Pessimists and optimists: Improving collaborative filtering through sentiment analysis

Miguel Á. García-Cumbreras *, Arturo Montejo-Ráez, Manuel C. Díaz-Galiano

Department of Computer Science, Escuela Politécnica Superior, Universidad de Jaén, E-23071 Jaén, Spain

A R T I C L E   I N F O

Keywords:
Collaborative filtering
Opinion mining
Recommender systems
IMDb corpus
Sentiment analysis
Polarity classification
User profile generation

A B S T R A C T

This work presents a novel application of Sentiment Analysis in Recommender Systems by categorizing users according to the average polarity of their comments. These categories are used as attributes in Collaborative Filtering algorithms. The experiments stress the informative value of comments. By applying Sentiment Analysis approaches some Collaborative Filtering algorithms can be improved in rating prediction tasks. The results indicate that we obtain a more reliable prediction considering only the opinion text (RMSE of 1.868), than when apply similarities over the entire user community (RMSE of 2.134) and sentiment analysis can be advantageous to recommender systems.

© 2013 Elsevier Ltd. All rights reserved.

1. Introduction

Today we find on the Internet a huge amount of social and unstructured information, which is called the Social Web. The number of online opinions or comments expressed in the thoughts about a variety of topics is constantly growing, and a large percentage of Internet users uses these opinions and assessments to make decisions. Thousands of opinions and assessments on books, movies, travels, products or services are populating the web every day.

In Information Retrieval, Recommender Systems (RS) are tools whose objective is to assist users in their information search processes, helping them to filter the retrieved items, using the proposed item recommendations (Peña Henríquez & Carrillo, 2008). These recommendations are generated from other user opinions on certain items or from the user profile and item description, leading to the two major RS approaches (Yager, 2003): collaborative-based or content-based. The former group tries to find, for a given user, those users with similar interests, rating new products or recommending new items to the user from similar user profiles.

The second group generates a profile of the user from their previously selected items and takes those items closer to this profile, which is characterized by item features rather than by similarities with other users. These systems are able to evaluate and filter the great amount of information available on the Internet to help users in their search and retrieval information processes (Hererra-Viedma, Herrera, Martínez, Herrera, & López, 2004). This is the reason why recommender systems have been so relevant to many commercial activities, like tourism (Ricci, 2002) or e-commerce (Schafer, Konstan, & Riedl, 1999) for more than a decade.

In this paper, a proposal for the application of Sentiment Analysis (SA) in recommender systems is detailed. First, the relation between comments and ratings is explored, to justify the consideration of to do comments as a valuable source of information. Then, a strategy for incorporating this knowledge is proposed. This approach categorizes users into two distinct groups: optimists and pessimists. The rest of experiments analyze how these categories can be used in collaborative filtering methods and how to perform this categorization using sentiment analysis solutions.

In order to perform these experiments, a corpus with both comments and ratings on a large set of items and users is needed. Main corpora known by the recommender systems community do not include textual opinions. Thus, a new corpus has been built from the Internet Movie Database (IMDb). Some details on the generation of this corpus are explained also in this paper.

The rest of the paper is organized as follows. In Section 2 a brief review on the state of the art in opinions mining and collaborative filtering is provided. Then, Section 3 describes the main corpus features and its generation. In the next section a walk through all the experiments performed allows the reader to understand how
valuable textual information can be and how it has been used in collaborative filtering algorithms. Finally, in Section 5 we highlight the different contributions of this work and future tasks to continue this line of research.

2. State of the art

Recommender systems (Ricci, Rokach, & Shapira, 2011) mainly attend to two kinds of problems: rating prediction and item recommendation. Rating prediction is focused on automatically calculate the score that a given user would assign to a given item, not known (or seen, bought . . . ) by this user. Item recommendation is an extension of the former, but proposing new products to the user that may satisfy him/her expectations. Basically, both problems are treated similarly. The first recommender systems were developed in the early 90s, introducing collaborative filtering solutions. In 1994, the first workshop was held in Berkeley, and the utility of the first simple algorithms of this type was proved (Foltz & Dumais, 1992; Resnick, Iacovou, Suchak, Bergstrom, & Riedl, 1994; Stodolsky, 1990).

The pioneering project in the field of collaborative filtering was the GroupLens project at the University of Minnesota (Galañ, 2007). It is still very active and has provided much of the basis of many algorithmic recommender systems. They were the first to introduce automated collaborative filtering, using a neighborhood search algorithm to provide predictions on USENET newsgroups.

From the collaborative filtering point of view, recommender systems profit from the selections of a community of users to suggest content of interest to individuals in that community (Resnick & Varian, 1997).

Another definition states that recommender systems are information filtering systems that solve two specific problems: (i) the problem of prediction, i.e., predicting whether a particular user will like a particular item, and (ii) the problem of recommendation, by determining a set of items to be proposed to a user based on personal preferences (Sarwar, Karypis, Konstan, & Riedl, 2001).

2.1. Opinion mining

In the social Web we often find web pages that allow a user to give a score and/or a review to a product, item or service. With the score (rating) it is possible to apply RS solutions, but the score is not always the best way to measure the affinity of the user to the item. Besides, ratings are not always available in many platforms (for example, in the blogosphere). In these scenarios some RS solutions would be hard to implement, despite the fact actual opinions are in there.

Opinion mining (OM) is a discipline derived from information retrieval and computational linguistics. It works with the analysis and extraction of opinions and feelings expressed in text (Esuli & Sebastiani, 2006; Pang & Lee, 2008). In recent years there is a huge amount of research related to the extraction and analysis of the opinions, mainly on the affective interpretation of such opinions. This field has been called Sentiment Analysis and is being applied in an increasing number of problems, becoming the so-called Sentiment Computing (Cambria & Hussain, 2012). These techniques can be classified into two distinct groups: those that make use of supervised methods, specific to text mining, and the ones that use language resources such as dictionaries or lexical ontologies, thus proposing unsupervised methods through the application of knowledge resources. The first group is pioneered by the work of Pang, Lee, and Vaithyanathan (2002), although currently more complex methods are proposed (Lin, Zhang, Wang, & Zhou, 2012; Tan, Na, Theng, & Chang, 2012). In the second group are of special mention (Boldrini, Balahur, Martinez-Barco, & Montoyo, 2010; Chen, Wang, Nagarajan, Wang, & Sheth, 2012) works. There are also hybrid solutions, applying learning algorithms with attributes obtained through lexical and grammatical resources, being the Twitter1 service an active object of study (Martínez-Cámara, Martín-Valdivia, Ureña López, & Montedo-Ráez, 2013).

2.2. Collaborative filtering

Although content-based recommender systems are still an intensive subject of study, with interesting solutions like the Movie Genome project,2 the majority of recommender systems are collaborative (Bafoutsou & Mentzas, 2002; Schafer, Konstan, & Riedl, 2002; Bell & Koren, 2007). A collaborative RS uses the knowledge implicit in a community of users with their preferences on offered items to determine the relevance of these items to other users within the community that have not expressed any preference on the proposed items. Collaborative filtering methods operate on the gathering and analysis of a large amount of information on user behavior, activities or preferences, in order to predict what they would like based on their similarity with other users.

The information on the community can be represented as a matrix of users and items, wherein each cell represents the rating given by a user to a particular item. These matrices are commonly generated even without rating information, relying only on the fact that the user purchased or acquired a product. In this way, it is possible to model users as vectors of items and, of course, the other way around, items as vectors of users (Blanco, 2007). Thus, distance metrics can be applied to calculate the affinity of preferences among users. The rationale behind is clear: I likely will be interested in those products that are of interest to those users with preferences similar to mine ones. However, collaborative filtering suffers from some major problems:

- **Cold start.** These systems often require a large amount of existing data on a user to make precise recommendations, caused by the high sparsity of the user-item matrix. Therefore, they can be applied only when the community is mature and has reached a critical mass.
- **Scalability.** In many environments in which these systems perform recommendations, there are millions of users and products, so it is a challenging task from the point of view of demanded computing resources.
- **Shortage.** Even the most active users tend to value only a small subset of the total database, so even the hottest items may have very few ratings. This has to do also with the fact that newly added items are poorly represented in the matrix.
- **Rating bias.** Users do not show common criteria when rating, or even there can exist users with clear opinion bias or tendencies, resulting in potential noise when interpreting the ratings (Dellarocas, 2000).

In any case, collaborative filtering has strongly placed its position in recommender systems and, with the advent of new social environments, where collaboration among users is a key aspect, new challenges arise (Marlin, Zemel, Roweis, & Slaney, 2007; Kuang & Kuang, 2013).

2.3. Combining opinion mining and collaborative filtering

There are few attempts to apply opinion mining or sentiment analysis techniques in recommender systems. The work by Leung

---

1 Available at http://twitter.com.
et al. already points out the potential benefits of integrating sentiment analysis solutions for improving collaborative filtering approaches (Leung, Chan, & Chung, 2006). They propose two possible solutions: to consider polarity (negative or positive orientation of opinions) as a feature where no user ratings are available, or to modify existing collaborative filtering equations to incorporate opinion distances among users. Although inspiring, the paper does not show any implementation of these ideas and, therefore, no evaluation on them is given, ending up with just hypotheses without contrasts.

Melville, Mooney, and Nagarajan (2002) consider user comments as a bag-of-words to generate additional features and to use them with other item information for a content-based recommender system on movies. The use of user reviews is not unusual in this type of recommender systems, as content-based ones attempt to model users and items on their explicit information (Burke, 2002). Text Mining techniques perform a more exhaustive processing of the textual information when this information is available and can be used to effectively model items and use these models for performing, again, content-based recommendations (Aciar, Zhang, Simoff, & Debenham, 2007).

There are several recommender systems based on the IMDb data that perform suggestions on movies to users applying collaborative filtering (Debnath, Ganguly, & Mitra, 2008), and even some recent work on using textual data for rating prediction (Oghina, Breuss, Tsagkias, & Rijke, 2012). The analysis of the style in reviews can be applied to detect which users are more “useful”, i.e., with more informative reviews (Otterbacher, 2012). Other approaches go even further, by generating user profiles based on their social network (Alves, Freitas, Moura, & Souza, 2013). The approach by Alves et al. explores the possibilities of the social graph to generate more advanced user profiles.

Our work goes beyond Leung ideas and materializes them by integrating polarity classification into collaborative filtering algorithms.

3. IMDb corpus

In order to perform the experiments, it is needed a corpus to train and test a recommender system (items rated by users) but incorporating textual reviews or opinions, so sentiment analysis approaches can be applied on these pieces of texts given by users on items. The Internet Movie Database 4 (IMDb) is a great online database that provides information on movies. It started in 1990 as a hobby by a group of fans of movies and TV shows. IMDb provides a selection of movies in different aspects (genres, directors, producers, players…). It allows registered users to express their opinions and ratings on movies, TV series and shows. It currently stores more than 2 million movies, from 1880 to 2019 (covering even coming productions). As shown in the previous section, the IMDb has been a subject of study due to its valuable amount of data, being considered the main source in many research works on recommender systems.

For instance, Pang and Lee (Pang et al., 2002) created data from IMDb for sentiment analysis, 4 and it has been used in many other works. This data consists of 752 negative and 1301 positive reviews, with a total of 144 reviewers represented. It has unlabeled HTML files from the IMDb archive of the rec.arts.movies.reviews newsgroup, with a limit of fewer than 20 reviews per author per sentiment category. It is not useful for our system because it only contains the author rating, not the user’s comments neither opinions.

3.1. Corpus creation

As textual data (reviews) is needed, none of the available corpora constructed from the IMDb data satisfied our requirements. Thus, a new corpus had to be built. An automatic extraction program was launched, starting from the updated lists of movies that the IMDb delivers in plain text, 5 compiled for movies already released. Our extraction system retrieves for each movie user’s ratings and user’s comments or opinions.

Several data cleansing procedures are performed on the extracted data focused in minimizing sparsity and bad represented users (with less than five rated movies) and movies (rated by less than five users). This process ended with the following data:

- 2713 movies (a 14.37% compared to the original retrieved data),
- 4112 users (a 3.6% compared to the original retrieved data),
- 80,848 opinions (a 34.8% compared to the original retrieved data).

The main reason for this reduction on the collection size is to ease experimentation, avoiding a high demand of resources. Although the data in the IMDb is very sparse (few comments in most movies, few movies rated by most users), sparsity is also present in this purged version. Anyhow, the same data has been used along the whole experimentation process, so results should be consistent.

The collection has an average of 28.8 opinions per movie and 19.66 per user. Fig. 1 shows this average number of opinions per movie or user. Each movie or user has at least five opinions. The movie with most opinions has 335, and the most active user has 735 opinions.

On the other hand, an average ratings per movie is of 6.24 and per user is of 6.43. Table 1 and Fig. 2 show the average, maximum and minimum rating per film and user.

4. Rating prediction experiments

One of the tasks that solve recommender systems is the prediction of the score (named as rating prediction). Collaborative filtering algorithms used in recommender systems usually do not pay attention on textual information. With the aim of checking whether user reviews are helpful in this task. We perform a series of experiments that allow us to answer the following questions in a sequence that defines the rationale behind our study:

\footnote{Available at http://www.imdb.com/}
\footnote{Available at http://www.cs.cornell.edu/people/pabo/movie-review-data/}

![Fig. 1. Average number of comments.](image-url)
1. Is there an implicit relationship between a user’s comments and expressed ratings? If so, it is possible to think on profiting from the information contained in these reviews.

2. How do perform collaborative filtering algorithms in rating prediction? This would allow us to establish a base line of results to beat by integrating textual information in some way.

3. By categorizing users into two groups, pessimist and optimists, based on their average ratings… can previous results be improved?

4. It is possible to identify these groups relying only on textual reviews? Here, sentiment analysis techniques are applied and evaluated considering rating-based categorization as the gold standard.

5. Finally, if our users can be categorized using their comments, will these newly created attributes lead to better performance? Answering the last question is the main goal of this work.

All experiments were carried out using RapidMiner\(^6\) with the Text Processing and Recommender Systems extensions. The latter has been developed within the European project e-Lico\(^7\) and provides operators for model training and evaluation on rating prediction and item recommendation. The former extension is directly available from Rapid-I web site.\(^8\)

The evaluation of strategies proposed in our experiments are based on two different estimators: RMSE (Root Mean Square Error) and MAE (Mean Absolute Error).

The RMSE value is the mean square error. With a set of \(n\) real scores \(y_i\) and their score estimates \(\hat{y}_i\), the formula for calculating the RMSE value is:

\[
\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{n}}
\]

The MAE value is simply the mean absolute error. The calculation of this estimate is performed using the following formula:

\[
\text{MAE} = \frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{n}
\]

These two measures are commonly used when evaluating recommender systems. The first one is based on the root squared error and the second one on the averaged absolute error. RMSE is more appropriate when the error distribution follows a Gaussian distribution, as is affected when heavy tailed errors are found. In that case, MAE seems a more robust choice in those cases (presence of outliers). We have not performed an descriptive analysis of the error distribution, so that’s why we offer both measures.

4.1. Computing the rating from user reviews

In this section we attempt to answer the first question written in Section 4: “Is there an implicit relationship between a user’s comments and expressed ratings?”

In this set of experiments text mining was carried out, with a normal processing flow to obtain, from the opinion texts, the vectors associated after tokenization, stop-words removal and application of stemming. The obtained vectors are converted into feature vectors used to generate predictive models of the score by supervised learning algorithms. Specifically, we applied k-nearest neighbors (KNN) and Support Vector Machines (SVM) (Joachims, 1998). Table 2 and Fig. 3 shows the results obtained using different number of neighbors in the KNN algorithm and the result obtained by SVM with a kernel function based on the dot product operator, which is the default configuration. The evaluation framework is a 10-fold cross validation with stratified sampling, with final averaging on errors measured. This type of validation ensures the statistical significance (Kohavi, 1995) and is considered on all the experiments performed hereafter.

We used the Euclidean distance between neighbors (thus, the proximity is equivalent to a cosine function). As shown in this table, given the values of RMSE and MAE obtained and given that corpus scores are values in the range [0,10], we can consider that predictive models are reliable, because the error does not reach 20% of the possible range. That is, we can predict the score (number of stars) that a user assigns to a movie with a maximum error of about two stars. It can also be derived from the results that a number of neighbors between 20 and 30 proved to be the most appropriate (although with little variation in the range between 10 and 100). It is clear that the SVM algorithm improved KNN in any configuration explored.

Table 1

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Average</th>
<th>Maximum</th>
<th>Minimum</th>
</tr>
</thead>
<tbody>
<tr>
<td>KNN-1</td>
<td>2.6230</td>
<td>1.8580</td>
<td></td>
</tr>
<tr>
<td>KNN-10</td>
<td>2.0042</td>
<td>1.5619</td>
<td></td>
</tr>
<tr>
<td>KNN-20</td>
<td>1.9847</td>
<td>1.5757</td>
<td></td>
</tr>
<tr>
<td>KNN-30</td>
<td>1.9971</td>
<td>1.3918</td>
<td></td>
</tr>
<tr>
<td>KNN-40</td>
<td>2.0869</td>
<td>1.5777</td>
<td></td>
</tr>
<tr>
<td>KNN-50</td>
<td>2.0264</td>
<td>1.6132</td>
<td></td>
</tr>
<tr>
<td>KNN-60</td>
<td>2.0428</td>
<td>1.6266</td>
<td></td>
</tr>
<tr>
<td>KNN-70</td>
<td>2.0631</td>
<td>1.6399</td>
<td></td>
</tr>
<tr>
<td>KNN-80</td>
<td>2.0732</td>
<td>1.6481</td>
<td></td>
</tr>
<tr>
<td>KNN-90</td>
<td>2.0790</td>
<td>1.6508</td>
<td></td>
</tr>
<tr>
<td>KNN-100</td>
<td>2.0882</td>
<td>1.6602</td>
<td></td>
</tr>
<tr>
<td>KNN-200</td>
<td>2.1699</td>
<td>1.7293</td>
<td></td>
</tr>
<tr>
<td>KNN-500</td>
<td>2.2845</td>
<td>1.8350</td>
<td></td>
</tr>
<tr>
<td>SVM</td>
<td>1.8687</td>
<td>1.4686</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 2. Average ratings in the collection.

---

\(^{6}\) Available at http://www.rapidminer.com/.

\(^{7}\) Available at http://www.e-lico.eu/.

\(^{8}\) Available at http://rapid-i.com/content/view/202/206/lang,en/#text.
4.2. Rating prediction through collaborative filtering

The second set of experiments try to answer the second question: “How do perform collaborative filtering algorithms in rating prediction?”. We analyze the performance of collaborative filtering by using KNN algorithm. Again, the Euclidean distance is used to compute the proximity of neighbors.

A variable number of neighbors is also explored and two common variants in collaborative filtering configurations have been accomplished: user-based and item-based. In the first case, vectors of users are generated with viewed movies as dimensions, where in the second case movies vectors from users are considered for computing the final rate of the 10 test sets generated.

Thus, features for item-based collaborative filtering are 1 for the user who actually watched (commented) the movie, and 0 when there is no comment on the user. This is easily encoded with a list of users for each movie. In the case of user-based collaborative filtering is the reverse: users are represented by the list of movies reviewed. (see Tables 3 and 4).

Figs. 4 and 5 show graphically the results.

It seems that the textual content in reviews, as stated in previous experiments, is better when predicting a rating on the movie for the author of the review, but as pointed before, that never occurs in recommendation scenarios. On this scenario, user-based performed better than item-based. This can be due to the larger number of items over users, so movies are better modeled as vectors of users than the other way around. We now move forward in improving these algorithms adding some characterization for users, first by using rating behavior, then by sentiment analysis of their authored reviews.

4.3. Rating prediction with user categorization through collaborative filtering

The KNN algorithm applied in our previous experiments allows the consideration of additional attributes on users and items (depending whether the user-based or the item-based approach is applied respectively). We have designed a new attribute on users with two possible values: 'optimist' or 'pessimist' to characterize the bias of users in their ratings, i.e., the tendency to evaluate negatively or positively. Before trying sentiment analysis on comments, we have performed this classification in basis of the average rating value for each user on all the rated movies. If this average is lower than 4 stars (0–3 stars), then the user is categorized as ‘pessimist’. If this average is above 6 stars (7–10 stars), then we consider the user as an ‘optimist’ person on its movie related criteria. We do not consider the number of rated movies per user to assign the category, there is not a cut point.

With this set of experiments we answer the third question: “Can results be improved by categorizing users into pessimist and optimist?”

Table 3

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>RMSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>KNN-5</td>
<td>2.226</td>
<td>1.649</td>
</tr>
<tr>
<td>KNN-10</td>
<td>2.178</td>
<td>1.616</td>
</tr>
<tr>
<td>KNN-15</td>
<td>2.166</td>
<td>1.610</td>
</tr>
<tr>
<td>KNN-25</td>
<td>2.160</td>
<td>1.607</td>
</tr>
<tr>
<td>KNN-50</td>
<td>2.160</td>
<td>1.610</td>
</tr>
<tr>
<td>KNN-80</td>
<td>2.160</td>
<td>1.610</td>
</tr>
</tbody>
</table>

Table 4

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>RMSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>KNN-5</td>
<td>2.268</td>
<td>1.724</td>
</tr>
<tr>
<td>KNN-10</td>
<td>2.208</td>
<td>1.676</td>
</tr>
<tr>
<td>KNN-15</td>
<td>2.191</td>
<td>1.662</td>
</tr>
<tr>
<td>KNN-25</td>
<td>2.178</td>
<td>1.651</td>
</tr>
<tr>
<td>KNN-50</td>
<td>2.173</td>
<td>1.647</td>
</tr>
<tr>
<td>KNN-80</td>
<td>2.172</td>
<td>1.647</td>
</tr>
</tbody>
</table>
According to this criteria, 84 users were categorized as pessimists, whether 1462 were considered optimists. This is an interesting finding, as it explains why collaborative filtering works well on this dataset: when people comment a movie it is because they watched it, and people tend to view movies they know would be rather enjoyed. Therefore, just the fact that the user rates or comments a movie could be considered as a favorable relation.

The configuration of the experiment is the same as for the best algorithm used in collaborative filtering with user-based KNN (80 neighbors and cosine distance).

Table 5 shows the results obtained with and without the new user attribute. The results show that using information about user behavior the error value is reduced slightly.

4.4. Polarity classification for user characterization

With the next set of experiments we want to check if the system could identify if a user is an optimist or pessimist person making use of comments or reviews, to answer the fourth question: “Is it possible to identify these groups relying only on textual reviews?”

Experiments in Section 4.1 showed that user comments are related to the rating score assigned to movies. Therefore, polarity classification on these reviews could also be used to determine user category into the proposed pessimist and optimists classes. Before that, we wanted to explore the affinity of our polarity classification algorithm with the pessimist and optimist labels based on ratings as golden rules. If the polarity average on user comments classifies the user into the given two sets as the ratings do, we could expect that the polarity can be straightforward considered as a user categorization strategy, in order to improve collaborative filtering as seen above.

To obtain the user polarity we have classified all the comments of every user as positive or negative and, then, the polarity average was calculated. We have used the resource WeFeelFine (WFF) (Kamvar & Harris, 2011) and the algorithm for polarity classification in English explained in (Montejo-Ráez, 2013), which proposes a polarity classification solution without the need of a learning phase on manually labeled texts.

WeFeelFine is constantly collecting affective sentences from different blogs and other social media sources. These sentences are labeled with a “feeling”. What the polarity classification algorithm does is, at a first step, to generate a document per feeling, composed by all the sentences labeled with that feeling found by WeFeelFine (it has an API9 available to obtain such information). Then, these documents are indexed by the Lucene search engine tool (see Fig. 6).

Once the feelings have been indexed, we can use each review as a query (see Fig. 7) and retrieve a ranked list of feelings, as the one show in Table 6 as result for the text “The Nike Training Club beta iPhone app looks very interesting”.

From that list of feelings with associate Ranked Status Value

<table>
<thead>
<tr>
<th>Rank</th>
<th>RSV</th>
<th>Feeling</th>
<th>Polarity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.0516</td>
<td>Cool</td>
<td>+1</td>
</tr>
<tr>
<td>2</td>
<td>0.0401</td>
<td>Dumb</td>
<td>–1</td>
</tr>
<tr>
<td>3</td>
<td>0.0314</td>
<td>Lucky</td>
<td>+1</td>
</tr>
<tr>
<td>4</td>
<td>0.0303</td>
<td>Awesome</td>
<td>+1</td>
</tr>
<tr>
<td>5</td>
<td>0.0296</td>
<td>Fine</td>
<td>+1</td>
</tr>
<tr>
<td>6</td>
<td>0.0294</td>
<td>Used</td>
<td>–1</td>
</tr>
<tr>
<td>7</td>
<td>0.0288</td>
<td>Low</td>
<td>–1</td>
</tr>
<tr>
<td>8</td>
<td>0.0278</td>
<td>Missing</td>
<td>–1</td>
</tr>
<tr>
<td>9</td>
<td>0.0270</td>
<td>Complete</td>
<td>+1</td>
</tr>
<tr>
<td>10</td>
<td>0.0268</td>
<td>Proud</td>
<td>+1</td>
</tr>
</tbody>
</table>

9 Available at http://wefeelfine.org/api.html.
10 Available at http://lucene.apache.org/.

Fig. 6. Indexing process.

Fig. 7. Classification process.
(RSV, the relevance value found by the search engine), and polarity values of the feelings we can obtain a final polarity value for the text according to Eq. (1):

$$p(t) = \frac{1}{|R|} \sum_{i \in R} RSV_i - l_i$$  \hspace{1cm} (1)

where \(p(t)\) is the polarity of text \(t\), \(R\) is the list of retrieved feelings, \(l_i\) is the polarity label of feeling \(r\) and \(RSV_i\) is the Ranked Status Value. The polarity value of each feeling has been assigned manually over the 200 most frequent feelings of WeFeelFine, so just 200 feelings were indexed by the engine.

Regarding the evaluation of the polarity score computed, we have measured the Accuracy, which is a traditional measure employed in text classification (see Eq. (2)); Accuracy is the opposite of error rates. It computes the proportion of true results over the total number of assignments (Sebastiani, 2002). In Eq. (2), \(TP\) (True Positives) are those positive assessments that were correct and \(TN\) (True Negatives) are those negative assignments that were also considered negative in by the gold rule.

$$\text{Accuracy} = \frac{TP + TN}{\text{total assignments}}$$ \hspace{1cm} (2)

In our experiments, we have obtained a value of 80.05% of accuracy. Therefore, only 20% of users were wrongly classified as optimists or pessimists. With these results, it is possible to expect a profitable effect on performance when incorporating this informative attribute from textual reviews.

### 4.5. Rating prediction through collaborative filtering and calculated polarity

Having calculated the polarity attribute for all users, we repeated the experiments of Section 4.3. This time we use the polarity attributes calculated from the user comments. For each review, a polarity value is computed through sentiment analysis, then, these values are averaged. The final polarity value obtained for the user determines whether the user is pessimist or optimist (as neutrality is not possible outcome). In this way, we found that 162 users were considered as pessimists, against a total of 3984 optimist users. It is important to note that our system always will categorize the user into pessimist or optimist, compared to the previous rating-based approach. Again, most of the users are found to be within the group of optimists. The rest of the experimental setup is the same as in the previous block of experiments.

This time we use the polarity attributes calculated from the user comments, in order to answer the last previous question “will these newly created attributes lead to better performance?”. Also, further common algorithms in collaborative filtering have been launched against the data, to establish a better comparison among solutions and to ensure the benefits of our approach. The algorithms considered are Slope One and Bipolar Slope One (Lemire & Maclachlan, 2005), Biased Matrix Factorization (Gemulla, Nijkamp, Haas, & Sismanis, 2011), Factor Wise Matrix Factorization (Bell, Koren, & Volinsky, 2007) and User Item Baseline (Koren, 2010).

Table 7 compares the results obtained in the experiments with all other algorithms, ranked by the RMSE value obtained. Error values of KNN with the categorization of users by sentiment analysis of reviews (in bold) are higher than those obtained by calculating the polarity from the score which is the best approach of all explored configurations. The last four algorithms do not allow specific user attributes, as KNN does. However, our hypothesis holds as we have found that the analysis of texts by means of its polarity can enhance collaborative filtering algorithms, so in scenarios where no rating is available, the proposed solution is worth exploring.

### 5. Conclusions and ongoing work

The most interesting aspect of collaborative filtering algorithms, compared with well-known text mining approaches, is that we can estimate a user’s score on a movie without having any comment, i.e., compute a distance between the user and the item when there is no relation at all. This is what really makes these algorithms very valuable for recommending new items, while the previous solution cannot recommend new products as we cannot know the opinion of a user previously. But the results indicate that we can obtain a more reliable prediction considering only the opinion text (RMSE of 1.868), than when apply similarities over the entire user community (RMSE of 2.134).

The use of the average polarity on the comments of every user has shown to be valuable in improving collaborative filtering algorithms, so the answer of our fifth question is that Sentiment analysis can be advantageous to recommender systems.

We are currently developing a system with a higher level of integration, changing the between-items and between-users similarity in the core of collaborative filtering algorithms. This will also take into account the distances between the opinion texts and the sentiments expressed in these opinions, integrating polarity values from comments as factors in inter-vector distances.

### References


