Chapter 6
News Recommendation in Real-Time

Benjamin Kille, Andreas Lommatzsch and Torben Brodt

Abstract Recommender systems support users facing information overload situations. Typically, such situations arise as users have to choose between an immense number of alternatives. Examples include deciding what songs to listen to, what movies to watch, and what news article to read. In this chapter, we outline the case of suggesting news articles. This task entails a number of challenges. First, news collections do not remain relevant unlike movies or songs. Users continue to request novel contents. Second, users avoid creating consistent profiles thus reject login procedures. Third, requests arrive in enormous streams. Having short consumption times, users quickly request the next article to read. Handling these challenges requires adaptations to existing recommendation strategies as well as developing novel ones.

Coffee Time

Suzanne shivered while looking out of the window. It was one of these cold December afternoons where you just want to stay at home, enjoy a cup of hot coffee, and relax next to the fireplace in the living room. "I hope Laura and Linda will make it on time today" she thought, a little worried about the safety of her friends. It was not the first time for them that they'd miss their little get-together—or "gossip club," as her husband Steve used to call it. She always complained when he said that, but actually, she secretly had to admit that he wasn't too far from the truth in the analysis of her circle of friends. They really were gossip! Especially Laura seemed to know everything about everyone in the neighborhood and was more than motivated to share

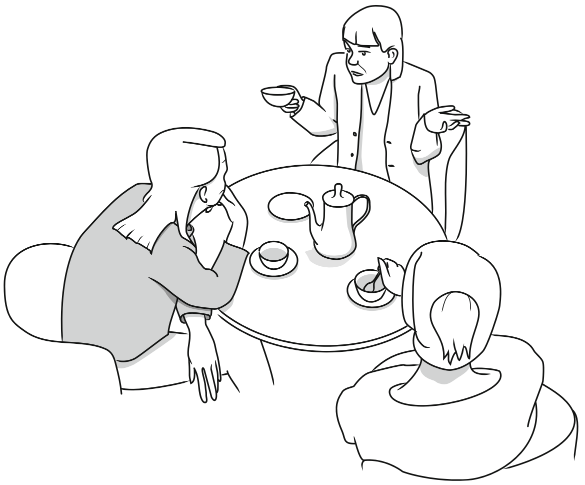
B. Kille (✉) · A. Lommatzsch
Technische Universität Berlin, Berlin, Germany
e-mail: benjamin.kille@dai-labor.de

A. Lommatzsch
e-mail: andreas.lommatzsch@dai-labor.de

T. Brodt
Plista GmbH, Berlin, Germany
e-mail: tb@plista.com

F. Hopfgartner (ed.), *Smart Information Systems*, Advances in Computer Vision and Pattern Recognition, DOI 10.1007/978-3-319-14178-7_6

her knowledge with anyone who couldn't run away on time. Finally, she saw Laura's car coming around the corner, Linda sitting right next to her. Quickly, Suzanne rushed to the door to welcome her two friends.



"Suzanne, I wanted to ask you…," started Laura while they enjoyed their first piece of cheesecake. "I heard that you no longer get your newspaper delivered?" Suzanne had to smile. That's Laura at her best! Always well informed about almost anything. "Yeah, that's true," she replied. "We realized that we read most of the news that we are interested in online anyway. So by the time we can read it in the newspaper, we could have read it online already." "What about local news?", Linda asked getting curious. "Can you get even those online?" "Sure, there are plenty of websites dedicated to all varieties of news. I have even found a website providing news about gardening. You know how much I like to rearrange my garden", Suzanne replied and pointed out of the window. "How do you find articles that are relevant within the masses of contents published online though?", Laura asked. Suzanne kept silent for a while. "That is actually a hard task. Usually, I just browse the home page of a selection of news portals until something catches my attention." Laura raised her hand, indicating that she wanted to say something after swallowing the piece of the cheesecake which she had just lifted from her plate. "I prefer the old-fashioned newspaper," she said after her throat was empty again. "You get a piece of all categories of importance. You do not miss any substantial story." Linda nodded her head. "That may be true", she added, "but I could certainly relinquish reading all those sport and business related articles." "Think of all the trees that had been logged for nothing!" Suzanne argued, causing her friends to roll their eyes in amusement. "I do prefer to have a newspaper in my hands which I can flip myself" Laura mentioned. "With the current generation of tablet computers you can almost get the same feeling. And you do consume less paper." Suzanne argued. "And you can actually search for terms and thus avoid to parse the text manually."

At this moment, the deafening noise of the teapot whistle interrupted their little chat. Suzanne went to the kitchen and returned with three cups of tea. Laura was the first to continue talking. “How are Carl and Carla doing?” she asked Suzanne. “Well, Carla just started her internship at that newspaper office. She is doing great. On the other hand, we are a bit worried about Carl. He seems to have a hard time in school.” “I am sorry to hear that,” Linda intervened. “I have read that the government is planning to revise the lessons plan. Perhaps, the lectures will become less difficult in the next term.” she added. “Where did you read that?” Laura was getting curious. “Actually, I read that online. While I was browsing my favorite news portal, I stumbled upon a section named *suggested readings*. There I found the article” Linda replied. Suzanne’s curiosity began to stir. “How exactly does this work? I mean, how does the news portal select articles that it deems relevant for you?” she asked Linda. “I have no idea.” Linda answered. “There were also four or five other suggestions which I found quite uninteresting” she added. “I think, I have seen this type of suggestions on different news portals.” Laura told the other two. “I could not find any article of interest though” she noted. The three started to think about occasions where they had seen similar services. Suzanne was the first to notice that her favorite online shop did offer a list of recommended items. The women agreed that this service had been in place for a couple of years at least. Conversely, suggested reading on news portals appeared to them as comparably new features. “It has to be much harder to suggest news compared to products in an online shop” Laura claimed. “Why is that?” Linda asked. “You have to consider that the online shop knows who you are after you bought something. As you log in you identify yourself. Conversely, the news portal does not know much about you, does it?” Laura explained. “You are right” Linda agreed. The three women started to discuss how they would suggest articles to one another. “You cannot go wrong suggesting Linda articles about animals” Suzanne claimed. All three started to laugh cheerfully. “I would suggest all articles about the latest gardening trends to you, Suzanne” Linda returned her joke. The women realized that they knew each other well after all these years.

6.1 Introduction

News reading behavior is considerably shifting toward online consumption. More and more users appreciate the advantages of reading news online. Users enjoy instantaneous access to breaking news. Conversely, old-fashioned newspapers delay access to breaking news due to the printing and distributing process. In addition, newspapers dictate the selection, quantity, and source of news which they comprise. Editors decide about which events to include, their articles’ position in the layout, and what space they may cover. However, anxiously editors prepare their newspapers, users’ preferences vary too extensively to consider the result a perfect fit for them. Users may request more information about certain events that exceed the available space. Further, users may enjoy reading articles enlighting events from different perspectives. Newspapers rarely publish several articles about an individual event. Additionally, users may prefer the writing styles, content focus, or presentation of different journalists and newspapers. For instance, users may prefer reading local news from a residential news source. Mainstream news sources may not cover their local events at all. Simultaneously, users may read sports-related news rather from mainstream news sources as they can afford journalists to travel to these events. Hence, users require services which online news portals provide in contrast to their analogous counterparts.

As a consequence thereof, users increasingly face the information overload problem. Recommender systems have established as the suited means to overcome information overload. They filter available items thus reduce the decision problem significantly. Users avoid to browse large sets of items. Instead, recommender systems provide a small fraction of items which they deem most relevant to the user at hand. Research has focused on recommender systems in terms of preference elicitation methods [55]. In the context of news, users have rather preferences for latent concepts than actual items. Recommendations of products such as movies, songs, or books differ from news article suggestions in this aspect. In the following, we present a recommendation method that allows dealing with requirements inherent to news recommendations. These requirements include dynamic item collections, incomplete user profiles, and differences between individual news portals.

Dynamic item collections refer to the rates at which items either enter or exit the systems. Editors add novel news items as they emerge to provide readers with information about recent events. On the other hand, news articles decrease in relevance over time as more and more users become aware of them. News collections exhibit much higher addition/deletion rates compared to collections of movies or songs. Users may want to reconsume their favorite movies or songs. Contrarily, readers will seldom read old news articles again.

Recommender systems' quality depends on how well their models reflect user preferences. Typically, system operators require users to create explicit profiles by design. Thus, they are able to feed preference directly linked to a specific user. Contrarily, news portals do rarely require explicit profiles to be created. Supposedly, readers are unwilling to spend time creating profiles. Privacy concerns represent another reason keeping users from providing their personal information. News portal operators tend to identify their users with session identifiers. However carefully they monitor session identifiers, user profiles may contain errors. We mention three kinds of such errors. First, readers may use several devices to consume news items. For instance, they may read news on their tablets as well as their desktop computers. News portal operators will struggle as they seek to merge these profiles based on session keys. Second, readers may share their computers with other. For instance, a couple which lives together might use the same computer for browsing news. Thus, a profile emerges which captures not one but two preferences. Third, users may block the session monitoring due to privacy concerns. Thus, the system operators monitor various users which they cannot differentiate.

Having spent time and resources to build a user profile, users expect to benefit of adequate recommendations. Conversely, users may consider continuing using the system and not abort. On the other hand, news readers behave differently. Users may choose to frequent several news portals. Consequently, users' profiles scatter over various domains. Incomplete profiles impede creating suggestions. The less information is available about the user at hand, the harder it becomes to select relevant readings.

A set of challenges arises to news recommender systems based on the specific characteristics of news. *What news item reflects a certain latent interest best?* We discuss strategies to deal with the dynamics of news. *How to link interactions to users profiles split over a variety of news portals?* We present ways to construct user profiles representing preferences that allow to provide relevant suggested readings. *How to handle the velocity, veracity, variety, and volume of large streams of interactions of popular news portals?* We elaborate on techniques to cope with big data requirements in the context of news recommendation.

This chapter is structured as follows. Section 6.2 introduces previous research on news article recommendations. Subsequently, we present specifics of our use case in Sect. 6.3. These characteristics include technicalities and requirements as well as system particularities. In Sect. 6.4, we show results of observing how users consume news online. We cover essential aspects including sparsity, popularity bias, as well as contextual factors. Section 6.5 illustrates recommendation algorithms which have been applied to a variety of recommendation problems. We discuss how individual methods suit news recommendation. Likewise, we highlight aspects impeding the application of certain methods. Section 6.6 details design choices faced as we seek to evaluate the performance of recommendation algorithms. Finally, we conclude and give an outlook to future research directions in Sect. 6.7.

6.2 Related Work

News portals have evidentially changed the way we consume news. This section presents related research dedicated to support users consuming news. Billsus and Pazzani [8] refer to four types of systems which have developed to support us consuming news. First, they introduce systems which enable personalized access to news. The personalization manifests as news portals present varying news items depending on individual preferences. News recommender systems rank among this kind of systems. Second, Billsus and Pazzani list adaptive news navigation systems. These systems control how news stories link together. Ideally, they reduce users' efforts to turn back to home pages before continuing reading. Third, Billsus and Pazzani mention contextualized news systems. These systems present their contents depending on users' current contexts. Context includes aspects such as location, time, and current interests. Finally, they introduce news aggregation systems. These systems take collections of news articles and automatically extract the very essential information. We focus particularly on systems recommending news articles. These systems became invaluable supportive to online news readers as more and more news became available. This growth induced an information overload problem. Recommender systems represent a specific kind of information filter. Information retrieval systems filter information contained in document collections having received a query [39]. In contrast, recommender systems attempt to learn preferences from previous interactions to avoid explicit querying. This feature becomes particularly helpful in situations where users lack a defined information need. Instead, users require systems to provide information that will likely be of interest to them.

Researchers have proposed a variety of ideas to carry out the selection process. The ideas range from rather simplistic approaches to highly sophisticated methods carrying a plethora of parameters with them. Trivial methods include randomly recommending items as well as suggesting items based on their popularity. Two paradigms cover a large fraction of the more advanced methods: collaborative filtering and content-based filtering. The former builds on the idea of leveraging other users' preferences to provide recommendations. The latter strives to discover items whose contents share similarities with items users have liked in the past. A comprehensive discussion of both exceeds our scope. Still, we present a selection of ideas tailored for the news domain. We refer readers interested in recommender systems in general to [1, 43, 52].

Proposed news recommendation approaches either utilize other users' interactions with news portals, (possibly enriched) news contents, or both. Thus, we recover both paradigms of regular recommender systems.

Liu et al. [40] introduce a Bayesian framework to allow hybrid recommendations of news articles to users in a personalized fashion. They showed that considering content features increased news consumption compared to a collaborative filtering baseline. Li et al. [38] model news recommendation as a contextual-bandit problem. They show that replaying recorded interactions enables researchers to consistently evaluate their recommendation methods. They provide the theoretical foundations for the unbiasedness of such a methodology. De Francisci et al. [21] make use of three kind of inputs to their news recommendation system. First, they consider interactions in terms of clicks. Second, they extract contents from micro-blogs. Finally, they consider the social relation between the micro-blogging service's users. They represent the problem as learning to rank task. The proposed method considers all three factors to adjust the ranking of news articles for target users. Son et al. [57] propose to consider users' current locations to improve the news item selection process. Additionally, the authors utilize semantic data to enrich the representations of users' interests and locations' relevant concepts. Capelle et al. [14] investigate whether semantic similarities between named entities in news articles can be leveraged to improve recommendation quality. The method requires name entity recognition as a preprocessing step. Bogers and van den Bosch [9] propose a probabilistic framework to provide better news suggestions. Their work looks at the problem from an information retrieval perspective. They analyze the impact of the selected relevance model on the recommendation quality. Li et al. [37] propose a personalized news recommendation framework. Their work emphasizes the issues arising due to the dynamics inherent in item collections. Consequently, they propose to represent the recommendation task as a contextual bandit problem. Li and Li [35] propose to leverage co-occurring interactions to improve news recommendations. Their method models relations between concepts in news texts as hypergraphs. The approach considers both user behaviors and contents. Garcin et al. [25] investigate whether context trees enable news recommender systems to provide relevant news items to anonymous users. Their method builds context trees based on observed user behaviors. The authors pay particular attention toward recommending novel and diverse items. Cantador et al. [12, 13] leverage two kinds of data to select more relevant news items. On the one hand, they derive semantic concepts from an existing ontology. This represents a content-based approach. On the other hand, they use contextual features to better account for recent trends. Das et al. [17] present insights from a large-scale news recommendation system operated by Google. Their work emphasizes the requirements which operating recommender systems face. They discuss how algorithms including MinHash and probabilistic latent semantic indexing enable news recommender systems to apply the collaborative filtering paradigm in large-scale settings. Montes-Garcia et al. [46] propose a news recommender system tailored specifically towards the needs of journalists. Their approach pays particular attention toward personal preferences as well as contextual factors. Gao et al. [24] analyze how well micro-blogs support news recommendation by indicating trends in an early stage. They investigate the trade-off between popular news and personal tastes. Phelan et al. [47] present a socially-driven news recommendation service which extracts data from micro-blogging services as well as RSS feeds. The authors compare whether RSS contents, micro-blog contents, or a combination of both lets news recommendation services select the most relevant news items. Kompan and Bielikova [32] present a news recommender system based on content similarities. The authors discuss the importance of low computational complexity induced by short response times. Lv et al. [44] propose a method utilizing a variety of factors to estimate articles' relatedness. These factors include relevance, novelty, connectivity, and transition smoothness. For a detailed survey on personalized news recommendation algorithms, we refer the reader to Li et al. [36].

Evaluating recommendation algorithms depends on a variety of factors. First, we have to define the recommendation algorithm's objective. This entails specifying the notion of a good recommendation. At first, this may appear trivial. Researchers have come up with several different specifications. Recommender systems attained increased attention with the "Netflix Prize" challenge [7]. This competitions seeked to reduce the error rate when predicting users' preferences for movies. The organizers decided to use the root mean squared error to compensate for larger deviations. Subsequently, researchers continued to optimize rating prediction scenarios [18, 29, 33, 50, 53, 58]. In addition, researchers started to define recommender systems as ranking mechanisms. They argued that recommender systems ought to rank items according the user preferences. Accurately estimated preferences yield such rankings. Still, they do not constitute an essential input as long as algorithms keep the pairwise order of preferences. Optimizing metrics including normalized discounted cumulative gain (nDCG) and mean reciprocal rank (MRR) provide such rankings [41, 51, 56, 60]. Some researchers argue that users refute to consider all available items. Instead, users limit their attention toward few most relevant items. We find evaluation criteria accounting for these desires in the field of information retrieval. Hereby, systems cut rankings at a pre-defined position. We measure recommendation quality in terms of precision, recall, or a combination thereof [4, 16, 19, 30, 49, 61]. In addition, evaluations may consider further factors determining systems' qualities.

These factors include diversity [34], novelty [60], stability [3], and scalability [5, 54, 58]. Having decided which criteria to optimize, we face another design choice: Do we rely on recorded data or do we aim to interactively conduct experiments with users [27, 55]? Both alternatives have advantages. Offline experiments entail little costs. Additionally, other researchers can reproduce results as the data used for evaluation is fixed. Conversely, conducting experiments with actual users may better reflect the actual use-case. User studies as well as deploying novel algorithms into existing recommender systems represent two alternatives for online experimentation.

Related work covers a wide spectrum of news recommendation's aspects. Most recent works focus on two of these aspects. First, researchers seek to improve recommendation quality by using additional data sources. These sources provide textual descriptions, interaction with users, and social relations. We still cannot satisfyingly tell how to determine additional data's value in advance. Second, research investigates potentials to algorithmically improve recommendations. Due to inherent requirements, we struggle to transport established, sophisticated methods to the news domain. Besides these two major aspects, researchers seek to discover better evaluation protocols along with means to deal with the real-time character of news recommendation.

6.3 The Plista Case

We introduced recommending news articles as a challenge for science and industry in Sect. 6.1. Subsequently, we outlined methods enabling news portals to suggest news articles in Sect. 6.2. Both occurred on a rather abstract level. In this section, we present an actual news recommendation scenario. The scenario focuses on the plista GmbH. Plista runs a content and advertisement recommendation service on thousands of premium websites. These websites include portals dedicated to news and entertainment among other topics. Having a large customer base, plista processes millions of user visits on a daily basis. Each visit has to be handled in real-time as web portals attempt to instantly deliver their contents. Portals include recommendations by means of a widget.

The quality of their recommendations represents a major asset to plista. Users accepting recommendations do not only provide revenues. Evidence for increased visitor satisfaction facilitates acquiring new portals to serve with recommendations. Consequently, plista continuously seeks to improve their recommendation algorithms. Similarly, *Netflix* seeked to improve their movie recommendations thus releasing a large rating data set in 2006. The *Netflix Prize* competition has shown that combinations of recommendation algorithms provide better recommendations [6]. Combinations of algorithms have shown to better reflect contextual factors [2]. Hence, plista seeks to acquire new algorithms thus improving their system's recommendation quality.

Acquiring novel algorithms represents an endeavor to plista. In contrast to Netflix, plista's item collections are subject to continuous changes (see Sect. 6.4). Thus, an algorithm which performs well on news of two months ago could provide inadequate suggestions today. We cannot guarantee that an algorithm will achieve similar performance on novel items. As a result, plista created a platform providing researchers and practitioners with access to actual interactions. The platform was first released in 2010 as the "Open Recommendation Platform" (ORP) for internal usage. ORP allowed plista's recommendation engineers to conveniently add novel recommendation algorithms to their eco-system. Three years later, plista opened the platform for interested researchers and other third parties to evaluate their recommendation algorithms. Moreover, the platform supported plista to stay connected with the research community and actively exchange ideas. ORP ought to provide a representative selection of news portals. Otherwise, evaluations may include biases toward certain aspects. Thus, plista directly included two large-scale general news portals along with a selection of minor, rather topic-specific clients. ORP enables participants to interact with real users in a real-time setting. Interaction takes place in a two-stage process. First, news portals visitors load a news page initiating a request for recommendations. Second, the participants' server receives the requests and returns a list of suggested news items. The news portal embeds the list in the news page shown to the visitor. This setup reflects a genuine use-case. Methodologically, we refer to such settings as "living labs". This is due to the unpredictability of future interactions. Note that ORP represents a subset of all news portals served by plista. Having the idea of ORP in mind, plista contacted publishers with whom they had long-term relationships. Insightful discussion covered both advantages and disadvantages of data sharing with and contributions by researchers. Plista managed to include a representative group of publishers into ORP. The group of publishers comprises minor, medium, and large scale news portals. Furthermore, the news topics cover general selections as well as news portals providing news for specific subjects. The selection contains some news portals which operate on a similar regional level allowing evaluating recommendation methods which exchange information between domains. The included publishers use different types of widgets. Thus, ORP allows us to eliminate biases due to graphical user interfaces to a certain degree. These biases include position relative to the news article and the number of recommendations among others. We describe major components as well as vital aspects of ORP in the following subsections.

6.3.1 Involved Parties

News recommender systems concern different interest groups. These groups include news portal operators, content providers, advertising companies, recommendation providers, and visitors amongst others. We outline the individual perspective of each group.

6.3.1.1 Visitors

Visitors represent the target group of news recommendations. They require recommender systems to filter relevant items from large collection which they cannot review themselves. Hence, recommender systems provide them value in terms of the returned items' *relevancy*. Systems may determine how relevant visitors perceive suggested news articles with different means. We may conduct surveys asking visitors about the relevancy of their recommendations. This entails high costs. Therefore, we may restrict surveys to rather small proportions of visitors. Alternatively, we may evaluate visitors' dwelling times, return frequencies, or click rates.

6.3.1.2 Content Providers

We refer to content providers as editors in the context of news. Editors create and/or select the contents to be distributed through news portals. They require recommender systems to reasonably link news articles. Recommender systems ought not to confuse readers with misleading suggestions but provide relevant resources. This reflects the newspapers paradigm of structuring contents by grouping them in categories. Visitors may expect to receive suggestion conforming to their previous interactions. We may gauge recommendation algorithms quality in terms of *representativeness* from editors' perspective. How well does a recommendation represent the previous interactions? Alternatively, we may consider assessing how quickly visitors find desired contents. For instance, we may count how often visitors immediately abandon contents.

6.3.1.3 Advertisers

Advertisers strive to attract visitors. They want them to pay attention to their advertisements and ideally buy their products or services. Typically, advertisers pay per click. Although the click-through-rate fails to reflect their interests. Conversely, advertisers prefer few clicks coinciding with a high conversion rate. Conversion refers to visitors turning to customers. In our use case, we restrict our focus on click rates. ORP does not provide access to data about visitors converting to customers.

6.3.1.4 Operators

News publishers pursue two main objectives. On the one hand, they try to distribute informative and/or entertaining news to readers. On the other hand, they seek to maximize their rentability. This causes them to align the targets of users and advertisers. Users have learned to ignore adverts on webpages [11]. Prompting users to continue reading news increases the chances that they will notice adverts. Enlarged dwelling times ought to lead to higher conversion rates. Consequently, news portals' earnings will increase and improve their cost-effectiveness.

6.3.1.5 Recommendation Providers

Recommendation providers capitalize on their algorithms. Typically, portal operators pay them by click. Hence, recommendation providers seek to maximize the probability of visitors clicking on their recommendations. Hereby, they face a dilemma which we refer to as "exploration exploitation trade-off". Recommendation providers prefer to use the methods most likely to maximize click rates. However, even if individual methods have performed successfully in some scenarios, it stays unclear, which method suits the current context best. Consequently, they have to evaluate different methods which in turn may perform worse.

6.3.2 Technical Requirements

Plista have an eco-system at their disposal tailored precisely to news recommendation. In contrast, researchers using ORP may have rather limited resources. A selection of technical challenges impedes applying highly sophisticated recommendation methods. Real-time response times represent such a challenge. ORP sets the maximum response time to 100 ms. This affects both computational complexity and model updates. News portal operators require ORP to provide recommendations within a predefined time slot. Exceeding this time slot, they cannot include the recommendation into the displayed web page. Simultaneously, real-time responses require recommendation models to be available at all times. On the other hand, recommendation models ought to include recent news since visitors are likely interested in what currently happens. Thus, operators have to find ways to update their models while concurrently continue to provide recommendations. Thereby, update frequency constitutes a significant parameter. Plista's observations indicate that decreasing update frequencies negatively affects the click-through rate. Evaluating recommendation algorithms on recorded data (cf. the "Netflix Prize" challenge [7]) cannot cover this time-related aspects. Plista simultaneously runs a variety of recommendation algorithms to account for different factors determining recommendation quality. The system continues updating algorithms as news items appear, new interactions occur, and articles get updated. The frequency with which the system updates algorithms depends on the method. We report findings which plista observed for certain types of algorithms. Recommenders based on content perform well even when updated in low-frequency. In contrast, collaborative filtering methods require high update frequencies as users' interests shift. Additionally, collaborative filtering struggles to recommend items which have not obtained interactions. Further, recommendation algorithms suggesting popular news articles performed best when updated with high frequencies. ORP's users will also have to deal with the technical requirements listed above.

6.3.3 System Communication

ORP operates on an event-driven interaction model. Events include visitors requesting recommendations, visitors responding to recommendations by clicking, and news articles added to the collection or updated thereafter. Events occur in predefined contexts. ORP represents context as feature vectors. These vectors comprise information such as publisher, article, categories, and more. Events trigger messages containing the contextual information. For instance, as a user visits a news article, all of ORP's participants will receive a message. Participants may use this information to build their recommendation models. Although, ORP will randomly select an individual recommendation provider to serve this very request. ORP provides participants with an application programming interface (API). The API allows participants to connect their recommendation servers with plista's eco-system. The API uses JSON for data encoding. ORP uses HTTP POST messages to exchange requests including item updates, event notification, and recommendation requests. The contextual data in the ORP is represented through vectors. The system represents such vectors as values mapped to IDs. IDs are represented as integers. They refer to certain types of context. Vectors comprise individual IDs or lists of them. Thus, vectors allow describing an object by layering attributes. ORP distinguishes two types of vectors. One type classifies input vectors while the other refers to output vectors. Input vectors describe the context of events and messages and may be used by the participants for contextual optimization. Input vectors are static and cannot be modified. Output vectors are used to convey information about calculations. During transmission, vectors are grouped together by their type and packaged in a map where the key is the vector's ID and the value related to an instance (depending on its type). The vectors group maps are again grouped together depending on their class. Internally, ORP adapts a multi-armed bandit component. Multi-armed bandit models enable systems to balance the exploration-exploitation trade-off [45]. This trade-off implies that the system fails to accurately estimate recommendation algorithms' performance beforehand. Therefore, the system has to occasionally select seemingly suboptimal strategies to verify that it continues to apply the best strategy. ORP randomly selects recommendation algorithms among active participants. The system disables participating algorithms in case they continuously fail to provide recommendations. Having fixed technical issues, participants can re-establish the communication with ORP and again receive requests. This approach guarantees simple exploration, minimal pre-testing, and low risks of recommenders crashing. Additionally, the system contains a fallback recommender which it activates as participating servers continue to fail.

Figure 6.1 depicts the system's structure and its components. Publishers integrate recommendations as static javascript. The javascript loads recommendations by asynchronously querying ORP. ORP returns a widget box captioned "You might also be interested in…", "Recommended articles:", or similar texts. Frequently, ORP includes small pictures next to recommendations. Recommendations consist of a headline and the initial phrases up to 256 words of the recommended articles.

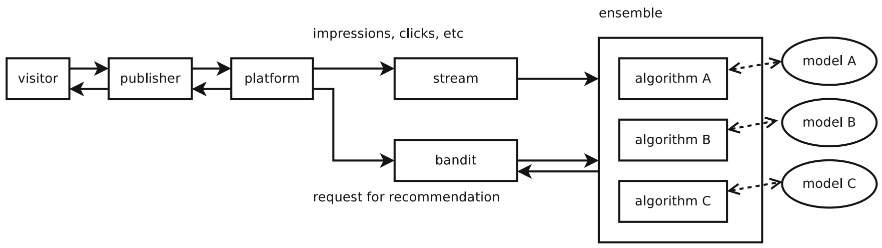
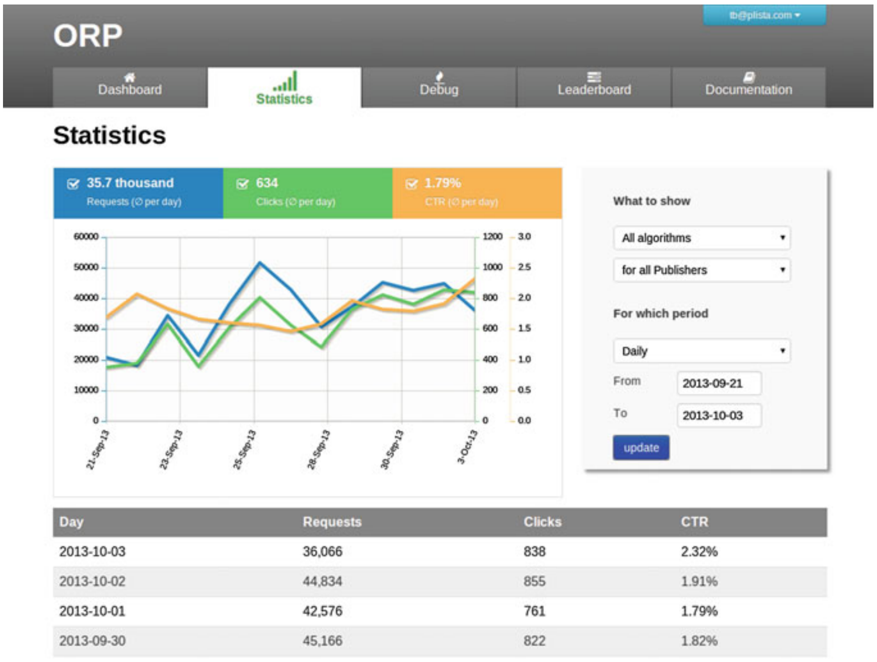


Fig. 6.1 Integration overview. Visiting a news portal initiates a circular, event-driven sequence of messages. Publishers pass the request for recommendations to the platform. Subsequently, the platform issues the request to recommendation servers. These reply with lists of recommended items. A bandit component blends these lists and forwards the resulting selection via platform and publisher to the visitor

6.3.4 *Graphical User Interface*

ORP supports participants with a graphical user interface displaying their algorithms' performances. We identify three performance affecting factors: impressions, clicks, and click-through rate (CTR). ORP shows all of them on a daily basis. Impressions



Day	Requests	Clicks	CTR
2013-10-03	36,066	838	2.32%
2013-10-02	44,834	855	1.91%
2013-10-01	42,576	761	1.79%
2013-09-30	45,166	822	1.82%

Fig. 6.2 Illustration of the graphical user interface (GUI) of ORP. The header section offers a sequence of tabs to access different sections. The figure shows the statistic section. ORP displays trajectories of the number of requests, clicks, and their relation. Additionally, ORP provides a table with absolute values for each variable

refer to requests participants received. Clicks represent recommendations which users followed by clicking. CTR describes the ratio of clicks to impressions. ORP's goal is to discover recommendation algorithms which maximize the CTR. Figure 6.2 shows an exemplary dashboard illustrating the clicks, impressions, and CTR graphically and as table. Additionally, ORP provides a leaderboard where participants can compare their performance to others.

6.3.5 Participation

There are plenty of reasons for both researchers and practitioners to contribute to ORP. There are hardly any opportunities to get access to actual interactions between users and items. Thus, ORP provides a unique way to evaluate recommendation algorithms. Existing implementations of application programming interfaces facilitate getting started. Plista as well as members of different research institutions have contributed implementations in Java,[1] PHP,[2] python,[3] and Node.js.[4] In addition, we have organized a variety of workshops and competition where researchers along with practitioners published results obtained through ORP. These events include the "International News Recommendation Workshop and Challenge"[5] [59], the "Workshop on Benchmarking Adaptive Retrieval and Recommender Systems"[6] [15], and *CLEF NEWSREEL*, the "News Recommendation Evaluation Lab"[7] [10, 28, 31].

The Open Recommendation Platform provides a unique chance for researchers to evaluate recommendation algorithms with actual user feedback. We have seen which technical requirements it entails. Systems have to reply to request within 100 ms. This prevents plista's performance from dropping below a level where customers suffer substantial losses. ORP commits to open standards with respect to data interchange and interfaces. Researchers and practitioners have already contributed implementations in a variety of programming languages. We encourage researchers to start or continue contributing recommendation algorithms to discover new ways to support users struggling to find relevant news.

[1] https://github.com/plista/kornakapi/, https://github.com/plista/orp-sdk-java/.

[2] https://github.com/plista/orp-sdk-php.

[3] https://github.com/plista/contest-py/.

[4] https://github.com/plista/contest-js/.

[5] http://recsys.acm.org/recsys13/nrs/.

[6] http://www.bars-workshop.org/.

[7] http://www.clef-newsreel.org/.

6.4 News Consumption Phenomena

This section introduces a variety of phenomena which we observed as users interact with online news portals. These phenomena distinguish the case of recommending news from other subjects such as movies, songs, or books. We dedicate a subsection to the aspects *sparsity* (6.4.1), *popularity* (6.4.2), *dynamics* (6.4.3), and *context* (6.4.4).

Recommender systems have established in a variety of use-cases. They support users' decision making. Typical use cases include deciding which movie to watch, which song to listen to, and which product to buy. Recommender systems have proofed to be valuable in those scenarios. In contrast, suggesting news entails a variety of challenges. We discuss sparsity, popularity biases, dynamic item collections, and contextual factors. These aspects represent the major challenges for operators of news portals running recommender systems.

6.4.1 Sparsity

We observe users interacting with items. Interactions cover a range of actions depending on the items. For instance, users may buy products, listen to music, watch movies, or read news articles. We can quantify interactions by the cardinalities of the involved sets of users and items. Let $u \in \mathcal{U}$ and $i \in \mathcal{I}$ denote users and items. Further, let $\text{card}(\cdot) = |\cdot|$ denote the function returning the number of elements contained in a set. Equation 6.1 defines sparsity. Sparsity reflects the fraction of interactions we actually observed by the number of possible interactions. Note that $\mathbb{I}(u, i)$ represents the indicator function returning 1 if u interacted with i and 0 otherwise (see Eq. 6.2).

$$\text{sparsity} = 1 - \frac{\sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} \mathbb{I}(u, i)}{|\mathcal{U}||\mathcal{I}|} \tag{6.1}$$

$$\mathbb{I}(u, i) = \begin{cases} 1 & : \text{ if we observe an interaction between } u \text{ and } i \\ 0 & : \text{ otherwise} \end{cases} \tag{6.2}$$

Recommender systems operate on domains with high sparsity. Recommending items with almost complete profiles represents a rather trivial problem. The lack of such comprehensive information induces the need for intelligent suggestion mechanisms. Table 6.1 displays sparsity levels of a selection of datasets. We observe that most datasets include less than 3 % of potential interactions. We determine potential interactions by multiplying the numbers of users and items. Additionally, Table 6.1 shows the relation of observed interactions to potential interactions. For instance, the *Netflix* data set exhibits 1 in 86.4 potential interactions. In contrast, we recorded data from two news portals where we observe 1 in 66622.8 potential interactions. This illustrates the difficulty to select appropriate news articles as recommendations.

Table 6.1 Levels of sparsity for a selection of well-known data sets

Data set	Sparsity	Proportion of interactions	References
Netflix prize challenge	0.98842593	86.4	[7]
Book-crossings	0.99998546	68796.6	[62]
Movielens 100 k	0.95840128	15.9	[26]
Movielens 1 M	0.98691797	23.9	[26]
Movielens 10 M	0.98827612	76.4	[26]
EachMovie	0.97631161	42.2	[58]
Jester	0.43662440	1.8	[58]
Y!Music	0.99915117	1178.8	[20]
News Portal 1	0.99998499	66622.8	
News Portal 2	0.99996663	2996.8	

6.4.2 Popularity

We encounter popularity as some items comprise a considerably larger fraction of interactions compared to others. Previous work has documented the occurrence of a popularity bias in a variety of domains. These domains include movies, songs, and books. We have grown accustomed to call popular items with specialized names. "Blockbuster", "hit", and "bestseller" refer to such popular movies, songs, and books. Recommender systems consider these type of items as adequate suggestions. We expect visitors to accept suggestions of popular items. The acceptance holds as users' tastes do not deviate from the majority of users. On the other hand, users may already be aware of the items. In such cases, the suggestion lacks serendipity. We discover popularity biases as we analyze the distribution of interactions over items. Popularity

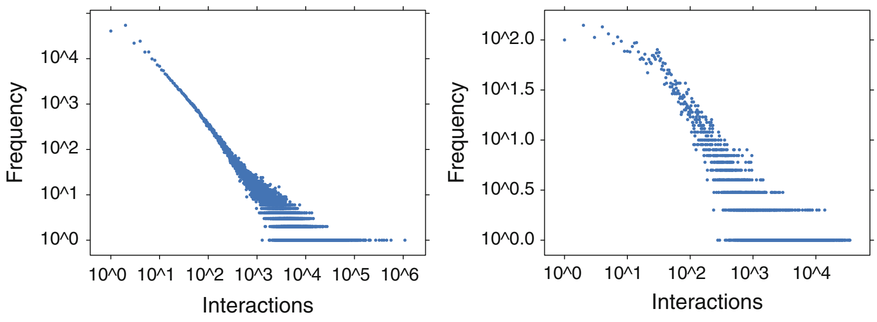


Fig. 6.3 Popularity distribution for a news portal (*left*) and the Movielens (*right*) data set

frequently induces a power-law distribution of interactions. A power-law distribution manifests as few items comprise a relatively large fraction of interactions. Conversely, a large fraction of items comprises only relatively few interactions. Popularity has been found to affect establishing users' trust into the recommender system [48]. Recommender systems which suggested popular items had a better chance to engage users to interact with them. Figure 6.3 shows the popularity distribution of a news portal's articles along with the Movielens movie rating data set. We observe that both exhibit similar shapes. Few individual items comprise a majority of interaction. Conversely, the majority of items comprises only few interactions.

6.4.3 *Item Collection Dynamics*

Continuously adding new items to existing collections represents a major reason for the information overload. Additions incur as film studios create new movies, music labels release new albums, or editors publish new books. Some of the novel items may become popular ones attracting plenty interactions. Others may remain barely known. The frequency with which items enter collections depends on the type of item. According to [22], European publishers released about 535,000 books in 2013. In contrast, news articles represent a much more high-frequency type of item. Individual news portals account for hundreds of thousands articles published per year.

News consumption differs from other domains. On the one hand, movies, songs, and books attract users throughout longer periods. For instance, we consider the rating data from the Movielens data set. Each interaction conveys a timestamp. Thus, we compute the duration in between the last and first interaction for each movie.

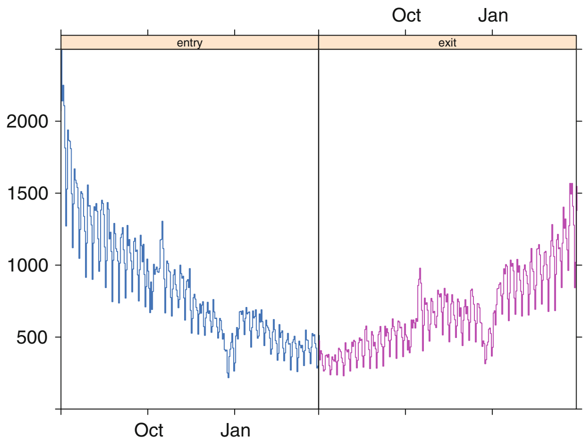


Fig. 6.4 Number of items entering and exiting the news portals over time. On the *left-hand side*, the figure shows how many items we observed whom users interact with for the first time. On the *right-hand side*, the figure illustrates how many items we observe no future interactions with

We observe the durations' median at 2,254 days. On average, news articles obtain more than half of their interaction within 24 h after their publication. The proportion of interactions concentrated on the first 24 h even increases for more popular news articles. This illustrates that users as a group consume news more rapidly than movies. On the other hand, users occasionally re-consume books and more frequently movies and songs. Users may willingly trigger the re-consumption as they choose to listen to their favorite songs or watch their favorite movie again. Additionally, broadcasters and television stations tend to re-air popular songs and movies. We have found no evidence that users frequently re-consume news articles. Figure 6.4 displays the evolution of news article collections with respect to user interactions. The data span a time of roughly eight months for a large-size news portal. On the left-hand side, we observe the number of items whom users start to interact with. Note that in the very beginning, there may be previous interactions which our data disregard. On the right-hand side, we observe the number of items for which we do not observe any future interactions. Notice that in the rightmost part, there may occur additional interaction which our data disregard. We observe a down-peak for both entries that exist during Christmas time.

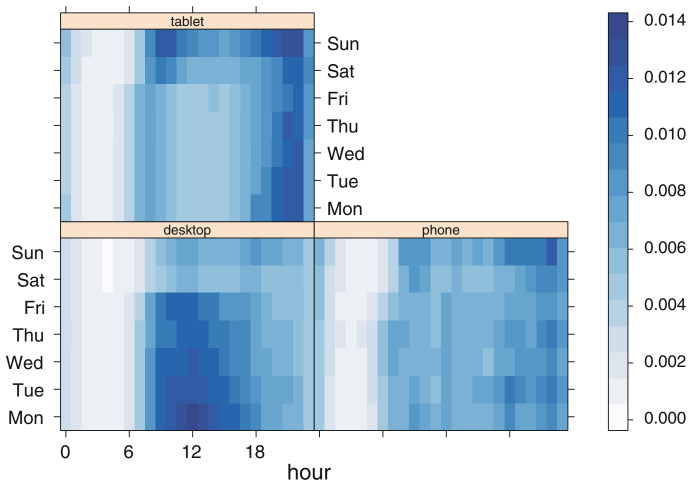


Fig. 6.5 Relative frequencies of interactions by daytime, weekday, and device. We observe that visitors tend to use their desktop computers in the typical working times (Monday to Friday, between 8 a.m. and 5 p.m.). Conversely, tablets account for relatively more interaction at the evenings as well as on the weekends. Smartphones lack such a clear tendency. All device types have comparably few interaction in the nights

6.4.4 Contextual Factors

News consumption is subject to a variety of contextual factors. Our news consumption differs with respect to the time of day, day of week, location, device, mood, and more. Determining the current context represents a difficult problem. In particular, confounding contextual factors impede recognizing situations correctly. Contexts manifest as combinations of contextual factors. For instance, users reading news on a weekday at noon on their desktop in good mood represent a specific context. Altering an individual factor may provide a context requiring a different kind of suggestions. For instance, users reading news on a weekday at noon in a good mood but on their tablet devices may dislike reading comprehensive articles due to their limited screen sizes. Figure 6.5 shows the relative frequencies of interactions grouped by daytime, weekday, and device. The majority of interactions recorded for desktop computers concentrates on the working times. Contrarily, phones as well as tablets account for larger proportions of interactions during evenings as well as weekends. Generally, we observe neglectable proportions of interactions during the night times for all device types. Suppose we ought to select a recommendation algorithm for a particular request. Context represents an important aspect we need to consider. Requests are more likely to arise from mobile devices on the weekend. Mobile devices provide less space to display recommendations on. Thus, we should consult the recommendation method which performs best under these circumstances.

We have seen that sparsity, popularity, dynamics, and context represent major impeding factors for news recommendation. Sparsity hampers establishing valuable user and item profiles. Sparsity represents a particular challenge for newly added users and items. This is due to the system having almost no knowledge about preference relation with the entity. The system struggles to determine what items a new user will like. Conversely, it cannot reliably select potential consumers. Popularity skews consumption distributions as few items concentrate large amounts of interactions. Contrarily, unpopular items see hardly any interactions. Dynamics refer to the system characteristic of fluctuating item collections. In established domains songs, movies, and books remain recommendable items. Conversely, news' relevance fades with time. Finally, systems have to consider users' current context to select enjoyable articles. Users may dislike reading comprehensive articles on mobile devices. In Sect. 6.5, we discuss a selection of algorithms and their abilities to deal with these specificities.

6.5 Recommendation Algorithms for News

Recommendation algorithms are subject to a vigorous research community. Researchers continuously propose and evaluate novel methods or extend existing ones. Methods differ with respect to complexity, applicability, and the underlying ideas. In the following, we introduce and discuss four kinds of such underlying ideas and their implementations:

Table 6.2 Notation used in the algorithmic descriptions

Symbol	Meaning
$\mathcal{I}$	Set of items
$\mathcal{U}$	Set of users
$\mathbb{I}(u, i)$	Interaction indicator function
R	Interaction matrix
$\text{top}(k, c, X)$	Function returning the k largest values with respect to criteria c of a collection X

- simple methods with low complexity
- collaborative filtering
- content-based filtering
- ensembles of the former 3 notions.

In addition, we highlight the applicability of various implementations for news recommendations. News recommendation entails specific requirements due to their characteristics (see Sect. 6.4). Table 6.2 introduces basic notation which we use in the algorithmic descriptions.

6.5.1 Simple Methods

Researchers introduce simple methods as baselines to elucidate the improvements which their novel method provides. Nevertheless, simple methods carry some advantages with them. Typically, simple methods can be easily implemented and exhibit low complexity. Frequently, simple methods target specific factors. In other words, simple methods follow a single idea. For instance, always recommend the most popular item the requesting user has not yet interacted with. We call this simple method the "most popular" recommender. We will elaborate on this method and introduce two additional ones.

6.5.1.1 Most Popular

The *most popular* recommender suggests items according to their popularity. This follows the notion that items comprising interactions with a majority of users will be relevant for other users as well. This resembles lead articles in newspapers—the analogous counterparts of digital news portals. Lead articles obtain more attention than articles situated in latter parts of newspapers. Algorithm 1 describes the procedure to build a model based on the most popular recommender. The algorithm requires a matrix of interactions, the set of items, and the number of items to recommend as input. Subsequently, the method iteratively evaluates the popularity of each item. Items enter the list of recommendations as they are amongst the k most popular items. The algorithm can be altered to consider different timeframes by restricting the interactions which it receives (see Sect. 6.4.2).

Algorithm 1 Most Popular Recommender

```
INPUT matrix of interactions $R$, set of items $\mathcal{I}$, number of items to recommend $k$
OUTPUT list of $k$ items sorted by popularity
1: for all $i \in \mathcal{I}$ do
2:     popularity$(i) \leftarrow \sum_{u \in \mathcal{U}} \mathbb{I}(i, u)$
3: end for
4: recommendations $\leftarrow$ top$(k$, popularity, $\mathcal{I})$
```

6.5.1.2 Most Recent

The *most recent* recommender builds upon the notion of recency. As Algorithm 2 illustrates, most recent recommendation ranks items according to their appearance in the collections. The algorithm takes the set of items, their creation time, the current time, and a specification of how many items to recommend as input. Subsequently, we obtain an items age subtracting the date of creation from the current time. The method determines which items to recommend by cutting the list of items ordered by their ages at position k. As new items enter the collection, they move on top of the list replacing the former top-ranked ones. Thus, the method keeps the items to recommend up to date (see Sect. 6.4.3).

Algorithm 2 Most Recent Recommender

```
INPUT set of items $\mathcal{I}$, timestamps of item creation $\tau(i)$, current time $T$, number of items to recommend $k$
OUTPUT list of $k$ items sorted by date of creation
1: for all $i \in \mathcal{I}$ do
2:     t$(i) \leftarrow T - \tau(i)$
3: end for
4: recs $\leftarrow$ top$(k, -t, \mathcal{I})$
```

6.5.1.3 Random

Recommending random items represents another simple method. Randomly picking items yields the risk of suggesting irrelevant items. On the other hand, it could provide access to items which are neither popular nor recent and thus would not have been found by users. Algorithm 3 depicts the random recommendation procedure. It randomly adds items to the list of recommendations until the list has the desired capacity. Items may not be redundant.

Algorithm 3 Random Recommender

```
INPUT set of items $\mathcal{I}$, number of items to recommend $k$
OUTPUT list of $k$ items to recommend
1: while |recommendations| $< k$ do
2:     $i \leftarrow$ rand($\mathcal{I}$)
3:     if $i \notin$ recommendations then
4:         recommendations $\leftarrow$ recommendations $\cup\, i$
5:     end if
6: end while
```

6.5.2 Collaborative Filtering

Collaborative filtering (CF) adapts the notion of taste similarity continuity. In other words, if two users exhibit similar tastes in the past, collaborative filtering assumes that they will continue to prefer similar items. Previous research provides an abundance of algorithms for collaborative filtering. Adomavicius and Tuzhilin [1] distinguish memory-based and model-based collaborative filtering algorithms. Memory-based CF uses all available data for recommendation. In contrast, model-based CF generalizes patterns apparent in interactions and provides recommendations based on these models. Matrix factorization techniques have established among the most successful model-based CF methods.

Algorithm 4 illustrates memory-based recommendation from the user perspective. The method requires the sets of users and items, a similarity function, the number of neighbors to consider, along with the length of the recommendation lists to produce. The algorithm iterates first the set of users to determine whose taste resembles the target user's taste. Subsequently, the method predicts the preferences for each item the target user is unaware of. The algorithm returns the k items with the highest scores.

Algorithm 5 shows memory-based recommendation from the item perspective. In contrast to Algorithm 4, the method compute similarities between items in terms of their interactions. This is advantageous in cases where $|\mathcal{I}| \ll |\mathcal{U}|$ since we skip the computational more expensive loops over the larger user dimension.

Matrix factorization has established as one of the most successful type of collaborative filtering. These algorithms reduce the dimensionality of a M by N interaction matrix R to a lower rank approximation. Projecting user and item profiles in this lower space enables recommender systems to compute similarities between them. We present two methods to learn these low rank approximations. Algorithm 6 learns low rank approximations with an alternating least squares procedure. Hereby, we randomly initialize two factor matrices. These matrices' dimension follows the number of users, items, and the desired latent factors. Subsequently, the algorithm iteratively optimizes a target function. This target function measures how close the predicted interactions match the observed interactions. Root mean squared error (RMSE) represents a popular choice for such a function. The algorithm keeps one feature matrix

Algorithm 4 User-based K-nearest Neighbor CF

```
INPUT set of users $\mathcal{U}$, set of items $\mathcal{I}$, similarity function $\sigma(\cdot,\cdot)$, number of neighbors $l$, number of item to recommend $k$
OUTPUT list of $k$ recommended items
1: $u$                                                    ▷ target user
2: $N \leftarrow \emptyset$                               ▷ set of neighbors
3: recommendations $\leftarrow \emptyset$                 ▷ list of recommendations
4: for all $v \in \mathcal{U} \setminus u$ do
5:     $s \leftarrow \sigma(u, v)$
6:     if $s \geq \sigma(u, N_l)$ then
7:         $N \leftarrow N \cup (v, s)$
8:     end if
9: end for
10: for all $i \in \mathcal{I} \setminus \mathcal{I}_u$ do   ▷ ($\mathcal{I}_u$ refers to items which $u$ already knows)
11:     for all $n \in N$ do
12:         if $\mathbb{I}(n, i) = 1$ then
13:             $\hat{r}_n \leftarrow s_n r(n, i)$
14:         end if
15:     end for
16:     $\hat{r} \leftarrow \sum_{\mathbb{I}(n,i)=1} \hat{r}_n$
17:     if $\hat{r} >$ sort(recommendations$_k$) then
18:         add$(i)$
19:         if |recommendations| $> k$ then
20:             remove(recommendations$_{k+1}$)
21:         end if
22:     end if
23: end for
```

Algorithm 5 Item-based K-nearest Neighbor CF

```
INPUT set of users $\mathcal{U}$, set of items $\mathcal{I}$, similarity function $\sigma(\cdot,\cdot)$, number of items to recommend $k$
OUTPUT list of $k$ recommended items
1: $u$                                                    ▷ target user
2: $S$                        ▷ $|\mathcal{I}| \times |\mathcal{I}|$ similarity matrix for all combinations of items
3: $N \leftarrow \emptyset$                               ▷ set of neighbors
4: recommendations $\leftarrow \emptyset$                 ▷ list of recommendations
5: for all $i \in \mathcal{I}$ do
6:     for all $j \in \mathcal{I} \setminus i$ do
7:         $S_{i,j} \leftarrow \sigma(i, j)$
8:     end for
9: end for
10: for all $i \in \mathcal{I}_u^c$ do   ▷ $\mathcal{I}_u^c$ refers to all items the target user $u$ did not interact with
11:     $\hat{r}_i \leftarrow u \otimes S_i$                ▷ $u$ refers to items a user has interacted with
12:     recommendations $\leftarrow$ top$(k, \hat{r}, \mathcal{I}_u^c)$
13: end for
```

fixed while determining the gradient with respect to the remaining matrix. The algorithm switches matrices in the next iterative step. As soon as a stopping criterion is matches, the procedure terminates providing the low rank approximation. Stopping criteria include thresholds as well as maximum iterations. Thresholds define a limit for the improvement between iterations. As we observe less improvement than defined, we terminate the procedure. Conversely, a maximum number of iterations aborts disregarding improvements. Both approaches have advantages. Thresholds guarantee convergence to the desired quality. Unfortunately, this may lead to long running times. In contrast, maximum iterations assure limited running. Still, the algorithm may provide only sub-optimal solutions. We obtain recommendations as we map user and item profiles onto the low ranked subspace.

Algorithm 6 Alternating Least Squares Matrix Factorization CF

```
INPUT interaction matrix $R_{u,i}$, number of latent factors to consider $k$, termination condition $\epsilon$,
optimization function $q(\cdot, \cdot)$
OUTPUT predicted interactions
1: $P \leftarrow \text{rand}(|\mathcal{U}|, k)$                    ▷ randomly initialize latent user factors
2: $Q \leftarrow \text{rand}(k, |\mathcal{I}|)$                    ▷ randomly initialize latent item factors
3: while $\epsilon = \text{false}$ do
4:     $P \leftarrow \arg\max_P q(R, PQ^\intercal)$                ▷ Optimize $P$ keeping $Q$ fixed
5:     $Q \leftarrow \arg\max_Q q(R, PQ^\intercal)$                ▷ Optimize $Q$ keeping $P$ fixed
6: end while
7: recommendations $\leftarrow \text{top}(k, \langle P_u, Q_i \rangle, R)$
```

Algorithm 7 illustrates an alternative way to obtain low rank approximations. Instead of iteratively optimizing user or item factors, the algorithm randomly picks interactions. Subsequently, we compute the gradients for both users and item factors and adjust the factor matrices accordingly. Identical stopping criteria apply to this setting.

Algorithm 7 Stochastic Gradient Descent Matrix Factorization CF

```
INPUT interaction matrix $R_{u,i}$, number of latent factors to consider $k$, termination condition $\epsilon$,
optimization function $q(\cdot, \cdot)$, learning rate $\nu$
OUTPUT predicted interactions
1: $P \leftarrow \text{rand}(|\mathcal{U}|, k)$                    ▷ randomly initialize latent user factors
2: $Q \leftarrow \text{rand}(k, |\mathcal{I}|)$                    ▷ randomly initialize latent item factors
3: while $\epsilon = \text{false}$ do
4:     $(u, i) \leftarrow \text{rand}(R)$                          ▷ pick random interactions
5:     $e \leftarrow q(R(u, i), P_u Q_i^\intercal)$                ▷ determine prediction quality
6:     $P \leftarrow P \cdot \nu \nabla_e P$                       ▷ update user factors
7:     $Q \leftarrow Q \cdot \nu \nabla_e Q$                       ▷ update item factors
8: end while
9: recommendations $\leftarrow \text{top}(k, \langle P_u, Q_i \rangle, R)$
```

6.5.3 Content-Based Filtering

Content-based Filtering (CBF) supposes that users will continue to interact with items that share similar contents. For instance, users interact with songs. The system observes that a user frequents a certain artist. As a consequence, the system suggests other items related to the artist. Algorithm 8 shows the content-based recommendation algorithm. The system requires the set of items, its features, a user profile, along with a similarity function. The algorithm computes the similarities between any combinations of items. Finally, we project the user profile onto the similarity matrix. As a result, we obtain a score for each item. The system recommends the top k items excluding items the users is already familiar with. This approach directs the major efforts towards the choice of similarity metrics as well as the decision on which features to use.

Algorithm 8 Content-based Filtering

```
INPUT set of items $\mathcal{I}$, item feature matrix $F$, user profile $U$, similarity function similarity$(X, Y)$,
number of recommendations $k$
OUTPUT similar items
1: $S \leftarrow \emptyset$                                          ▷ Initialize similarity matrix $S$
2: for all do$i \in \mathcal{I}$
3:     for all do$j \in \mathcal{I} \setminus i$
4:         $S_{i,j} \leftarrow$ similarity$(F_i, F_j)$
5:     end for
6: end for
7: recommendations $\leftarrow$ top$(k, \langle U, S, \rangle, \mathcal{I} \setminus U)$
```

6.5.4 Ensembles

So far, we have introduced a variety of recommendation algorithms. These algorithms entail different ideas and require varying data. Machine learning research has shown that combining various algorithms yields potential improvements [23].

In the context of news recommendation, we may combine individual algorithms using different methods. Multi-armed bandits represent such a method [37]. Multi-armed bandits target the problem of uncertainty with respect to the choice of algorithm, parameter, or data. Uncertainty arises as the system cannot determine which algorithm, parameter, or data will perform best. We refer to this problem as "exploration–exploitation" dilemma. The problem manifests as systems try to avoid selecting sub-optimal algorithms, parameter, or data. Conversely, system cannot judge the performance differences between different algorithms, parameter, or data unless they continuously evaluate them against each other. We may define multi-armed bandits in different forms. First, we use them to switch different methods. For

Table 6.3 Computational complexity of recommendation algorithms for news

Algorithm	Complexity
Most popular	$\mathcal{O}(MN)$
Most recent	$\mathcal{O}(N)$
Random	$\mathcal{O}(N)$
User-based CF	$\mathcal{O}(M(M-1)N)$
Item-based CF	$\mathcal{O}(MN^2)$
ALS CF	$\mathcal{O}(MNk^2)$
SGD CF	$\mathcal{O}(S)$
Content-based filtering	$\mathcal{O}(MN^2)$

M refers to the number of users while *N* refers to the number of items. *S* represents an unknown variable which depends on the configuration with which (user, item) pairs are selected

instance, the system switches between implementations of collaborative filtering, content-based filtering, and other methods. Second, we may keep the algorithm fixed. The multi-armed bandit switches parameters in this scenario. For instance, we select item-based collaborative filtering. This algorithm expects inputs including similarity function. Pearson's correlation coefficient and cosine similarity represent examples of such similarity functions. The multi-armed bandit may then switch these. Finally, we may limit the data we use to learn a model representing interaction patterns. For instance, we may argue that with time passing the relevancy of news diminishes. Thus, we may consider various time frames. For instance, we may learn a model based on interactions which occurred up to 3 h, up to 6 h, and up to a day ago. The multi-armed bandit may switch which data to use. Lommatzsch [42] describes a sophisticated way to allay negative effects induced by exploration. The proposed method evaluates all configurations in a slightly delayed time. In other words, instead of averaging performances over time, the method re-issues every request to all configurations. Thus, the system assesses performances more reliably. Consequently, the system learns to select the most promising configuration more quickly. Results show that algorithms performances strongly depend on contextual factors. As a result, individual algorithms cannot dominate other algorithms consistently.

6.5.5 Scalability

As discussed in Sect. 6.4, recommending news articles entails technical requirements. In particular, systems must deal with a large volume of requests arriving in high rates. Consequently, recommendation algorithms have to scale at such conditions.

Table 6.3 refers each algorithms to an estimated complexity. Note that intelligent ways of situating data and similar tools may decrease the actual complexity. The table ought to illustrate differences between individual methods. For instance, random and most recent methods operate independent from the user dimension. The complexity of the more sophisticated methods including ALS and SGD collaborative filtering depends on the stopping criterion. These methods either stop as the optimization target surpasses a threshold or after a predefined number of iterations.

Besides algorithmic optimization, a selection of frameworks enables systems to parallelize their computation thus achieving considerable speed-ups. These frameworks include *hadoop*,[8] *spark*,[9] and *storm*[10] amongst others. Additionally, news recommender system operators may consider to pre-compute recommendations as soon as possible. For instance, they may estimate the probability that a novel article will become popular. If the probability estimate is sufficiently high, the system could start recommending it more often.

6.6 Evaluation Criteria

This section treats aspects related to news recommender systems' evaluation protocols. Section 6.2 discussed aspects which we need to consider when evaluating news recommender systems. First, we have to define quality criteria. These criteria relate to the use-case introduced in Sect. 6.3. We aim to assess how visitors, advertisers, as well as operators benefit of having the recommender system in place. ORP does not reveal information about earnings or users converted to customers. Hence, we rely on the interactions which we observe. These interaction represent implicit preference indicators. In contrast, users may explicitly rate items on a pre-defined scale. Lacking such graded feedback, we dismiss error-based metrics—e.g., RMSE, MAE—as we disregard ranking-based criteria including normalized discounted cumulative gain (nDCG) and mean reciprocal rank (MRR). Measures used in information retrieval dispense with numerical preferences. Recall and precision require knowing whether or not a certain item is relevant to a user. Our observations fail to provide such information for all (user, item)-pairs. We may infer relevancy as users select news articles. Still, articles remain ambiguous until we observe interactions with users. Have users missed to see the article? Have users seen the article and decided not to read them? We can evaluate search engine as we predefine each document's relevance given a query. Unfortunately, we have no analogous concept for recommender systems. This is due to individual users' varying preferences. We cannot tell whether a specific news article interests a user unless the user reads it. Thus, we adhere to the notion of click-through-rates (CTR). CTR relates the number of clicks to the number of requests which the recommender system received.

ORP supports evaluating recommendation algorithms by means of live interactions with users. Additionally, we may record such interactions. Subsequently, we can use these records to replay the stream of interactions. We can apply various recommendations methods and assess their qualities having future click events recorded.

[8] http://hadoop.apache.org/.

[9] https://spark.apache.org/.

[10] https://storm.incubator.apache.org/.

Li et al. [38] showed that this methodology yields unbiased results as long as we disregard recommendations which have not been shown to users. We cause the offline evaluation to fine-tune our methods and obtain better strategies for exploration.

6.7 Discussion

In this section, we summarize our findings and provide an outlook to future research directions. Suggesting relevant news articles to visitors represents a major challenge to online news portals. In particular, as systems typically have to deal with insufficient information about users preferences. Most users refrain from interacting with plenty news articles but focus their attention on smaller subsets. In addition, a stream of new articles continuously enters the portals' collections. This blurs relations between visitors and articles. Established recommendation algorithms generally assume rather static preferences. Thus, news portals had to come up with novel methods to support visitors as they seek for relevant news. Portals use to implement various recommendation algorithms in order to cover plenty of aspects reflecting different facets of relevancy. Combinations of these algorithms serve visitors with recommended readings. They consider factors including context, popularity, recency, and more. Barriers between academia and research impede further improving the algorithmic performance. Companies avoid publishing data. On the one hand, they may fear privacy issues. On the other hand, they consider their data as asset to their company which they seek to preserve. Conversely, academia generates ideas on how to provide better suggestions. Although, they struggle to evaluate their approaches due to lacking data. Recently, the company plista constructed the "Open Recommendation Platform" (ORP). The platform provides researches access to an actual news recommendation system. Plista expects to improve their recommendation quality. Researchers get the chance to evaluate their ideas with the feedback of actual users. Simultaneously, research faces the technical requirements of a large-scale content provider. A large volume of requests has to be handled at high rates. The system grants as much as 100 ms to send the list of recommended items. Researchers who manage to overcome these restrictions have the unique opportunity to evaluate on a large scale. Millions of users request news article recommendation through ORP. Evaluation concentrates on the click-through-rate (CTR). Other evaluation criteria require graded feedback. For instance, root mean squared error (RMSE; evaluation criteria of the *Netflix Prize*) requires numerically expressed preferences. Users reading news online tend to express their preferences by selection at most.

We identify various directions for future research. We admit that the CTR might not fully capture user preferences. Users may accidentally click on recommendations. Other may immediately abandon the recommended item. Conversely, users may not click on recommendations as they did not perceive them. For instance, recommendations placed on the bottom of the web page require users to scroll down to be seen. Future research may enrich evaluation with additional factors such as dwelling times. Detecting hidden patterns in interactions represents another future research topic. User profiles are typically sparse as they interact with few items. We may consider recommending not for individual users but for groups of similar users. This idea reflects the notion of certain users sharing similar preferences. For instance, some users may focus on sports-related news. Hence, news recommender systems could recommend articles to the group of these users rather than to each individual. Discovering similarities in highly sparse data represents a major scientific challenge. Finally, we consider early trend detection as a means to further improve recommendation quality. Imagine that a novel item enters the collection of news articles. Systems ought to estimate how likely it will attract a lot of interest. If the system manages to accurately estimate the probability, it will be able to boost interesting items early. Thus, the system will collect a larger amount of clicks than continuing to recommend items which users disregard.

Acknowledgments This work was funded by the Federal Ministry of Economic Affairs and Energy (BMWi) under funding reference number KF2392313KM2.

References

1. G. Adomavicius, A. Tuzhilin, Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. IEEE Trans. Knowl. Data Eng. **17**(6), 734–749 (2005)
2. G. Adomavicius, A. Tuzhilin, Context-aware recommender systems, in *Proceedings of the 2008 ACM Conference on Recommender Systems—RecSys'08*, p. 335 (2008)
3. G. Adomavicius, J. Zhang, Stability of recommendation algorithms. ACM Trans. Inf. Syst. **30**(4), 1–31 (2012)
4. F. Aiolli, Efficient top-n recommendation for very large scale binary rated datasets, in *ACM RecSys*, pp. 273–280 (2013)
5. X. Amatriain, Mining large streams of user data for personalized recommendations. ACM SIGKDD Explor. Newsl. **14**(2), 37 (2013)
6. R. Bell, Y. Koren, Lessons from the Netflix prize challenge. ACM SIGKDD Explor. **9**(2), 75–79 (2007)
7. J. Bennett, S. Lanning, The Netflix prize, in *KDDCup*, pp. 3–6 (2007)
8. D. Billsus, M.J. Pazzani, Adaptive news access, in *The Adaptive Web*, Chapter 18, ed. by P. Brusilovsky, A. Kobsa, W. Nejdl (Springer, New York, 2007), pp. 550–570
9. T. Bogers, A. van den Bosch, Comparing and evaluating information retrieval algorithms for news recommendation, in *Proceedings of the 2007 ACM Conference on Recommender Systems—RecSys'07* (ACM Press, New York, 2007) p. 141
10. T. Brodt, F. Hopfgartner, Shedding light on a living lab: the CLEF NEWSREEL open recommendation platform, in *IIiX'14: Proceedings of Information Interaction in Context Conference* (ACM, 2014), pp. 223–226
11. M. Burke, A. Hornof, E. Nilsen, N. Gorman, High-cost banner blindness: Ads increase perceived workload, hinder visual search, and are forgotten. ACM Trans. Comput.-Hum. Interact. (TOCHI) **12**(4), 423–445 (2005)
12. I. Cantador, A. Bellogín, P. Castells, News @ hand: a semantic web approach to recommending news. Adapt. Hypermed. Adapt. Web-based Syst. **5149**, 279–283 (2008)
13. I. Cantador, A. Bellogín, P. Castells, Ontology-based personalised and context-aware recommendations of news items, in *2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology* (IEEE, 2008), pp. 562–565

14. M. Capelle, F. Hogenboom, A. Hogenboom, F. Frasincar, Semantic news recommendation using WordNet and Bing similarities categories and subject descriptors, in *Symposium on Applied, Computing*, pp. 296–302 (2013)
15. P. Castells, F. Hopfgartner, A. Said, M. Lalmas, Workshop on benchmarking adaptive retrieval and recommender systems: Bars 2013, in *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR'13* (ACM, New York, 2013) p. 1133
16. P. Cremonesi, Performance of recommender algorithms on top-n recommendation tasks categories and subject descriptors, in *Proceedings of the 2010 ACM Conference on Recommender Systems*, pp. 39–46 (2010)
17. A. Das, M. Datar, A. Garg, S. Rajaram, Google news personalization: scalable online, in *WWW*, pp. 271–280 (2007)
18. M.S. Desarkar, Aggregating preference graphs for collaborative rating prediction, in *Proceedings of the 2010 ACM Conference on Recommender Systems*, pp. 21–28 (2010)
19. M. Deshpande, G. Karypis, Item-based top-n recommendation algorithms. ACM Trans. Inf. Syst. **22**(1), 143–177 (2004)
20. G. Dror, N. Koenigstein, Y. Koren, M. Weimer, The Yahoo! music dataset and KDD-Cup'11, in *KDD Cup*, pp. 8–18 (2012)
21. G. De Francisci, A. Gionis, C. Lucchese, From chatter to headlines: harnessing the real-time web for personalized news recommendation, in *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining*, pp. 153–162 (2012)
22. Federation of European Publishers. European book publishing statistics (2013)
23. Y. Freund, R.E. Schapire, A desicion-theoretic generalization of on-line learning and an application to boosting, in *Computational Learning Theory*, pp. 23–37 (1995)
24. Q. Gao, F. Abel, G.-J. Houben, K. Tao, Interweaving trend and user modeling for personalized news recommendation, in *2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology* (IEEE, 2011), pp. 100–103
25. F. Garcin, C. Dimitrakakis, B. Faltings, Personalized news recommendation with context trees, in *ACM RecSys*, pp. 105–112 (2013)
26. J.L. Herlocker, J.A. Konstan, A. Borchers, J. Riedl, An algorithmic framework for performing collaborative filtering, in *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (ACM, 1999), pp. 230–237
27. J.L. Herlocker, J.A. Konstan, L.G. Terveen, J.T. Riedl, Evaluating collaborative filtering recommender systems. ACM Trans. Inf. Syst. (TOIS) **22**(1), 5–53 (2004)
28. F. Hopfgartner, B. Kille, A. Lommatzsch, T. Plumbaum, T. Brodt, T. Heintz, Benchmarking news recommendations in a living lab, in *Proceedings of the Fifth International Conference of the CLEF Initiative, CLEF'14* (Springer, New York, 2014), pp. 250–267
29. M. Jahrer, A. Töscher, R. Legenstein, Combining predictions for accurate recommender systems, in *KDD*, pp. 693–701 (2010)
30. A. Karatzoglou, M. Larson, GAPfm: optimal top-n recommendations for graded relevance domains, in *Proceedings of the 22nd ACM Conference on Information and Knowledge Management* (ACM, 2013) pp. 2261–2266
31. B. Kille, T. Brodt, T. Heintz, F. Hopfgartner, A. Lommatzsch, J. Seiler, Overview of CLEF NEWSREEL 2014: news recommendation evaluation labs, in *Proceedings of the Fifth International Conference of the CLEF Initiative, CLEF'14* (Springer, New York, 2014)
32. M. Kompan, M. Bielikova, Content-based news recommendation, in *EC-Web*, pp. 1–12 (2010)
33. N. Lathia, S. Hailes, L. Capra, Temporal collaborative filtering with adaptive neighbourhoods, in *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval—SIGIR'09* (ACM Press, New York, 2009), p. 796
34. N. Lathia, S. Hailes, L. Capra, X. Amatriain, Temporal diversity in recommender systems, in *Proceeding of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval—SIGIR'10*, (ACM Press, New York, 2010), p. 210
35. L. Li, T. Li, News recommendation via hypergraph learning: encapsulation of user behavior and news content, in *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, pp. 305–314 (2013)

36. L. Li, D.-D. Wang, S.-Z. Zhu, T. Li, Personalized news recommendation: a review and an experimental investigation. J. Comput. Sci. Technol. **26**(5), 754–766 (2011)
37. L. Li, W. Chu, J. Langford, R.E. Schapire, A contextual-bandit approach to personalized news article recommendation, in *Proceedings of the 19th International Conference on World Wide Web—WWW'10* (ACM Press, New York, 2010), p. 661
38. L. Li, W. Chu, J. Langford, X. Wang, Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms, in *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining—WSDM'11* (ACM Press, New York, 2011), p. 297
39. B. Liu, *Web Data Mining* (Springer, New York, 2008)
40. J. Liu, P. Dolan, E. Rønby Pedersen. Personalized news recommendation based on click behavior, in *Proceedings of the International Conference on Intelligent User Interfaces*, pp. 31–40 (2010)
41. N.N. Liu, Q. Yang, Eigenrank: a ranking-oriented approach to collaborative filtering, in *SIGIR*, pp. 83–90 (2008)
42. A. Lommatzsch, Real-time news recommendation using context-aware ensembles, in *Advances in Information Retrieval* (Springer, New York, 2014), pp. 51–62
43. L. Lü, M. Medo, C.-H. Yeung, Y.-C. Zhang, Y.-K. Zhang, T. Zhou, Recommender systems. Phys. Rep. **519**, 1–49 (2012)
44. Y. Lv, T. Moon, P. Kolari, Z. Zheng, X. Wang, Y. Chang, Learning to model relatedness for news recommendation, in *Proceedings of the 20th International Conference on World Wide Web—WWW'11*, p. 57 (2011)
45. F. Maes, L. Wehenkel, D. Ernst, Learning to play k-armed bandit problems, in *Proceedings of the 4th International Conference on Agents and Artificial Intelligence* (*ICAART 2012*) (2012)
46. A. Montes-García, J.M.Á. Rodríguez, J.E. Labra-Gayo, M. Martínez-Merino, Towards a journalist-based news recommendation system: the Wesomender approach. Expert Syst. Appl. **40**(17), 6735–6741 (2013)
47. O. Phelan, K. Mccarthy, M. Bennett, B. Smyth, Terms of a feather: content-based news discovery and recommendation using twitter, in *ECIR*, pp. 448–459 (2011)
48. P. Pu, L. Chen, Trust-inspiring explanation interfaces for recommender systems. Knowl.-Based Syst. **20**(6), 542–556 (2007)
49. Y. Ren, G. Li, W. Zhou, Learning user preference patterns for top-n recommendations, in *2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, pp. 137–144 (2012)
50. S. Rendle. Learning recommender systems with adaptive regularization, in *WSDM*, pp. 133–142 (2012)
51. S. Rendle, C. Freudenthaler, Improving pairwise learning for item recommendation from implicit feedback, in *Proceedings of the 7th ACM International Conference on Web Search and Data Mining—WSDM'14*, pp. 273–282 (2014)
52. F. Ricci, L. Rokach, B. Shapira, P.B. Kantor, *Recommender Systems Handbook* (Springer, New York, 2011)
53. R. Salakhutdinov, A. Mnih, G. Hinton, Restricted boltzmann machines for collaborative filtering, in *Proceedings of the 24th International Conference on Machine Learning—ICML'07* (ACM Press, New York, 2007), pp. 791–798
54. M. Seeger, Scalable collaborative bayesian preference learning, in *AISTATS*, vol. 33 (2014)
55. G. Shani, A. Gunawardana, Evaluating recommendation systems, in *Recommender Systems Handbook* (Springer, New York, 2011)
56. Y. Shi, A. Karatzoglou, L. Baltrunas, M. Larson, N. Oliver, A. Hanjalic, CLiMF: learning to maximize reciprocal rank with collaborative less-is-more filtering, in *RecSys*, pp. 139–146 (2012)
57. J.-W. Son, A.-Y. Kim, S.-B. Park, A location-based news article recommendation with explicit localized semantic analysis, in *SIGIR*, pp. 293–302 (2013)
58. G. Takacs, I. Pilaszy, B. Nemeth, D. Tikk, Scalable collaborative filtering approaches for large recommender systems. J. Mach. Learn. Res. **10**, 623–656 (2009)

59. M. Tavakolifard, J.A. Gulla, K.C. Almeroth, F. Hopfgartner, B. Kille, T. Plumbaum, A. Lommatzsch, T. Brodt, A. Bucko, T. Heintz, Workshop and challenge on news recommender systems, in *Proceedings of the 7th ACM Conference on Recommender Systems*, pp. 481–482 (2013)
60. S. Vargas, P. Castells, Rank and relevance in novelty and diversity metrics for recommender systems, in *Proceedings of the Fifth ACM Conference on Recommender Systems—RecSys'11*, p. 109 (2011)
61. X. Yang, H. Steck, Y. Guo, Y. Liu, On top-k recommendation using social networks, in *Proceedings of the Sixth ACM Conference on Recommender systems—RecSys'12* (ACM Press, New York, 2012), p. 67
62. C.-N. Ziegler, S.M. McNee, J.A. Konstan, G. Lausen, Improving recommendation lists through topic diversification, in *Proceedings of the 14th International Conference on World Wide Web* (ACM, 2005), pp. 22–32