classification accuracy metrics try to assess the *successful decision making capacity* (SDMC) of recommendation algorithms. They measure the amount of correct and incorrect classifications as relevant or irrelevant items that are made by the recommender system and are therefore useful for user tasks such as finding good items. SDMC metrics ignore the exact rating or ranking of items as only the correct or incorrect classification is measured [9, 4]. This type of metric is particularly suitable for applications in e-commerce that try to convince users of making certain decisions such as purchasing products or services.

A comprehensive overview of classic, basic information retrieval metrics, such as recall and precision and further improved metrics, is given in the survey by Powers [13]. The following short summary is based on his terminology and descriptions for these metrics. In section 5.1 we will discuss three less well-known metrics that were described in this paper as well and that we deem very useful. To the best of our knowledge they have not been used for the evaluation of recommender systems so far.

The common basic information retrieval (IR) metrics are calculated from the number of items that are either relevant or irrelevant and either contained in the recommendation set of a user or not. These numbers can be clearly arranged in a contingency table that is sometimes also called the *confusion matrix* (see Table 1).

Precision or *true positive accuracy* (also *confidence* in data mining) is calculated as the ratio of recommended items that are relevant to the total number of recommended items:

$$\text{precision} = \text{tpa} = \frac{tp}{tp + fp}$$

This is the probability that a recommended item corresponds to the user's preferences. The behavior of this (and the following) IR metrics can be observed in Figure 1 which shows a potential ranking of four relevant items in a set of ten items by a recommender. In examples (a) to (j) the length k of the top-k list varies, while the item order remains fixed.

	Relevant	Irrelevant	Total
Recommended	tp	fp	$tp + fp$
Not Recommended	fn	tn	$fn + tn$
Total	$tp + fn$	$fp + tn$	N

Table 1: A contingency table or confusion matrix that accumulates the numbers of true/false positive/negative recommendations.

Recall or *true positive rate* (also called *sensitivity* in psychology) is calculated as the ratio of recommended items that are relevant to the total number of relevant items:

$$\text{recall} = \text{tpr} = \frac{tp}{tp + fn}$$

This is the probability that a relevant item is recommended. The two measures, precision and recall, are inversely related which we notice if we vary the size of the set of recommendations (cf. Fig. 1). In most cases, increasing the size of the recommendation set will increase recall but decrease precision. Because of this mutual dependence it makes sense to consider precision and recall in conjunction with two other metrics that are called *fallout* and *miss rate*.

Fallout or *false positive rate* is calculated as the ratio of recommended items that are irrelevant to the total number of irrelevant items:

$$\text{fallout} = \text{fpr} = \frac{fp}{fp + tn}$$

This is the probability that an irrelevant item is recommended.

Miss rate or *false negative rate* is calculated as the ratio of items not recommended but actually relevant to the total number of relevant items:

$$\text{missRate} = \text{fnr} = \frac{fn}{tp + fn}$$

This is the probability that a relevant item is not recommended.

Just as precision and recall describe the recommender's performance regarding the true positives, two similar metrics can be defined that measure how the algorithm behaves regarding true negatives. Since this corresponds to interchanging the true and false values of the underlying data for the precision and recall metric, they are called *inverse precision* and *inverse recall*.

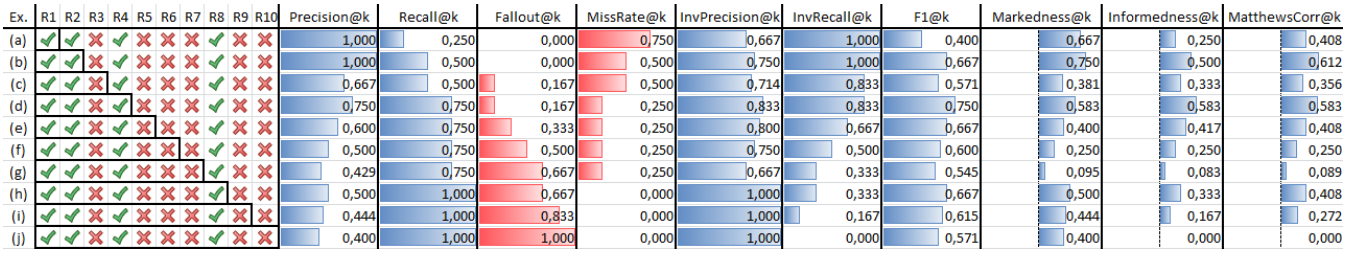
Inverse precision or *true negative accuracy* is calculated as the ratio of items not recommended that are indeed irrelevant to the total number of not recommended items:

$$\text{inversePrecision} = \text{tna} = \frac{tn}{fn + tn}$$

This is the probability that an item which is not recommended is indeed irrelevant.

Inverse recall or *true negative rate* (also called *specificity*) is calculated as the ratio of items not recommended that are really irrelevant to the total number of irrelevant items:

$$\text{inverseRecall} = \text{tnr} = \frac{tn}{fp + tn} = 1 - \text{fpr}$$



Ex.	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	Precision@k	Recall@k	Fallout@k	MissRate@k	InvPrecision@k	InvRecall@k	F1@k	Markedness@k	Informedness@k	MatthewsCorr@k
(a)	✓	✓	✗	✓	✗	✗	✗	✓	✗	✗	1,000	0,250	0,000	0,750	0,667	1,000	0,400	0,667	0,250	0,408
(b)	✓	✓	✗	✓	✗	✗	✗	✓	✗	✗	1,000	0,500	0,000	0,500	0,750	1,000	0,667	0,750	0,500	0,612
(c)	✓	✓	✗	✓	✗	✗	✗	✓	✗	✗	0,667	0,500	0,167	0,500	0,714	0,833	0,571	0,381	0,333	0,356
(d)	✓	✓	✗	✓	✗	✗	✗	✓	✗	✗	0,750	0,750	0,167	0,250	0,833	0,833	0,750	0,583	0,583	0,583
(e)	✓	✓	✗	✓	✗	✗	✗	✓	✗	✗	0,600	0,750	0,333	0,250	0,800	0,667	0,667	0,400	0,417	0,408
(f)	✓	✓	✗	✓	✗	✗	✗	✓	✗	✗	0,500	0,750	0,500	0,250	0,750	0,500	0,600	0,250	0,250	0,250
(g)	✓	✓	✗	✓	✗	✗	✗	✓	✗	✗	0,429	0,750	0,667	0,250	0,667	0,333	0,545	0,095	0,083	0,089
(h)	✓	✓	✗	✓	✗	✗	✗	✓	✗	✗	0,500	1,000	0,667	0,000	1,000	0,333	0,667	0,500	0,333	0,408
(i)	✓	✓	✗	✓	✗	✗	✗	✓	✗	✗	0,444	1,000	0,833	0,000	1,000	0,167	0,615	0,444	0,167	0,272
(j)	✓	✓	✗	✓	✗	✗	✗	✓	✗	✗	0,400	1,000	1,000	0,000	1,000	0,000	0,571	0,400	0,000	0,000

Figure 1: Varying the length k of a top-k list for a possible recommender's ranking of ten items.

This is the probability that an irrelevant item is indeed not recommended.

The F_1-*measure* and F_β-*measure* try to combine precision and recall into a single score by calculating different types of means of both metrics. The F_1-measure or F_1-score is calculated as the standard *harmonic mean* of precision and recall:

$$\mathrm{F}_1 = \frac{2}{\frac{1}{\mathrm{precision}} + \frac{1}{\mathrm{recall}}} = \frac{2 \cdot \mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$$

Moreover there is a large variety of metrics that are derived from these basic information retrieval metrics [9, 5]. *Mean average precision* (MAP) is a popular metric for search engines and is applied, for example, to report results at the Text Retrieval Conference (TREC) [5]. It takes each relevant item and calculates the precision of the recommendation set with the size that corresponds to the rank of the relevant item. Then the arithmetic mean of all these precisions is formed.

$$\mathrm{AP} = \frac{\sum_{r=1}^{N} (P(r) \times \mathrm{rel}(r))}{\text{number of relevant documents}}$$

Afterwards we calculate the arithmetic mean of the average precisions of all users to get the final mean average precision:

$$\mathrm{MAP} = \frac{\sum_{u=1}^{M} \mathrm{AP}_u}{M}$$

Various other means including the geometric mean (GMAP), harmonic mean (HMAP) and quadratic mean (QMAP) can be applied instead. All of these measures emphasize true positives at the top of a ranking list. This behaviour can be observed in Figure 2 (a) to (j) where the position of a single relevant item within a ranking of ten items is varied.

Further derived measures that we will discuss later are *ROC curves* and the *area under the curve* (AUC) measure.

4.3 Rank Accuracy Metrics

A *rank accuracy* or *ranking prediction* metric measures the ability of a recommender to estimate the correct order of items concerning the user's preference, which is called the measurement of *rank correlation* in statistics. Therefore this type of measure is most adequate if the user is presented with a long ordered list of items recommended to him. A rank prediction metric uses only the relative ordering of preference values so that is independent of the exact values that are estimated by a recommender. For example, a recommender that constantly estimates items' ratings to be lower than the true user preferences, would still achieve a perfect score as long as the ranking is correct.

For the usage of rank accuracy metrics, it is important to know whether they measure total or partial orderings. Most rank accuracy metrics such as *Kendall's tau* and *Spearman's rho* compare two total orderings. The problem with these measures is that in most cases we are not provided with a full ranking of all items. Many recommendation algorithms can generate only a partial list of items that are most likely to be preferred by a user. All other items would have to be concatenated to the list in an arbitrary order. Furthermore, the recommendation list can contain groups of items with a similar rating that can appear in varying orders. The same applies to the true preferences of a user. In order to create a full ranking of the items all preference values for the user have to be known. Since the user might express the same rating for several items the list will again contain groups of items that can appear in an arbitrary order. The largest problem is posed by items for which no user rating is known. These items could in fact hold an arbitrary place within the ranking. Sometimes it is assumed that ratings for items that are preferred are known, so that the unknown items are concatenated to the end of the list. In general, however, the unknown items could as well contain items that would appear within the top ranks if rated by the user [9].

The bottom line is that in most cases a rank metric for partial orderings would be more appropriate for comparing recommendation lists that are produced by recommenders to item rankings from known user preferences. One possibility is to use an arbitrary total ordering that complies with the partial ordering, though the results will become less meaningful when the number and size of item groups with identical rating increases. An evaluation might state that a certain ranking is inferior even though only items within groups of identical ratings switched their places. A better solution is to compare all or a subset of the total orderings that comply with the partial orderings and average the results. However, this can easily become combinatorially challenging when both rankings are in fact partial orderings. Fagin et al. discuss several derived metrics for comparing partial orderings that could be applied for recommender systems [6]. Bansal and Fernandez-Baca provide runtime efficient implementations of these metrics [1].

We think that rank accuracy metrics are very well suited for e-commerce applications if they allow to compare partial rankings in a meaningful way. In our implementation we used a variation of the Kendall's tau metric for partial rankings on boolean classifications. That is, we compare the partial ranking provided by recommended and not recommended items with the partial ranking of relevant and irrelevant items. This boolean Kendall's tau is in fact highly correlated to the AUC metric (e.g. Fig. 2).

5. IMPROVED METRICS FOR RECOMMENDER SYSTEMS

5.1 Informedness, Markedness and Matthews Correlation

A major problem with the frequently used metrics precision, recall and F_1-measure is that they suffer from severe biases. The outcome of these measures not only depends on the accuracy of the recommender (or classifier) but also very much on the ratio of relevant items. In order to illustrate this we imagine an ideal recommender and its inverse and apply them to generate top-k lists of four items for varying ratios of relevant items (see Fig. 3). The examples show that the informatory value of all three measures varies. In extreme cases with a highly skewed ratio its value can be even misleading (cf. Fig. 3 e – h).
To solve this problem Powers [13] introduced three new metrics that try avoid these biases by integrating the inverse precision and inverse recall respectively.
Markedness combines precision and inverse precision into a single measure and expresses how marked the classifications of a recommender are in comparison to chance:

$$\begin{aligned}\text{markedness} &= \text{precision} + \text{inversePrecision} - 1 \\ &= \frac{tp}{tp+fp} + \frac{tn}{fn+tn} - 1\end{aligned}$$

Informedness combines recall and inverse recall into a single measure and expresses how informed the classifications of a recommender are in comparison to chance:

$$\begin{aligned}\text{informedness} &= \text{recall} + \text{inverseRecall} - 1 \\ &= \frac{tp}{tp+fn} + \frac{tn}{fp+tn} - 1\end{aligned}$$

Both markedness and informedness return values in the range $[-1, 1]$.
The *Matthews Correlation* combines the informedness and markedness measures into a single metric by calculating their geometric mean:

$$\begin{aligned}\text{correlation} &= \frac{(tp \cdot tn) - (fp \cdot fn)}{\sqrt{(tp+fn) \cdot (fp+tn) \cdot (tp+fp) \cdot (fn+tn)}} \\ &= \pm\sqrt{\text{informedness} \cdot \text{markedness}}\end{aligned}$$

The range of the Matthews Correlation is $[-1, 1]$, so the sign in the second representation of the formula actually depends on the respective signs for markedness and informedness.
We propose to replace the measures precision and recall by markedness and informedness for most evaluation purposes in order to avoid being misled by underlying biases. As we can see in Figures 1 and 3, markedness, informedness and Matthews Correlation are significantly more helpful in choosing an adequate size for a top-k recommendation list. Furthermore derived metrics such as mean average precision (MAP) could as well be replaced by their respective equivalents (e.g. mean average markedness).

5.2 Limited Area Under the Curve

ROC curves provide a graphical representation for the performance of a recommender system, an information retrieval system or any other type of binary classifier. A ROC curve plots recall (true positive rate) against fallout (false positive rate) for increasing recommendation set size. An in-depth discussion of ROC curves can be found in the paper by Fawcett [7].

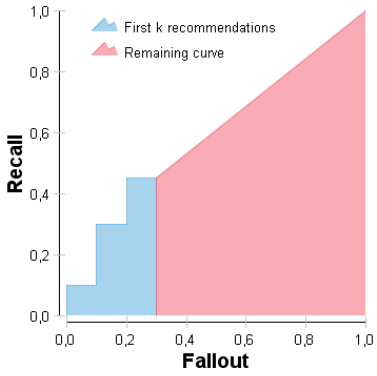
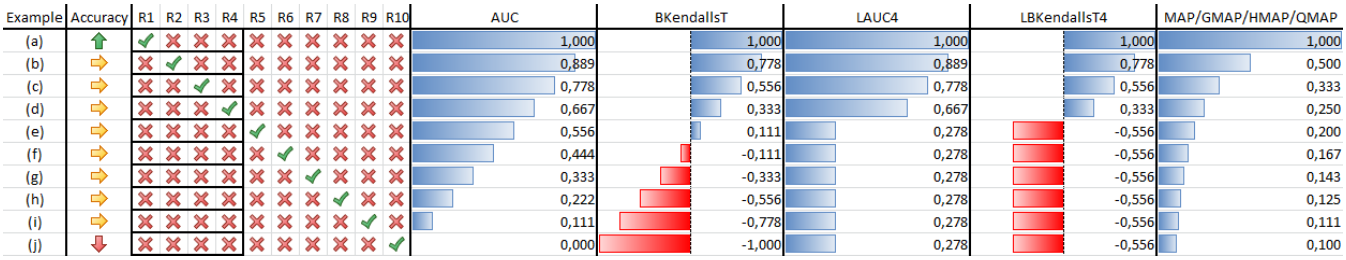


Figure 5: An example for the ROC curve used by our variant of the AUC measure.

A perfect recommender would yield a ROC curve that goes straight up towards 1.0 recall and 0.0 fallout until all relevant items are retrieved. Afterwards it would go straight right towards 1.0 fallout while the remaining irrelevant items follow. The obvious aim is consequently to maximize the area under the ROC curve. The *area under the curve* (AUC) can therefore be used as a single measure for the overall quality of a recommender system. However, a frequently uttered point of criticism is that users are often more interested in the items at the top of recommendation lists but that the AUC measure is equally affected by swaps at the top or the bottom of a recommendation list (cf. Figures 2 and 4 e – h). This may be a disadvantage if we are mainly interested in finding the top ranked items and thus care mostly for the first part of the ROC graph. Therefore, in addition to the standard AUC measure, we implemented a slight variation of the AUC measure. This *limited area under the curve* (LAUC) measure uses the same approach but instead of measuring the complete area, it takes only the area under the first part of the curve into account. This is the part of the curve which is formed by the first k recommendations.
However, measuring this partial area is not easy due to several problems:

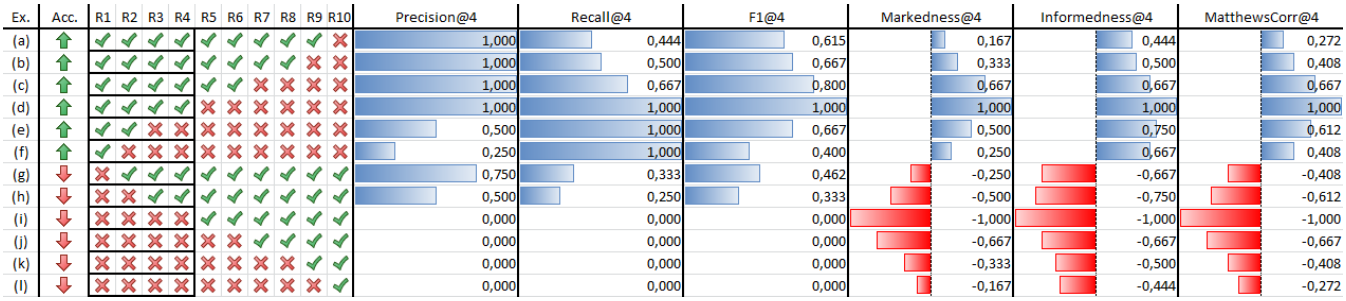
- The measure has to be normalized in a certain way to be comparable.
- The normalized measure should return one if all relevant items are retrieved first by the recommendation algorithm and fit within the recommendation list.
- If the recommendation list contains only relevant items, then the area under the curve is in fact zero. Nevertheless the normalized measure should still return one.
- Relevant items that are retrieved at the end of the list with no irrelevant items following do not add to the area under the limited curve.

A good solution for these problems is to generate just a limited recommendation list that only contains a fixed number of items and to assume that all other relevant items will be distributed uniformly over the rest of the ranking list until all items are retrieved. This means that we calculate the AUC measure for the first part of the ROC curve in the standard way until the first k recommendations have been retrieved. Then we take the end point of the ROC curve



Example	Accuracy	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	AUC	BKendallsT	LAUC4	LBKendallsT4	MAP/GMAP/HMAP/QMAP
(a)	↑	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	1,000	1,000	1,000	1,000	1,000
(b)	→	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗	0,889	0,778	0,889	0,778	0,500
(c)	→	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	0,778	0,556	0,778	0,556	0,333
(d)	→	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	0,667	0,333	0,667	0,333	0,250
(e)	→	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	0,556	0,111	0,278	-0,556	0,200
(f)	→	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	0,444	-0,111	0,278	-0,556	0,167
(g)	→	✗	✗	✗	✗	✗	✗	✓	✗	✗	✗	0,333	-0,333	0,278	-0,556	0,143
(h)	→	✗	✗	✗	✗	✗	✗	✗	✓	✗	✗	0,222	-0,556	0,278	-0,556	0,125
(i)	→	✗	✗	✗	✗	✗	✗	✗	✗	✓	✗	0,111	-0,778	0,278	-0,556	0,111
(j)	↓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	0,000	-1,000	0,278	-0,556	0,100

Figure 2: Varying the position of a single relevant item on a four out of ten recommendation list.



Ex.	Acc.	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	Precision@4	Recall@4	F1@4	Markedness@4	Informedness@4	MatthewsCorr@4
(a)	↑	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	1,000	0,444	0,615	0,167	0,444	0,272
(b)	↑	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	1,000	0,500	0,667	0,333	0,500	0,408
(c)	↑	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	1,000	0,667	0,800	0,667	0,667	0,667
(d)	↑	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗	1,000	1,000	1,000	1,000	1,000	1,000
(e)	↑	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	0,500	1,000	0,667	0,500	0,750	0,612
(f)	↑	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	0,250	1,000	0,400	0,250	0,667	0,408
(g)	↓	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	0,750	0,333	0,462	-0,250	-0,667	-0,408
(h)	↓	✗	✗	✓	✓	✓	✓	✓	✓	✓	✓	0,500	0,250	0,333	-0,500	-0,750	-0,612
(i)	↓	✗	✗	✗	✗	✓	✓	✓	✓	✓	✓	0,000	0,000	0,000	-1,000	-1,000	-1,000
(j)	↓	✗	✗	✗	✗	✗	✗	✓	✓	✓	✓	0,000	0,000	0,000	-0,667	-0,667	-0,667
(k)	↓	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	0,000	0,000	0,000	-0,333	-0,500	-0,408
(l)	↓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	0,000	0,000	0,000	-0,167	-0,444	-0,272

Figure 3: Varying the ratio of relevant items in a four out of ten recommendation list and applying two recommenders with perfect and inverse accuracy.

formed by the first k recommendations and draw a straight line to the upper right corner (see Fig. 5). The area under the curve that is situated to the right of the curve formed by the top-k list thus is a simple trapezoid.
The resulting LAUC measure is very useful for the evaluation of recommender systems that are applied to generate top-k lists of items (cf. Figures 2 and 4):

- The measure returns one if all relevant items are retrieved within the top-k list.
- The measure becomes minimal if no relevant items are retrieved within the top-k list. If all irrelevant items are retrieved first and fit within the top-k list the measure returns zero.
- A top-k list that contains more relevant items will yield a higher score than a list with less relevant items, except if the length of the list is close to the total number of items. In this case the order of relevant and irrelevant items within the recommendation list would have a higher influence on the overall score.
- If a relevant item moves towards the top of the list the measure increases.
- A swap at the top or bottom of the top-k list has the same effect on the measure's value.
- All changes beyond the end of the top-k list are ignored by the measure.

A similar variation of the boolean Kendall's tau, which considers only the top-k recommendations, is another useful and highly correlated metric (e.g. Fig. 2).

6. SOME GUIDELINES FOR CHOOSING A METRIC

In order to choose an appropriate evaluation metric for a given recommendation scenario, it is helpful to answer the following questions:

Is there a distinction between rated and unrated items? If all items are implicitly rated predictive accuracy metrics are not applicable, because there are no unrated items for which we can predict a rating and measure the accuracy.

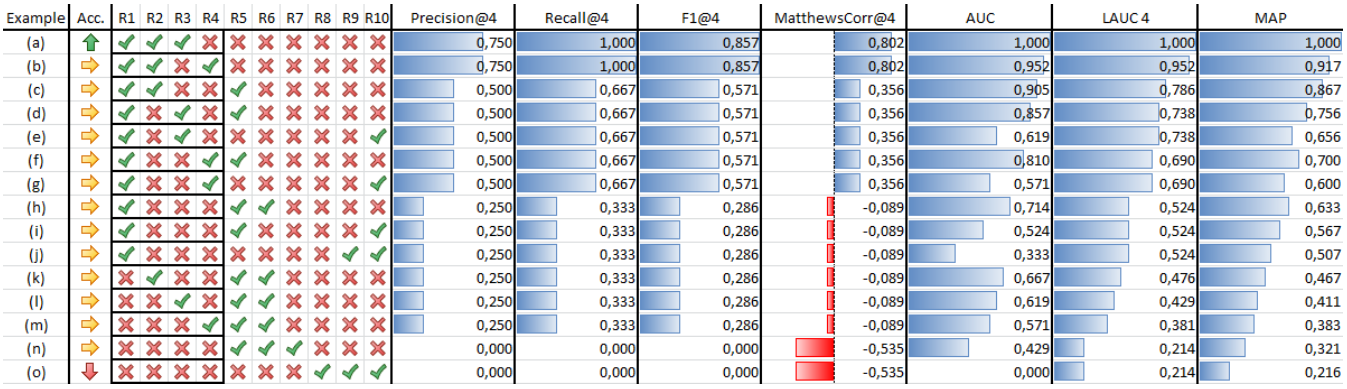
Are items rated on a numerical or a binary scale? A binary rating scale usually suggests a classification or ranking task.

Are users interested in rating predictions or only in top-ranked items? If users only care about top-ranked (or lowest-ranked) items and not about individual rating scores for items this suggests a classification or ranking task. Although an algorithm may use predicted ratings internally, it has to succeed in estimating the top-ranked (or lowest-ranked) items and a higher error on rating predictions for items is acceptable as long as top-ranked (or lowest-ranked) items are identified correctly. Therefore an evaluation should focus on classification or ranking metrics.

Is a limited list of top-ranked items shown? If yes, a metric that measures the overall predictive accuracy or overall ranking accuracy is not appropriate. The exact rating predictions and ranking of other items are irrelevant to users and should not be considered by the metric.

Do the recommended items have or imply an order? Users will usually consider recommendations in a certain order, in particular if many recommendations are shown. If this is the case, basic information retrieval metrics such as precision, recall, markedness and informedness are not sufficient since they ignore the order among the recommended items. A metric that considers the order of recommended items as well is more appropriate for this purpose.

How fast does the user's interest in lower ranked items decay? The metric should reflect the user's decay in interest. MAP and GMAP, for example, emphasize the first recommendations in contrast to the AUC or Kendall's tau measure that weighs swaps in lower and higher ranks in the



Example	Acc.	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	Precision@4	Recall@4	F1@4	MatthewsCorr@4	AUC	LAUC 4	MAP
(a)	↑	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	0,750	1,000	0,857	0,802	1,000	1,000	1,000
(b)	→	✓	✓	✗	✓	✗	✗	✗	✗	✗	✗	0,750	1,000	0,857	0,802	0,952	0,952	0,917
(c)	→	✓	✓	✗	✗	✓	✗	✗	✗	✗	✗	0,500	0,667	0,571	0,356	0,905	0,786	0,867
(d)	→	✓	✗	✓	✗	✓	✗	✗	✗	✗	✗	0,500	0,667	0,571	0,356	0,857	0,738	0,756
(e)	→	✓	✗	✓	✗	✗	✗	✗	✗	✗	✓	0,500	0,667	0,571	0,356	0,619	0,738	0,656
(f)	→	✓	✗	✗	✓	✓	✗	✗	✗	✗	✗	0,500	0,667	0,571	0,356	0,810	0,690	0,700
(g)	→	✓	✗	✗	✓	✗	✗	✗	✗	✗	✓	0,500	0,667	0,571	0,356	0,571	0,690	0,600
(h)	→	✓	✗	✗	✗	✓	✓	✗	✗	✗	✗	0,250	0,333	0,286	-0,089	0,714	0,524	0,633
(i)	→	✓	✗	✗	✗	✓	✗	✗	✗	✗	✓	0,250	0,333	0,286	-0,089	0,524	0,524	0,567
(j)	→	✓	✗	✗	✗	✗	✗	✗	✗	✓	✓	0,250	0,333	0,286	-0,089	0,333	0,524	0,507
(k)	→	✗	✓	✗	✗	✓	✓	✗	✗	✗	✗	0,250	0,333	0,286	-0,089	0,667	0,476	0,467
(l)	→	✗	✗	✓	✗	✓	✓	✗	✗	✗	✗	0,250	0,333	0,286	-0,089	0,619	0,429	0,411
(m)	→	✗	✗	✗	✓	✓	✓	✗	✗	✗	✗	0,250	0,333	0,286	-0,089	0,571	0,381	0,383
(n)	→	✗	✗	✗	✗	✓	✓	✓	✗	✗	✗	0,000	0,000	0,000	-0,535	0,429	0,214	0,321
(o)	↓	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓	0,000	0,000	0,000	-0,535	0,000	0,214	0,216

Figure 4: Varying the position of three relevant items on a four out of ten recommendation list.

same way (cf. Fig. 2). If users hardly look at the ranking of lower ranked items, a classification or ranking error for lower ranked items becomes irrelevant.

7. CONCLUSION

In this paper we analyzed the relation between objectives for applying recommender systems and metrics used for their evaluation. We emphasized the importance of defining a precise goal for the evaluation and discussed how the data used by the recommender as well as its user interface affect the specific task for the recommender. We gave an overview of the three main classes of accuracy related evaluation metrics and discussed their applicability for different types of recommender systems. In order to illustrate the specific advantages and disadvantages of various metrics discussed, we compared them using several informative examples. We proposed to utilize markedness, informedness and Matthews correlation as classification metrics since they are superior to precision, recall and F_1-measure for most purposes. We presented a new variation of the area under the curve measure that is particularly suited for top-k recommendations which are used in many e-commerce applications. This limited area under the curve measure combines classification and ranking accuracy to create a better measure for this purpose. Furthermore, we provided some crisp guidelines that help to choose an appropriate evaluation metric for a specific usage scenario.

8. ACKNOWLEDGMENTS

The work presented in this paper is funded by the European Social Fund and the Free State of Saxony under the grant agreement number 080954843.

9. REFERENCES

[1] M. S. Bansal and D. Fernández-Baca. Computing distances between partial rankings. *Information Processing Letters*, 109(4):238 – 241, 2009.

[2] P. Cremonesi, R. Turrin, E. Lentini, and M. Matteucci. An evaluation methodology for collaborative recommender systems. In *AXMEDIS*, pages 224–231, Washington, DC, USA, 2008.

[3] T. Crook, B. Frasca, R. Kohavi, and R. Longbotham. Seven pitfalls to avoid when running controlled experiments on the web. In *Proceedings of the 15th ACM SIGKDD*, pages 1105–1114, 2009.

[4] F. H. del Olmo and E. Gaudioso. Evaluation of recommender systems: A new approach. *Expert Systems with Applications*, 35(3):790–804, October 2008.

[5] Eighteenth Text REtrieval Conference. Appendix a: Common evaluation measures. In *The Eighteenth Text REtrieval Conference (TREC 2009) Proceedings*, 2009.

[6] R. Fagin, R. Kumar, M. Mahdian, D. Sivakumar, and E. Vee. Comparing partial rankings. *SIAM Journal on Discrete Mathematics*, 20(3):628–648, 2006.

[7] T. Fawcett. An introduction to roc analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.

[8] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4(2):133–151, 2001.

[9] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Transaction on Information Systems*, 22(1):5–53, January 2004.

[10] D. Jannach, M. Zanker, A. Felfernig, and G. Friedich. *Recommender Systems: An Introduction*. Cambridge University Press, 1 edition, 9 2010.

[11] R. Kohavi, R. Longbotham, D. Sommerfield, and R. M. Henne. Controlled experiments on the web: survey and practical guide. *Data Min. Knowl. Discov.*, 18:140–181, February 2009.

[12] J. A. Konstan, S. M. McNee, C.-N. Ziegler, R. Torres, N. Kapoor, and J. Riedl. Lessons on applying automated recommender systems to information-seeking tasks. In *AAAI*, 2006.

[13] D. M. Powers. Evaluation: From precision, recall and f-factor to roc, informedness, markedness and correlation. Technical report, School of Informatics and Engineering, Flinders University Adelaide, South Australia, 2007.

[14] F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors. *Recommender Systems Handbook*. Springer, Berlin, 1st edition, 10 2010.

[15] M. Zanker, M. Bricman, S. Gordea, D. Jannach, and M. Jessenitschnig. Persuasive online-selling in quality and taste domains. In *Electronic Commerce and Web Technologies*, pages 51–60, 2006.