Combining a Content Filtering Heuristic and Sentiment Analysis for Movie Recommendations

Vivek Kumar Singh, Mousumi Mukherjee, and Ghanshyam Kumar Mehta

Department of Computer Science,
Banaras Hindu University, Varanasi, India
vivek@bhu.ac.in, {mou.sonai,ghanshyam4u2000}@gmail.com

Abstract. This paper presents our experimental work on a new approach to hybrid recommender systems. Traditionally recommender systems are categorized into three classes: content-based, collaborative filtering and hybrid approaches. The hybrid approach to recommender systems combines both content-based and collaborative filtering schemes to obtain better recommendations to harness advantages of both the approaches. We have experimented with an alternative formulation, where we combined the content-based approach with a sentiment analysis task to improve the recommendation results. The final recommended list contains only those items that are both similar in content to the items liked by the user in the past, and also labeled as ′positive′ on sentiment classification. The paper concludes with a discussion of experimental results on Movie review domain and usefulness of this work.

Keywords: Content-based Recommendations, Movie Reviews, Opinion Mining, Recommender Systems, Sentiment Classification.

1 Introduction

A recommender system is a program that automatically filters and presents items of interest to users. The increased penetration of the Internet and the new participative *Web 2.0* has facilitated a large number of users to use and interact with web applications. The users are now interacting with web applications in a more participative way by contributing in a variety of forms, such as rating, tagging, writing blogs, social networking and sharing items with friends. This increased interaction is generating huge amount of information. This information overload often becomes problematic for users when they try to search and find a desired item on the *Web*. Recommender systems address this problem and provide personalized recommendation to users. *Amazon, Netflix, Google News personalization* and *MovieLens* are some of the popular industry scale recommender systems. There are three kinds of recommender systems: content-based, collaborative filtering and hybrid approaches. Content-based methods memorize a user's preferences and recommend him those items that are similar in content to the ones liked by him in the past. Collaborative filtering also recommends items to a user based on his past transaction history, however, it does not take into account the content of the items for recommendation decision. It treats items as black box and generates recommendations on the basis of either item-to-item similarity or user-to-user similarity. In item-to-item similarity a user is recommended those items that are similarly rated (by other users) to the items liked by the user in the past; whereas in user-to-user similarity a user is recommended those items that are liked by the other users who are similar (in terms of their rating preferences) to the user. Collaborative filtering requires data about user transaction history (browsing, rating, tagging etc.,), which however may not be readily available.

Both content-based and collaborative filtering have some disadvantages. A hybrid method (the third approach) combines content-based and collaborative approaches to minimize the disadvantages and produce better results. In this paper, we present an alternative kind of hybrid recommender system which combines content-based method with sentiment analysis to improve the recommendations in movie recommendation domain. Sentiment analysis labels each item as 'positive' or 'negative'. We collected data about a large number of movies including their name, description, genre and 10 user reviews each from IMDB [1] and then applied both, a content-based filtering to obtain a first level list of movies of interest to a user and then the sentiment analysis to label every movie in the first level result as 'positive' or 'negative', based on their user reviews. The final recommended list contains only those movies which match with user's interests and are also labeled as 'positive'.

2 Content-Based Recommendation

Recommender systems aim to show items of interest to a user. In order to do so a recommender system may exploit the following inputs: (a) user's profile - attributes like age, gender, geographical location, net worth etc., (b) information about various items available - content associated with item, (c) interactions of users - such as ratings, tagging, bookmarking, saving, emailing, browsing history etc., and (d) the context of where the items will be shown. These inputs may be used by a recommender system to produce following kinds of results: (a) users who watched this video watched these other videos, (b) new items related to a particular item, (c) users similar to a particular user, and (d) the products a user may be interested in. The recommendation problem can be formally defined as follows: Let U be the set of all users and I be the set of all possible items that can be recommended. The set I of items can be very large, with possibly millions of items. And similarly the set U of users can also be very large with millions of users in some cases. Let r be a recommender utility function that measures the usefulness of an item i to a user u, i.e., $\mathrm{r} : \mathrm{U} \ \mathrm{X} \ \mathrm{I} \longrightarrow R$, where R is a totally ordered set of non-negative integers or real numbers. Then the recommendation problem is to choose for each user $u \epsilon U$, such items $i \epsilon I$ that maximizes the users utility, i.e., $i_u = \arg max_{i \epsilon I} \mathrm{r} (\mathrm{u}, \mathrm{i})$. The utility usually refers to some rating value associated with different items [2]. However, the central problem is that the recommender utility r is usually not defined on the whole of U X I space, but only on some subset of it. Therefore, the recommender system needs to extrapolate from known to unknown ratings. This extrapolation can be done in a number of ways using machine learning, approximation theory and heuristics.

In content-based approach a user is recommended those items that are similar in content to the items liked by him in the past. The items in the set I are compared with the items liked by the user in the past and only the best matching items are recommended. The items in content-based recommendation approach are textual in nature, which need to be represented using appropriate data structure. A commonly used scheme is the vector space model [3],[4], in which every text document is represented as a term vector. A term vector consists of the distinct terms appearing in the document and their relative weights. The weight associated with each term could be either a *tf* measure or a *tf-idf* measure. The vector *V(d)* derived from the document *d* thus contain one component for each distinct term in the entire vocabulary of the text space. Once we have all texts represented as document vectors, their similarity can be computed using cosine similarity measure (Eq. 1) as follows:

$$CosineSimilarity(d_1, d_2) = [V(d_1).V(d_2)] \, / \, [|V(d_1)||V(d_2)|] \, . \qquad (1)$$

The content-based recommendation problem is thus to obtain $r(u,i)$ = score [content-based profile (u), content(i)]. This is done for all items $i \epsilon$ I and the best matching items are recommended. We have performed content matching in two ways: (a) through a heuristic vector for new users, and (b) through cosine-based similarity computation for a known user whose rating profile is known.

3 Sentiment Analysis

The sentiment analysis problem can be formally defined as follows: Given a set of documents D, a sentiment classifier classifies each document d ϵ D into one of the two classes, *positive* and *negative*. Positive means that d expresses a positive opinion and negative means that d expresses a negative opinion. The sentiment analysis task usually employs one of the two approaches: (a) using a text classifier - such as Naive Bayes, SVM or kNN- that takes a machine learning approach to categorize the documents in positive and negative groups; and (b) using an unsupervised semantic orientation approach that computes sentiment of documents based on aggregated semantic orientation values of selected opinionated POS tags in it. Some of the prominent research works on sentiment analysis can be found in [5-9]. We have used the unsupervised semantic orientation approach for classifying texts as $'$positive$'$ and $'$negative$'$.

In semantic orientation approach we first extract selected POS tags that conform to a specified pattern [10]. Thereafter the semantic orientation of extracted phrase is computed using the Pointwise Mutual Information (PMI) measure as in Eq. 2,

$$PMI(term_1, term_2) = log_2 \, [Pr(term_1 \bigtriangleup term_2)/Pr(term_1).Pr(term_2)] \, . \quad (2)$$

where, *Pr(term*$_1$ $\triangle$ *term*$_2$*)* is the co-occurrence probability of term1 and term2 and *Pr(term*$_1$*) . Pr(term*$_2$*)* gives the probability that two terms co-occur if they are statistically independent. The ratio between *Pr(term*$_1$ $\triangle$ *term*$_2$*)* and *Pr(term*$_2$*).Pr(term*$_2$*)* measures the degree of statistical independence between them. The log of this ratio is the amount of information that we acquire about the presence of one word when we observe the other. The Semantic Orientation (SO) of a phrase can thus be computed by using Eq. 3 below,

$$SO(phrase) = PMI(phrase,''excellent'') - PMI(phrase,''poor''). \quad (3)$$

where, PMI (phrase, ″excellent″) measures the association of the phrase with positive reference word ″excellent″ and PMI (phrase, ″poor″) measures the association of phrase with negative reference word ″poor″. These probabilities are calculated by issuing search query of the form ″phrase * excellent″ and ″phrase * poor″ to search engine. The number of hits obtained is used as a measure of probability value. To determine the semantic orientation of the entire document, the SO values of the opinionated phrases in it is aggregated, and if the average SO is above a threshold value the document is labeled as ′positive′, and ′negative′ otherwise.

4 Experimental Setup and Results

We have designed our hybrid recommender system for use in Movie review domain. We collected a total of 2000 movie reviews (10 reviews each for 200 movies) and applied our hybrid design to obtain relevant recommendations for users. For every movie, we collected its name, genre, year of release, its rating, and 10 user reviews. The genre of a movie is an indicator of its type such as action, comedy, drama etc. For every review written by a user we also recorded the name (id) of the reviewer. Movie recommendation involved a two level filtering process using content-based similarity at first level and sentiment analysis at second level. Since the collected review data was textual, we transformed it into vector space model. First we preprocessed the data to remove stop words (words like ′is′, ′am′, ′are′, ′to′, ′from′ etc.). Then every review was transformed into a term vector. For new users we used the term values rather than their frequency.

4.1 Computing Content Similarity

We used content-based approach as our first level of filtering. In case the recommendation is to be generated for an existing user who has already reviewed some movie in the dataset, we compute cosine-based similarity between user's reviews with the review sets of all the movies. All the movies having cosine similarity value above a threshold are included in first level filtered results. We kept the threshold low (around 0.3) to obtain a good number of movies. When the recommendation is to be made for a new user, we ask the user about what kind (genre) of movies he is interested in. The genre information entered by the user is used to employ a predefined query vector as a heuristic and compare it with the review vectors of all the movies. The query vector for a genre contains its synonyms and such other terms that are frequently used by reviewers in positively reviewing that kind of movies. All the movies having terms in the query vector present in their review are included in the first level filtered result.

4.2 Sentiment Analysis

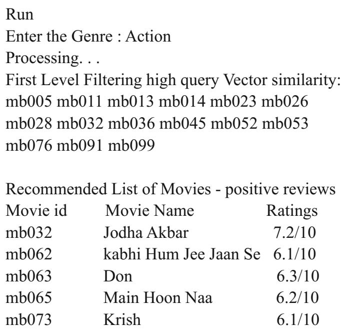
We used SO-PMI-IR algorithm as described in section 3 to compute the sentiment of a user review. The overall sentiment of a movie was aggregate of the sentiment labels of the 10 reviews of that movie, i.e., a movie was labeled as ′positive′ only if at least 6 (or say 7) of the 10 reviews for that movie was labeled as ′positive′. Every review is labeled as positive or negative based on the aggregate SO value of the selected POS tags in that review. Every term having SO value greater than a threshold (say 1) adds ′+1′ to aggregate score of the review and every term having SO value below that adds ′-1′ to the aggregate score. These values are then added and if the aggregate score for a review is greater than a given threshold value, we label that review as ′positive′. This is done for all review vectors and every movie is then accordingly labeled. The final list of recommended movies, from among the first level filtered result, then contain only those movies which are labeled as ′positive′. This is logically equivalent to recommending those movies which have been appreciated by a good number of users. We have verified the accuracy of sentiment labeling of movie reviews by SO-PMI-IR algorithm through a Naive Bayes machine learning classifier, trained on another popular movie review dataset [11].

4.3 Results

We obtained interesting results. While the first level filtering produced a list of interest; the second level of filtering (using sentiment analysis) produced final list of recommended movies. The recommended movies were also rated sufficiently high on overall rating. A snapshot a sample run is given in Fig. 1. The final recommended list in virtually all the cases contains movies that have been appreciated by reviewers. The method also has the advantage that it can produce relatively good recommendations for new users as well. The hybrid that we designed thus not simply recommends all the movies that match a user's taste, but only those movies that are also rated high.

5 Conclusions

We designed a new kind of hybrid recommender system that combined content-based recommendation system with sentiment analysis to produce high quality recommendations. We experimented with movie review domain and the results obtained were quite accurate and of high quality. The first level filtering was based on content similarity and the second level filtering labeled a review as ′positive′ or ′negative′. The final recommendation list comprised of those results



```
Run
Enter the Genre : Action
Processing. . .
First Level Filtering high query Vector similarity:
mb005 mb011 mb013 mb014 mb023 mb026
mb028 mb032 mb036 mb045 mb052 mb053
mb076 mb091 mb099

Recommended List of Movies - positive reviews
Movie id     Movie Name               Ratings
mb032        Jodha Akbar               7.2/10
mb062        kabhi Hum Jee Jaan Se     6.1/10
mb063        Don                       6.3/10
mb065        Main Hoon Naa             6.2/10
mb073        Krish                     6.1/10
```

Fig. 1. A sample run of the system for a new user

which rated favourable in both levels of filtering. Further, the number of recommendations can be preset by using suitable threshold values. This new kind of hybrid approach to recommender system can be successfully applied in many other domains which use textual review data, more particularly opinionated texts.

References

1. Internet Movie Database, `http://www.imdb.com`
2. Adomavicius, G., Tuzhilin, A.: Towards the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. IEEE Transactions on Knowledge and Data Engineering 17(6), 734–749 (2006)
3. Alag, S.: Collective Intelligence in Action, pp. 41–48. Manning, New York (2009)
4. Manning, C.D., Raghavan, P., Schutze, H.: Introduction to Information Retrieval, pp. 107–116. Cambridge University Press, New York (2008)
5. Turney, P.: Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. In: 40th Annual Meeting of the Association for Computational Linguistics, Philadelphia, US, pp. 417–424 (2002)
6. Esuli, A., Sebastiani, F.: Determining the Semantic Orientation of Terms through Gloss Analysis. In: 14th ACM International Conference on Information and Knowledge Management, CIKM 2005, Bremen, DE, pp. 617–624 (2005)
7. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs Up? Sentiment Classification Using Machine Learning Techniques. In: Conference on Empirical Methods in Natural Language Processing, Philadelphia, US, pp. 79–86 (2002)
8. Kim, S.M., Hovy, E.: Determining Sentiment of Opinions. In: Proceedings of the COLING Conference, Geneva (2004)
9. Durant, K.T., Smith, M.D.: Mining Sentiment Classification from Political Web Logs. In: Proceedings of WEBKDD 2006. ACM, New York (2006)
10. Liu, B.: Web Data Mining: Exploring Hyperlinks, Contents and Usage Data, pp. 411–416. Springer, Heidelberg (2002)
11. Movie Review Dataset,
`http://www.cs.cornell.edu/people/pabo/movie-review-data/`