# PROBABILISTIC RECOMMENDATION SYSTEM

## BACKGROUND

Technical Field

[0001] The present disclosure relates to computer processes and systems for generating personalized item recommendations.

Description of the Related Art

[0002] Web sites and other types of interactive systems commonly include recommendation systems for providing personalized recommendations of items stored or represented in a data repository. The recommendations are typically generated based on monitored user activities or behaviors, such as item purchases, item viewing events, item rentals, and/or other types of item selection actions. In some systems, the recommendations are additionally or alternatively based on users' explicit ratings of items.

[0003] Traditional collaborative recommendations processes operate by attempting to match users to other users having similar behaviors or interests. For example, once Users A and B have been matched, items favorably sampled by User A but not yet sampled by User B may be recommended to User B. In contrast, content-based recommendation systems seek to identify items having content (e.g., text) that is similar to the content of items selected by the target user.

[0004] Other recommendation systems use item-to-item similarity mappings to generate the personalized recommendations. The item-to-item mappings may be generated periodically based on computer-detected correlations between the item purchases, item viewing events, or other types of item selection actions of a population of users. Once generated, a dataset of item-to-item mappings may be used to identify and recommend items similar to those already "known" to be of interest to the target user.

[0005] Of course, these and other recommendations methods can be used in combination within a given system.

## SUMMARY OF THE DISCLOSURE

**[0006]** A significant deficiency in existing recommendation systems, including but not limited to recommendation systems of the types described above, is that they ordinarily seek to recommend the "best" (most highly ranked) items to the exclusion of other items representing good recommendation candidates. As a result, the personalized recommendations provided to a given target user tend to be the same each time the user accesses the recommendations. This is especially true if (1) the target user frequently views his or her recommendations, or (2) the target user rarely performs actions that reveal his or her item interests. Thus, currently available recommendation algorithms may provide stale recommendations or recommendations that fail to capture certain interests of the user.

**[0007]** These and other deficiencies are addressed in some embodiments by using probabilistic selection or scoring methods to select specific items to recommend to users. These methods effectively introduce noise into the recommendations process, causing the recommendations presented to the target user to vary in a controlled manner between recommendation access events. Using these methods significantly increases the likelihood that at least some of the items recommended over a sequence of visits will be useful to the target user. This is especially true if the user accesses his or her recommendations relatively frequently (e.g., multiple times per week.) Because the methods are preferably stateless (meaning that the system need not keep track of which items have been recommended to which users), they can advantageously be implemented very efficiently, without placing a large storage burden on the system.

**[0008]** The disclosed processes may be implemented via computer in conjunction with any of a variety of types of recommendation systems, including but not limited to systems that use traditional collaborative recommendations methods, content-based recommendation methods, methods that use item-to-item similarity mappings, or some combination thereof.

**[0009]** One aspect of the disclosure is a recommendation system for selecting items to recommend. The recommendation system comprises a candidate generator operative to generate a ranked set of candidate items to recommend to a target user; a probabilistic scorer operative to probabilistically vary rankings of at least some of the candidate items in

the ranked set to generate a probabilistically-varied ranked set of candidate items; and a candidate filter operative to select, from said probabilistically-varied ranked set, a subset of items to present to the user.

**[0010]** Another aspect of the disclosure is a computer-implemented method of selecting items to recommend. The method comprises generating a ranked set of candidate items to recommend to a target user; probabilistically varying rankings of at least some of the candidate items in the ranked set to generate a probabilistically-varied ranked set of candidate items; and selecting, from said probabilistically-varied ranked set, a subset of items to present to the user.

**[0011]** Another aspect of the disclosure is a computer-implemented method of selecting items to recommend. The method comprises identifying, for a candidate item, a plurality of reasons for recommending the candidate item to a target user, each reason corresponding to a different respective event or set of events recorded in an event history of the user; assigning a score to the candidate item responsive to a value probabilistically selected according to a probability distribution, the probability distribution having at least one characteristic that depends on the number of said reasons; and using the score to determine whether to recommend the item to the target user.

**[0012]** Another aspect of the disclosure is a computer-implemented method comprising: assigning initial scores to candidate items generated as preliminary recommendations to a target user, the initial scores based on a number of reasons corresponding to different respective events recorded in an event history of the user; adding probabilistic noise to at least some of the initial scores; sampling values from a probability distribution that includes the probabilistic noise; and generating adjusted scores for the at least some of the candidate items, the adjusted scores being responsive to the sampled values.

**[0013]** Neither this summary nor the following detailed description purports to define the invention. The invention is defined by the claims.

## BRIEF DESCRIPTION OF THE DRAWINGS

**[0014]** Specific embodiments will now be described with reference to the drawings, which are intended to illustrate and not limit the various features of the invention.

[0015] FIGURE 1 illustrates an embodiment of a probabilistic recommendation system;

[0016] FIGURE 2 illustrates an embodiment of a process for generating item recommendations for a user;

[0017] FIGURE 3 illustrates another embodiment of a process for generating recommendations for a user;

[0018] FIGURE 4 illustrates an embodiment of a process for scoring items;

[0019] FIGURE 5 illustrates example probability distributions of the type used with the process of FIGURE 4;

[0020] FIGURE 6 illustrates a portion of a web page showing an example recommendation interface; and

[0021] FIGURE 7 illustrates one example of how the various recommendation features may be implemented in the context of a web-based electronic catalog system.

## DETAILED DESCRIPTION OF SPECIFIC EMBODIMENTS

[0022] Several different computer-implemented processes will now be described for using probability models to improve item recommendations provided to users. These processes may be embodied individually or in any combination in a multi-user computer system ("system") that implements a recommendation system. For purposes of illustration, the processes are described primarily in the context of a recommendation system that uses item-to-item similarity mappings. As will be recognized, however, the processes may also be used with other types or recommendation systems, including but not limited to traditional collaborative recommendation systems, content-based recommendation systems, and hybrid systems that combine multiple classes of recommendation algorithms.

[0023] For purposes of illustration, the processes are described primarily in the context of a system that recommends catalog items to users of an e-commerce web site. As will be apparent, however, the disclosed processes can also be used to recommend other types of items, such as but not limited to web sites, news articles, blogs, travel destinations, service providers, other users, and events. In addition, the disclosed processes need not be implemented as part of, or in conjunction with, a web site.

[0024] As mentioned above, currently available recommendation services provide a set of theoretically best recommendations to a user. This set of recommendations often remains unchanged each time the user views his or her list of recommendations. Thus, the user often fails to discover the numerous items identified by the system as merely good, but not ideal, recommendation candidates. As a result, the probability that the user will actually discover items of interest is less than optimal. Moreover, the stale nature of the recommendations may lead to infrequent visits to, or even discontinued use of, the recommendation system.

[0025] One option for attempting to solve these problems is to keep track of the items that have been recommended to each user during a given time interval, such as one day. On subsequent visits during this time interval, these items could be filtered out of the recommendation set, causing lower-ranked items to be displayed. Unfortunately, such a method can place a very high storage and processing burden on the system, especially if the user population is large (e.g., tens of millions of users).

[0026] Turning to FIGURE 1, an embodiment of a probabilistic recommendation service or system 100 is shown that addresses the foregoing problems, among others. For example, the various components of the probabilistic recommendation system 100 facilitate generating varied recommendation lists.

[0027] In one embodiment, the process of generating the recommendations is controlled such that the list of recommendations seen by a given user will remain substantially static (unchanged) within a certain time interval, such as one day, one browsing session or one hour, but will otherwise be varied. Thus, for example, if a user accesses or otherwise initiates the generation of recommendations multiple times within such a time interval, the recommendations may remain unchanged. As discussed below, this may be accomplished by seeding the recommendations process the same way during different recommendation generation events, such that the same recommendations can be reproduced without having to cache the recommendation lists for extended periods of time.

[0028] The probabilistic recommendation system 100 of various embodiments includes a source selector 102, a candidate generator 106, a probabilistic scorer 110, and a candidate filter 114, each of which may be implemented as a respective software component.

The candidate generator 106 in this embodiment is a recommendations engine that uses item-to-item similarity mappings 108 to select items to recommend. The item-to-item similarity mappings may be based on observed behavioral correlations (e.g., purchase-based or view-based correlations) as is known in the art, and may include similarity scores representing degrees to which specific pairs of items are related.

[0029] In other embodiments, the candidate generator 106 may use another type of recommendations method to select the candidate items, such as a traditional collaborative recommendations method and/or content-based recommendations method. Regardless of the type of recommendations method/engine used as the candidate generator 106, one or more of the components 102, 110, and 114 in certain embodiments are responsible for probabilistically selecting and/or re-ranking the candidate items for presentation to the user.

[0030] Where item-to-item similarity mappings are used as shown, the candidate generator 106 operates generally by receiving a set of input or "source" items associated with the target user (optionally together with corresponding item weights), and outputting a ranked set of candidate items that are collectively similar to the set of source items. As part of this process, according to some embodiments, the candidate selector can score each candidate item based on both (a) whether it is similar to more than one of the source items, and (b) the degree to which it is similar to each such source item. Specific examples of methods that may be used by the candidate generator to select the candidate items, and to generate the item-to-item similarity mappings, are described in U.S. Patent No. U.S. Patent No. 6,853,982, the disclosure of which is incorporated by reference in its entirety.

[0031] The source selector 102 is responsible for selecting a set of source items to be used to generate personalized recommendations. In the context of an electronic catalog system in which the items are products, the source selector may select these items from, for example, the target user's purchase history, rental history, item viewing history, item ratings profile, wish list, rental queue, shopping cart, tagged items, or any other item collection(s) representing the target users interests or preferences. In the context of a news site in which the items are news articles, the source selector 102 may, for example, select the source items from a collection of items (articles) the user has viewed, forwarded, rated, and/or commented on. Thus, as depicted in FIGURE 1, the source selector 102 uses one or more types of

recorded user activity data 104 associated with the target user (also referred to as the target user's "event history") to select items likely representing the preferences or interests of the target user. As described below, appropriate sampling methods may be used for this purpose to limit the size of the set of source items, and to thereby limit the processing burden placed on the candidate generator/recommendations engine 106.

[0032] The source selector 102 preferably uses probabilistic sampling to select the source items, such that the set of source items used for a given user varies between new recommendation generation events. For example, the source selector may define a probability distribution over the items in the user activity data 104, or collection, and sample these items according to the distribution. Probabilistically sampling items in one embodiment beneficially adds probabilistic "noise" to the set of items, which provides robustness against model errors. Since statistical models are generally not perfect, sampling differently for different visits increases the odds of at least one successful recommendation over multiple visits. Examples of sampling methods are described in more detail below with respect to FIGURE 3. Although probabilistic sampling is used in some embodiments, it is not essential. Further, the source selector may be omitted in some embodiments. In such a case, for example, the candidate generator 106 may select the candidates based on an entire item collection (e.g., purchase history) of the target user.

[0033] In one embodiment, the source selector 102 generates the set of source items by taking samples from multiple different categories of items in the target user's overall collection or event history. For instance, the source selector may sample from any two or more of the following categories: (a) items purchased or rented by the user, (b) items recently viewed by the user, (c) items rated positively by the user, (d) items rated poorly by the user, (e) items on the user's wish list, (f) items in the user's rental queue, (g) un-purchased items added to the user's shopping cart or wish list, and (h) items tagged by the target user. The foregoing are merely examples, and numerous other categories are possible. The source selector 102 may also optionally use item clustering methods to select the source items, as described in U.S. Patent Application No. 11/694,707, filed March 30, 2007, titled "Cluster-Based Assessment of User Interests," the disclosure of which is hereby incorporated by reference in its entirety.

[0034] The source selector 102 provides the selected sources to the candidate generator 106. As described above, the candidate generator 106 in this embodiment uses the database of similar items data 108 to generate a ranked set of candidate items. The candidate generator 106 may also output, for each candidate item, a candidate score generally representing the predicted likelihood that the target user will be interested in that item. In some implementations, the candidate score may be the same as the similarity score.

[0035] Where the source items are selected from multiple item categories as described above, the candidate generator 106 may optionally use different recommendation algorithms for different categories of items. As one example, the candidate generator may use purchase-based item-to-item similarity mappings to select candidate items that are similar to those in the user's purchase history and may use viewing-based item-to-item similarity mappings to select candidate items similar to items the user recently viewed. The item-to-item similarity mappings, including the associated similarity scores, may be stored in one or more tables.

[0036] The candidate generator 106 passes a list of the identified candidate items to the probabilistic scorer 110, which assigns adjusted scores to the candidate items. In one embodiment, the probabilistic scorer 110 thereby re-ranks the candidate items, since the candidate items already have candidate scores. The probabilistic scorer 110 may provide both probabilistic and non-probabilistic (e.g., deterministic) adjusted scores.

[0037] The probabilistic scorer 110 in one embodiment scores items probabilistically by defining a probability distribution for at least some of the candidate items, and by sampling a value from each probability distribution. This value is then used to generate an adjusted score for the candidate item, or alternatively, the value is determined to be the adjusted score for that item. In one embodiment, by adding probabilistic scores, the probabilistic scorer 110 also adds probabilistic "noise" to the set of candidate items. Like the probabilistic noise described above with respect to probabilistic sampling, probabilistic noise added to candidate scores provides a more robust recommendation model. In one embodiment, this noise causes the candidate items to be re-ranked, as the adjusted scores are assigned differently than the candidate scores. In effect, the adjusted scores may be made theoretically worse than the theoretically best candidate scores. Paradoxically,

recommendations based on these adjusted scores may be more valuable to users who frequently check their recommendations list, as the recommendations are varied according to the probability distribution.

**[0038]** The probabilistic scorer 110 passes the score-adjusted candidate items to the candidate filter 114. The candidate filter 114 selects a subset of the score-adjusted candidate items to recommend to the user based on the items' adjusted scores. For example, the candidate filter 114 may select the N most highly scored items to recommend. Alternatively, the subset may be selected probabilistically as above, based on a probabilistic distribution. In one such embodiment, the distribution depends on the adjusted scores, such that items with higher adjusted scores are more likely to be selected than items with lower adjusted scores.

**[0039]** Rather than actually filtering out items from the re-ranked recommendations set, the system may present the entire recommendations set to the user in the re-ranked order. Because this set is typically large (e.g., several thousand items), the system's user interface may allow the user to page or scroll through this re-ranked recommendations set from highest to lowest ranking. Because users commonly do not take the time to scroll or page through the entire set of recommendations, the practical effect is the same as filtering, i.e., the user is only presented with those items falling near the top of the

**[0040]** FIGURE 2 illustrates an embodiment of a process 200 for generating item recommendations for a user. The process 200 is implemented in one embodiment by a probabilistic recommendation system, such as the probabilistic recommendation system 100 of FIGURE 1.

**[0041]** The process 200 begins at 202 by retrieving the target user's relevant activity data or item collection. This step may be performed, for example, by the source selector 102 of FIGURE 1. At 204, the process 202 selects a sample of sources from the activity data. This step may also be performed by the source selector 102. As mentioned above, at least a portion of the sources may be sampled probabilistically. Preferably, at least some of the sources are sampled according to a probability distribution (see FIGURE 3, discussed below). In some embodiments, the source selector 102 may also assign weights to specific source items to cause more weight to be given to some source items than others;

these weights may, for example, depend on the type of user action performed on the item, the user's rating (if any) of the item, and/or the amount of time since the user selected the item.

[0042] The process 200 then proceeds to identify similar (candidate) items corresponding to the sampled sources at 206. This step may be performed, e.g., by the candidate generator 106 of FIGURE 1. In one embodiment, the process 200 obtains candidate items by accessing a similar items table or dataset, which identifies similarity scores between items. In one such embodiment, the process 200 analyzes similarities found in the table to develop candidate scores for the candidate items. Alternatively, the process 200 identifies the candidate items through traditional collaborative filtering, content-based filtering, or another type of recommendation algorithm.

[0043] In one embodiment, the step of identifying candidate items further includes discarding items having candidate scores below a threshold. By discarding candidate items below a particular threshold, weak candidates are weeded out. An example of using thresholds in this manner is described in more detail below with respect to FIGURE 4.

[0044] Thereafter, the process 200 computes initial recommendation scores for the similar/candidate items at 208. This step may be performed, for example, by the probabilistic scorer 114 of FIG. 1. The initial recommendation scores according to one model represent the best theoretical scores for a given candidate item.

[0045] The initial recommendation scores of various embodiments are based on the number of reasons for recommending an item. Reasons in one embodiment include any event by the user (e.g., the purchase of a particular item) or set of events (e.g., the purchase of multiple items) that points to recommending an item in a given similarity pairing. Examples of using reasons to score recommendations are discussed below with respect to FIGURE 3. In some implementations, the initial recommendation scores also depend on the "strengths" of the associated reasons, which in turn depend on the associated similarity scores.

[0046] After generating the initial recommendation scores, in one embodiment the process 200 generates adjusted scores at 210 by adding probabilistic noise to the initial recommendation scores and sampling. Adding probabilistic noise in one implementation includes providing a probability distribution based at least in part on the initial

recommendation scores. Adjusted scores are generated in one embodiment by sampling values from the probability distribution. An example of one such probability distribution and related sampling operations are discussed below with respect to FIGURES 4 and 5. Advantageously, adding probabilistic noise to the initial recommendation scores enables at least some different recommendations to be made by successive recommendation generation events.

[0047] Finally, at 212, the process 200 selects a subset of the items based on the items' adjusted scores to present as recommendations to the user. This step may be performed by the candidate filter 114 described above, in the manner described above.

[0048] FIGURE 3 illustrates a more specific embodiment in which multiple categories of source items are used. The process 300 is implemented in one embodiment by a probabilistic recommendation system, such as the probabilistic recommendation system 100 of FIGURE 1. The process 300 may be implemented in the context of a web-based electronic catalog system (see FIGURE 7), and may be implemented in other contexts, such as in the contexts of electronic news or rental services.

[0049] The process 300 may be invoked in real-time in response to an online action of the user. For example, the user may specifically request recommendations. Alternatively, the process 300 may be invoked in response to an online action of the user that does not directly request the recommendations. For instance, the process 300 may be invoked when the user requests a web page that is configured to display personalized recommendations, possibly in combination with other content. Any of a variety of other methods can be used to initiate the recommendations generation process and to display or otherwise convey the recommendations to the user. For example, the recommendations can automatically be generated periodically and sent to the user by e-mail, in which case the e-mail listing may contain hyperlinks to the product information pages of the recommended items. Further, the personal recommendations could be generated in advance of any request or action by the user, and cached until requested.

[0050] At 302, the process 300 retrieves the target user's relevant activity data or item collection. This step, as well as the following two steps 304 and 306, may be performed by the source selector 102 of FIGURE 1. In one embodiment, the user's activity data may

include any of the data described above with respect to the process 200 (FIGURE 2). The process 300 at 304 then partitions the user activity data or item collection into multiple categories of sources. This step also may be performed, for example, by the source selector 102 of FIGURE 1. In the example embodiment shown, these categories include recent activity, unbought interests, purchase history, and negative feedback. Other categories may be chosen depending on the application.

**[0051]** In one embodiment, the recent activity category includes or consists of items recently viewed and/or rated by the user, the unbought interests category consists of items in the user's wish list or shopping cart, the purchase history category consists of items purchased by the user, and the negative feedback category consists of items the user has explicitly rated poorly, such as by designating as "not interested" or by rating two stars or less on a scale of 1-5 stars (see FIGURE 6). Other types of negative feedback, including implicit negative feedback, may be used to define the negative feedback category.

**[0052]** The example categories shown are used to categorize source items from which recommendations may be generated. More generally, in one embodiment, these categories are implemented as recommender components ("recommenders"). These recommenders include a purchase history recommender, an unbought interests recommender, a recent activity recommender, and a negative feedback recommender, among others. These separate recommenders may be implemented as software components or applications which each separately samples sources, generates similarities (candidate items) for those sources, and scores the candidate items. In one embodiment, these separate recommenders are also implemented with the same underlying logic of the candidate generator/recommendation engine 106.

**[0053]** Intuitively, the categories of recent activity, unbought interests, and purchase history indicate items which the user likes; hence, similar items may be good recommendations. Items in the negative feedback category reflect items which the user does not like. Items similar to those items in the negative feedback category may also be disliked by the user. Negative feedback may be used to improve the quality of recommendations to the user, as is described in further detail below.

[0054] At 306, the process 300 samples N items from each category, where N is an integer such as 64. This step may be performed, for example, by the source selector 102 of FIGURE 1. In one embodiment, the N most recent items from the recent activity category are selected. The most recent items are selected as sources in certain embodiments because they may be a more accurate representation of the user's current interests than older activity, and therefore better recommendations may be generated therefrom. Some embodiments can use the user's past history in addition to or in place of the recent history, since the user may still be interested in items researched in the past.

[0055] Rather than take the N most recent items from the remaining categories, the process 300, in one embodiment, takes a probabilistic sample of items in these categories. Using a probabilistic sample beneficially adds "probabilistic noise" to the recommendation process 300.

[0056] In one embodiment, the process 300 samples the N items without replacement, e.g., by not sampling any member of the item population more than once. In addition, the process 300 of various embodiments samples the items according to a probability distribution. While the probability distribution may take many forms, in one embodiment, a uniform distribution is used for the negative feedback category. In other words, each item in the negative feedback category has an equally likely chance to be sampled. This example distribution assumes that the user dislikes each item in the negative feedback category equally as much. However, in alternative embodiments, different levels of negative feedback may be taken into account in the distribution. For example, items given a one star (worst) rating may be more likely to be selected than items given a higher rating.

[0057] For items in categories other than the recent activity category, the probability distribution in one implementation is dependent on age (as determined from event timestamps in the user's activity data), such that the likelihood of an item being selected decreases with age. Such a distribution can be used, for example, when it is believed that more recent items in the purchase history, unbought interests, and recent activity categories may more accurately reflect the user's current interests. However, older items may still be selected, enabling the recommendation process 300 to generate more varied recommendations. In one embodiment, the age-dependent probability distribution (or

density/mass function) is equal or proportional to 1/sqrt(age), where sqrt() is the square root of a number. The age may be given in hours, days, weeks, or some other unit of time.

**[0058]** At 308, the process 300 identifies similar items corresponding to the sampled sources in each category, such as by accessing item-to-item similarity mappings 108. As mentioned above, different types of similarity mappings or "similarities" may be used to generate similar items for different categories. For example, complementary similarities (e.g., purchase-based similarities) are used to generate similarities for items in the purchase history category, while substitution similarities (e.g., item-viewing-history-based similarities) are used to generate similarities for items in the other categories. Complementary similar items are items which complement each other, such as a digital media player and an accessory for the player. Substitution similar items are items which may be (subjectively or objectively) substituted for each other, such as different models of a digital media player. In some instances, a user may be more interested in complementary similarities for items which the user has already purchased; hence the use of complementary similarities in the purchase history category. In other categories, the user may be more interested in items which are similar to those items that the user has viewed but not purchased, and so substitution similarities are used. As will be recognized, various other types of similarity mappings, including content-based similarity mappings based on an automated analysis of item content, may additionally or alternatively be used.

**[0059]** At 310, the process computes negative feedback scores or values for items in the negative feedback category and category scores for each category in which items appear. This step may be performed by the probabilistic scorer 114 of FIGURE 1. An item in the depicted embodiment that appears in all four categories, for example, would have a negative feedback score and three separate category scores. Certain items may have fewer than four scores. For instance, an item may not have a negative feedback score, or may have fewer than three category scores. In addition, certain items may have only a negative feedback score in some embodiments.

**[0060]** The negative feedback and category scores of certain embodiments can depend on the number of reasons for recommending an item (or for providing negative feedback on an item). The more reasons there are to recommend an item, the higher (or

greater in magnitude, when negative scores are used) the items' negative feedback and/or category scores will tend to be, and vice versa. The negative feedback and category scores do not have to depend on reasons, however. These scores may instead (or additionally) depend on correlations between items or on some other mechanism.

**[0061]** Examples of reasons for recommending items follow. The iPod™ digital media player may be considered substitution similar to the Zune™ digital media player. If a user adds an iPod™ to his wish list, this wish list add is one reason to show the Zune™ to the user as a recommendation. Similarly, if the user also clicks on an iPod™ accessory that is considered similar (e.g., complementary similar) to the Zune™, this click is another reason to recommend the Zune™ to the user.

**[0062]** In another example, if users who dislike the iPod Nano™ also tend to dislike the iPod Shuffle™ digital media player (e.g., as indicated by a similarity table), giving a low rating to the iPod Nano™ may be a reason to include the iPod Shuffle™ in the negative feedback category. Many different types of reasons may be considered for each category, such as adding items to a shopping cart or wish list, viewing items, rating items, tagging items, deselecting items or categories in an existing recommendations list, deselecting items in a shopping cart or wish list, items that the customer has returned, and the like.

**[0063]** In alternative embodiments, the negative feedback and/or category scores do not depend on reasons, but instead depend on similarity (e.g., correlation) scores provided by a similar items table. The negative feedback and/or category scores may also depend on the similarity scores in addition to depending on reasons, such that the strength of a particular reason may be taken into account. For example, a similarity score for a particular similarity pairing may be used as a weight multiplied with a reason to generate a weighted negative feedback or category score.

**[0064]** The process 300 adds probabilistic noise to the category scores at 312. This step may also be performed by the probabilistic scorer 114 of FIGURE 1. This step in one embodiment includes defining probability distributions that depend on the category scores. In one such embodiment, for each category score, a probability distribution is defined in which a parameter of the distribution depends on the category score. For example, the probability distribution may be a normal distribution with parameters mean ($\mu$) and standard

deviation ($\sigma$). Other distributions may be chosen, such as the binomial distribution, Poisson distribution, or the like. In addition, parameters other than the mean and standard deviation may be used.

**[0065]** In one embodiment, because a particular category score depends on the number of reasons for recommending an item, the parameters of a probability distribution depend on the number of reasons. Thus, for example, the mean or standard deviation could equal the number of reasons or could be a function of the number of reasons. One example implementation of using probability distributions with reasons is described in further detail below, with respect to FIGURE 4.

**[0066]** The process 300 samples adjusted category scores from the probabilistic noise added to the category scores at 314. In an embodiment, sampling the adjusted category scores includes probabilistically sampling values according to a probability distribution. Since the adjusted category scores are probabilistically sampled, the adjusted category scores may be worse than the theoretical best scores. Advantageously, probabilistic sampling facilitates providing different recommendations to users which are good, but perhaps not the best, recommendations.

**[0067]** The process 300 at 316 sums the adjusted category scores and the negative feedback score for each item to achieve an overall adjusted score for each item. In an embodiment, the sum of the adjusted category scores is indicative of the number of reasons for a particular product, adjusted with probabilistic noise. The negative feedback score in one embodiment is negative-valued, or rather is subtracted from the sum of the adjusted category scores. The negative feedback score therefore reduces the strength of the overall adjusted score, taking into account that even though there may be reasons for recommending an item, there may be reasons for not recommending an item. In various embodiments, the negative feedback reasons (and hence the negative feedback score) may outweigh the reasons for recommending an item (and hence the sum of the adjusted category scores), and vice versa.

**[0068]** Finally, at 318, the process 300 selects a subset of the items based on the items' overall adjusted scores to present as recommendations to the user. This step may be performed by the candidate filter 114 described above, in the manner described above.

[0069] FIGURE 4 illustrates an embodiment of a process 400 for scoring recommendations. The process 400 is implemented in one embodiment by one or more components of a probabilistic recommendation system, such as the probabilistic recommendation system 100 of FIGURE 1. Likewise, the process 400 may be implemented as using one or more categories or recommenders, such as the recommenders described above, to generate scores. These categories (or recommenders) include purchase history, unbought interests, recent activity, and negative feedback. As described above, additional or fewer categories may also be used.

[0070] At 402, for each category, the process 400 selects candidate items having a candidate score above a threshold. This step may be implemented, for example, by the candidate generator 106, or alternatively, by the probabilistic scorer 114 of FIG. 1. In one embodiment, the threshold used for purchase history is lower than the threshold used for recent activity and unbought interests. In other words, more candidate items will be discarded in such embodiments in the categories of recent activity and unbought interests. This example embodiment assumes that candidate items in the purchase history are more relevant to a user than unbought items or items recently viewed, and this assumption may be different in different embodiments.

[0071] A threshold may also be applied to candidate items in the negative feedback category. This threshold in one embodiment is higher than the thresholds for the other categories, such that fewer negative feedback items are used than items in the other categories. The threshold for negative feedback may be adjusted to increase or decrease the amount of negative feedback provided. In one embodiment, the threshold is set high to provide a less-aggressive level of negative feedback.

[0072] At 404, the process 400 determines whether the negative feedback category is to be evaluated. Steps 404 and the remaining steps 406 through 414 may be implemented, for example, by the probabilistic scorer 114 of FIG. 1. If the negative feedback category is to be evaluated, the process proceeds to 512, where the process 400 computes a negative feedback score for an item as

$$a*R^b \qquad (1)$$

where $R$ is the number of reasons and $a$ and $b$ are constants. This expression (1) illustrates that the negative feedback score for an item is exponentially related to the number of reasons provided for that item. When $b$ is greater than 1 ($b > 1$), the value of (1) increases exponentially as $R$ increases. The expression (1) models the assumption that the more reasons exist for a user to dislike an item, the more likely it is that the user will dislike similar items. In one embodiment, $a$ is less than zero, so that when the adjusted category scores are added in step 414 below, the negative feedback score is effectively subtracted from the sum of the adjusted category scores. The constants $a$ and $b$ may be tuned to increase or decrease the effect of negative feedback on the overall adjusted score for an item.

**[0073]** For each other category in which the item appears, the process 400 at 406 computes a category score

$$R^c \qquad (2)$$

where $R$ is the number of reasons and $c$ is a constant. Like the expression (1), the expression (2) generates category scores that are exponentially related to the number of reasons for recommending an item. In an embodiment, $c$ is also greater than 1. In one specific example, $c$ is 2.

**[0074]** The negative feedback and category scores computed by the expressions (1) and (2) in one embodiment are examples of initial recommendation scores. These scores in various embodiments represent the best theoretical scores for candidate items in a particular category. Thus, if the best theoretical overall score of a candidate item is desired, it can be computed from a sum of the initial recommendation scores.

**[0075]** In an embodiment, the negative feedback score computed by expression (1) is deterministic, e.g., not probabilistic. As a result, no probabilistic noise is added to the negative feedback score. Since negative feedback is sampled in certain embodiments according to a uniform distribution, where all items are assumed to be equally negative, no further probabilistic manipulation is done. However, in alternative embodiments, this score may also be computed probabilistically.

[0076] In contrast, the category scores computed by (2) are manipulated further to produce adjusted category scores, as follows. The process 400 adds noise at 408 by generating a normal distribution for each category score based on the number of reasons, where the normal distribution has mean

$$\mu = R^c \qquad (3)$$

and standard deviation

$$\sigma = (N\text{-}R)^d \qquad (4)$$

where $N$ is the maximum number of reasons and $d$ is a constant. The mean and the standard deviation are parameters of the normal distribution. In an embodiment, these parameters characterize the normal distribution. The characteristics of the normal distribution can therefore depend on the number of reasons $R$.

[0077] In the depicted embodiment, the mean is set to equal the category score for a particular category. Thus, as $R$ increases, the mean increases. The standard deviation, on the other hand, is a function of the maximum number of reasons $N$ minus the number of reasons $R$. Consequently, as $R$ increases, the standard deviation decreases. In one embodiment, $N$ is set to be the theoretical maximum number of possible reasons, for example, 60 reasons. The constant $d$ may be adjusted to increase or decrease the rate at which the standard deviation increases or decreases. In an example embodiment, the value of $d$ is 0.5.

[0078] Example normal distributions 510, 520 (e.g., probability density functions thereof) constructed according to the expressions (3) and (4) are illustrated in FIGURE 5. For convenience, continuous normal distributions are shown graphed on an X axis versus a P(X) axis, where X is a random variable having real values and P(X) is the probability of X. In an alternative embodiment, a discrete normal distribution may be used instead.

[0079] The two example distributions 510, 520 shown have different characteristics. The distribution 510 has a mean value of $R_1^c$, and the distribution 520 has a

mean value of $R_2^c$. The illustrated embodiment shows that $R_2^c$ is greater than $R_1^c$. In general, as the mean of a normal distribution increases, the distribution (or probability density function) is shifted to the right, and vice versa. Thus, the distribution 520 has a higher mean and is therefore farther to the right of the distribution 510. In addition, as the standard deviation increases, a normal distribution generally spreads out. The distribution 510 has a higher standard deviation and is therefore more spread out than the distribution 520.

**[0080]** Referring back to FIGURE 4, the purpose of adjusting the characteristics shown in FIGURE 5 will be described in the context of sampling. At 410, the process 400 samples a value from the normal distribution for each category except the negative feedback category. This value is determined to be the adjusted category score for the item, or in other embodiments, the adjusted category score is derived from this value. This value in one embodiment is a value from the X-axis shown in FIGURE 5.

**[0081]** The narrower the particular normal distribution, the more likely it is that the sampled value will be closer to the mean. As the number of reasons increases, the standard deviation decreases and hence the normal distribution narrows. In addition, as the number of reasons increase, the mean increases and hence the distribution is shifted farther to the right. Since the mean is dependent on the number of reasons (e.g., the best theoretical score for a particular category), the higher this score, the more likely that the sampled value will be the best theoretical score for the particular category (or close thereto). Conversely, if there are fewer reasons for recommending an item, the distribution will be more spread out. The resulting sampled value or adjusted category score will be more likely to vary in such instances.

**[0082]** In one embodiment, the entropy of the distribution is a measure of how spread out the distribution is. The entropy for a normal distribution is

$$ln\ (\sigma * sqrt(2 * pi * e)). \qquad (5)$$

Since the entropy of a distribution depends on the standard deviation ($\sigma$), the higher the standard deviation, the greater the entropy of the distribution. Said another way, the fewer

reasons there are to recommend an item, the greater the entropy of the distribution will be, and there is a greater likelihood that the adjusted category score will be varied for that item.

**[0083]** In one implementation, the sampling operation at step 412 is performed by using a pseudo-random number generator (PRNG). The PRNG generates uniformly-distributed pseudo-random numbers. Thereafter, the output of the PRNG may be passed through an inverse cumulative distribution function for the normal distribution having a desired mean and standard deviation to generate the desired distribution. Alternatively, the output of the PRNG may be passed through a Box-Mueller or ziggurat transform to generate the normal distribution. Since the PRNG is a deterministic function, in this embodiment, sampling the probability distribution is not truly probabilistic. However, for the purposes of this disclosure, this sampling operation is considered probabilistic.

**[0084]** The PRNG of certain embodiments may also be used to generate recommendations statelessly. Stateless functionality may be achieved in one embodiment by seeding the PRNG with the same seed each time the list is generated. Since the output of a PRNG is deterministic, e.g., is predictable based on the seed value, probabilistic sampling done using the PRNG will also be deterministic. Probabilistic sampling therefore results in the same sampled values each time the list is generated with the same seed. In addition, a PRNG may be used similarly to implement the probabilistic sampling of the source selector described above. Thus, recommendations may not be changed until the seed of the PRNG is changed.

**[0085]** Various embodiments of the process 400 contemplate using this stateless property of the PRNG to adjust the amount of variation in a user's recommendation list to give a certain amount of overlap between new recommendation generation events. Adjusting this variation beneficially reduces potential user confusion, which might occur if the recommendations change rapidly. Adjusting the variation also reduces potential broken pagination problems, which may occur with lists spanning multiple web pages. For these lists, broken pagination can occur when a user sees recommendations from an earlier page on a subsequent page, due to the list regenerating on each page access. As described above, variation overlap may be implemented such that a set of recommendations does not change for a certain period of time (e.g., 24 hours) regardless of the number of recommendation list

accesses by the user. After this period of time, a different seed is provided to the PRNG to generate new recommendations.

**[0086]** In one embodiment, the amount of variation in the recommendation list between new recommendation events is tuned by adjusting the variance of the normal distribution, which is the standard deviation squared ($\sigma^2$). For example, if the variance equals 0, there may be 100% overlap between visits. The constants c and d may be adjusted to increase the variance to create the desired overlap. One approach is to auto-parameterize the variance according to a specific user's frequency of return. Using the variance, a balance may be achieved between the number of identical recommendations over a period of time and the quality of recommendations. In one embodiment, for example, these parameters are adjusted to preferably create a 5 to 20% overlap between new recommendation generation events. More preferably, the parameters are adjusted to create a 10 to 15% overlap. Other ranges may be chosen in different embodiments.

**[0087]** Finally, the process 400 at 414 sums the adjusted category scores and the negative feedback score for each item to achieve an overall adjusted score for each item. The overall adjusted scores may be used to rank the items and choose items for recommendation. Thus, as a result of certain embodiments of the process 400, the items' rankings are probabilistically varied according to the number of reasons for recommending the item. More generally, the process 400 of certain embodiments probabilistically re-ranks items according to the category scores for the items.

**[0088]** FIGURE 6 illustrates a portion of a web page showing an example recommendation interface. The example recommendations page 600 displays recommendations for a user. The recommendations page 600 comprises various details about the listed products 610 (four products shown), and includes buttons for adding each product to an electronic shopping cart or wish list. In this example, the recommendation page 600 also includes a set of controls 612, 614 for rating, indicating ownership of, and indicating a lack of interest in, each listed product 610. The recommendations service may use this information to improve the recommendations it makes. In an embodiment, this process is stateless, such that no information about which items have been recommended to which users needs be retained. A

refresh option 618 allows a user to see an updated list of recommendations, which may be updated when the user adjusts the controls 612, 614.

[0089] FIGURE 7 illustrates how the various features described above may be implemented in the context of a web-based system that provides functionality for users to browse and purchase items from an electronic catalog. As will be recognized, the various features of the disclosed embodiments can also be implemented in other types of systems, including, e.g., music download systems that recommend music titles based on users' music download histories, video rental sites that recommend video/DVD titles based on users video rental histories, news sites that recommend news articles based on users' news browsing histories, browser toolbar based systems that recommend web sites and other network resources based on users' web browsing histories, web services for providing recommendations, checkout/wallet services that enable users to pay for goods from various participating merchants, and Internet advertising systems.

[0090] The system shown in FIGURE 7 includes one or more web server machines 701 that generate and serve pages of a host web site in response to page requests from user computing devices 702. The web servers 701 provide user access to a catalog of items represented in a database 708 or collection of databases. The items preferably include or consist of items that may be purchased via the web site (e.g., book, music and video titles in physical or downloadable form; consumer electronics products; household appliances; magazine and other subscriptions, etc.). The database 708 also stores data regarding how the items are arranged within a hierarchical browse structure. Data regarding the catalog items and the browse structure is accessible via a catalog service 706, which may be implemented as a web service.

[0091] The system also includes a data repository 704 (e.g., one or more databases) that stores various types of user data, including identifiers of the items in each user's collection. For example, the data repository 704 may store users' purchase histories, item viewing histories, item ratings, and item tags. The purchase histories and item viewing histories may be stored as lists of item identifiers together with associated event timestamps. The various types of user data may be accessible to other components of the system via a data service (not shown), which may be implemented as a web service.

[0092] The system also includes a recommendation service 710 that generates recommendations in real time in response to requests from users. The recommendation service 710 includes a probabilistic recommendation system 700 that implements some or all of the probabilistic scoring features described herein. Separate executable components may optionally be provided for performing, e.g., source selection, candidate generation, probabilistic scoring, and candidate filtering; these components may all run on a single computer or on separate computers.

[0093] The probabilistic recommendation system 700 operates as described above with respect to FIGURES 1 through 5. More specifically, the recommendations service 710 may use one or more "similar items" tables/datasets 708 to look up items that are similar or related to the source items, together with associated data values indicating the strengths of such relationships. The similar items table(s) 708 may be generated off-line by an item association mining component 718 that analyzes users' purchase histories, item viewing histories, or some other type of user activity data, and detects and quantifies behavior-based associations between specific items. For instance, if purchase histories are used, item A may be mapped to item B in a purchase-based similar items table 708 if a relatively large number of the users who purchased item A also purchased item B. Other types of recommendation engines, including recommendation engines that do not use item-to-item mappings, may also be used.

[0094] The electronic catalog system may also include services for handling various other types of tasks, such as user authentication, transaction processing, search query processing, storing user-assigned tags and ratings, processing of user-submitted sales listings, etc.

[0095] The web servers 701 use a data repository of web page templates 705 to dynamically generate web pages in response to browser requests. The templates directly or indirectly specify the service calls that are made to the services to, e.g., request data needed to generate the requested page. For instance, an appropriate template may be provided for generating recommendation pages of the type shown in FIGURE 6, along with various other pages of the site.

[0096] When a user clicks on a link for viewing recommendations, a web server 701 requests recommendations for the user from the recommendations service 710. The recommendations service 710 then uses all or a portion of the user's activity data or collection (typically depending upon the context of the user's request) to generate the recommendations. As part of this process, the recommendations service 710 may use some or all of the candidate generation, probabilistic scoring, and candidate filtering operations described above. Regardless of which of these features is/are used, the resulting list of recommended items (or a portion of this list) is incorporated into the requested web page for transmission to the user's browser/computer 702.

[0097] The services and other application components 700, 706, 710, and 718 shown in FIGURE 7 may be implemented in software code modules executed by any number of general purpose computers or processors, with different services optionally but not necessarily implemented on different machines interconnected by a network. The code modules may be stored in any type or types of computer storage, such as hard disk drives and solid state memory devices. The various data repositories 704, 705, 708 may similarly be implemented using any type of computer storage, and may be implemented using databases, flat files, or any other type of computer storage architecture.

[0098] Each of the processes and algorithms described above may be embodied in, and fully automated by, code modules executed by one or more computers or computer processors. The code modules may be stored on any type of computer-readable medium or computer storage device. The processes and algorithms may also be implemented partially or wholly in application-specific circuitry. The results of the disclosed processes and process steps may be stored, persistently or otherwise, in any type of computer storage, such as those mentioned above.

[0099] The various features and processes described above may be used independently of one another, or may be combined in various ways. All possible combinations are intended to fall within the scope of this disclosure.

[0100] Although this invention has been described in terms of certain preferred embodiments and applications, other embodiments and applications that are apparent to those of ordinary skill in the art, including embodiments and applications that do not provide all of

the benefits described herein, are also within the scope of this invention. The scope of the invention is defined only by the claims, which are intended to be construed without reference to any definitions that may be explicitly or implicitly included in any of the incorporated-by-reference materials.

WHAT IS CLAIMED IS:

1. A recommendation system for selecting items to recommend, the recommendation system comprising:

a candidate generator operative to generate a ranked set of candidate items to recommend to a target user;

a probabilistic scorer operative to probabilistically vary rankings of at least some of the candidate items in the ranked set to generate a probabilistically-varied ranked set of candidate items; and

a candidate filter operative to select, from said probabilistically-varied ranked set, a subset of items to present to the user.

2. The recommendation system of Claim 1, wherein the degree to which the probabilistic scorer probabilistically varies the ranking of a candidate item depends on a number of identified reasons for recommending the candidate item to the target user.

3. The recommendation system of Claim 2, wherein probabilistic scorer causes candidate items having relatively low numbers of reasons to be probabilistically varied to a greater degree than candidate items having relatively high numbers of reasons.

4. A computer-implemented method of selecting items to recommend, the method comprising:

generating a ranked set of candidate items to recommend to a target user;

probabilistically varying rankings of at least some of the candidate items in the ranked set to generate a probabilistically-varied ranked set of candidate items; and

selecting, from said probabilistically-varied ranked set, a subset of items to present to the user.

5. The computer-implemented method of Claim 4, wherein selecting, from said probabilistically-varied ranked set, a subset of items to present to the user comprises selecting a most-highly ranked subset of items.

6. The computer-implemented method of Claim 4, wherein at least some of the candidate items are associated with one or more identified reasons for being recommended to the target user.

7. The computer-implemented method of Claim 6, wherein the degree to which the ranking of a candidate item is probabilistically varied depends on the number of identified reasons for recommending the candidate item to the target user.

8. The computer-implemented method of Claim 7, wherein probabilistically varying the rankings comprises causing candidate items having relatively low numbers of reasons to be probabilistically varied to a greater degree than candidate items having relatively high numbers of reasons.

9. The computer-implemented method of Claim 4, wherein probabilistically varying the rankings comprises causing candidate items having relatively low category scores to be probabilistically varied to a greater degree than candidate items having relatively high category scores.

10. The computer-implemented method of Claim 4, wherein probabilistically varying the rankings comprises generating values for at least a portion of the candidate items and adding probabilistic noise to the values.

11. The computer-implemented method of Claim 10, wherein adding probabilistic noise to the values comprises generating a probability distribution corresponding to each value, wherein the characteristics of the probability distribution depend on the corresponding value.

12. The computer-implemented method of Claim 11, further comprising sampling values from each probability distribution to generate a score for said at least a portion of the candidate items, wherein the score is responsive to the sampled value.

13. The computer-implemented method of Claim 4, wherein probabilistically varying rankings of at least some of the candidate items in the ranked set further comprises probabilistically varying the rankings to achieve an overlap of 5 to 20 percent of recommended items between recommendation generation events.

14. A computer-implemented method of selecting items to recommend, the method comprising:

identifying, for a candidate item, a plurality of reasons for recommending the candidate item to a target user, each reason corresponding to a different respective event or set of events recorded in an event history of the user;

assigning a score to the candidate item responsive to a value probabilistically selected according to a probability distribution, the probability distribution having at least one characteristic that depends on the number of said reasons; and

using the score to determine whether to recommend the item to the target user.

15. The computer-implemented method of Claim 14, wherein the at least one characteristic of the probability distribution comprises entropy.

16. The computer-implemented method of Claim 15, wherein the entropy of the probability distribution decreases as the number of reasons increases.

17. The computer-implemented method of Claim 14, wherein the probability distribution is a normal distribution.

18. The computer-implemented method of Claim 17, wherein the at least one characteristic of the probability distribution comprises a mean configured to depend on the number of reasons, and wherein the score is responsive to the mean.

19. The computer-implemented method of Claim 17, wherein the at least one characteristic of the probability distribution comprises a standard deviation configured to depend on the number of reasons, and wherein the score is responsive to the standard deviation.

20. The computer-implemented method of Claim 14, wherein the score depends upon strength values associated with specific reasons.

21. The computer-implemented method of Claim 14, further comprising assigning a negative feedback score to the candidate item, wherein the negative feedback score is dependent on the number of said reasons.

22. The computer-implemented method of Claim 21, wherein assigning a score to the candidate item further comprises adding the one or more category scores and the negative feedback score.

23. The computer-implemented method of Claim 14, wherein assigning a score to the candidate item responsive to a value probabilistically selected according to a probability distribution further comprises assigning the score such that an overlap of 5 to 20 percent of recommended items between recommendation generation events is achieved.

24. A computer-implemented method, the method comprising:

assigning initial scores to candidate items generated as preliminary recommendations to a target user, the initial scores based on a number of reasons corresponding to different respective events recorded in an event history of the user;

adding probabilistic noise to at least some of the initial scores;

sampling values from a probability distribution that includes the probabilistic noise; and

generating adjusted scores for the at least some of the candidate items, the adjusted scores being responsive to the sampled values.

25. The computer-implemented method of Claim 24, wherein sampling values from a probability distribution that includes the probabilistic noise comprises probabilistically selecting values according to a probability distribution having at least one characteristic that depends on the number of said reasons.

26. The computer-implemented method of Claim 24, wherein the probability distribution is a normal distribution.

27. The computer-implemented method of Claim 26, wherein a mean of the normal distribution depends on the number of reasons, and wherein the score is responsive to the mean.

28. The computer-implemented method of Claim 26, wherein a standard deviation of the normal distribution depends on the number of reasons, and wherein the score is responsive to the standard deviation.

29. The computer-implemented method of Claim 24, wherein generating adjusted scores for at least some of the candidate items further comprises determining a negative feedback score dependent on the number of reasons.

30. The computer-implemented method of Claim 24, wherein generating adjusted scores for the at least some of the candidate items further comprises generating adjusted scores such that an overlap of 5 to 20 percent of recommended items between recommendation generation events is achieved.