NewsRec, a Personal Recommendation System for News Websites

Christian Bomhardt and Wolfgang Gaul

Institut für Entscheidungstheorie und Unternehmensforschung,
Universität Karlsruhe (TH), 76128 Karlsruhe, Germany

Abstract. The behavior of how individuals select and read news depends on the underlying media for reproduction. Today, the use of news websites is increasing. Online readers usually have to click on abstracts or headlines in order to see full articles. This kind of selection of information is less pleasant than in traditional newspapers where glancing over the whole layout of double pages is possible. Personalization is a possible solution for this article selection problem. So far, most types of personalization are controlled by website owners. In our work, we discuss design aspects and empirical results of our personal recommendation system for news websites, which uses text classification techniques.

1 Introduction

The internet can be seen as an enabling technology for the development of innovative product ideas. File sharing services or online news papers are well-known examples. One handicap of web sites in general is the necessity to navigate through them by following hyperlinks - it is, e.g., not possible to quickly run over the pages of an online newspaper - in contrast to printed media. On the hyperlinked internet, information selection is therefore more "expensive" in terms of time required as every click results in a transfer delay. A common approach for this information selection problem is personalization. Mobasher et al. (1999) describe web personalization as "any action that makes the web experience of a user personalized to the user's taste". In Gaul et al. (2002) personalization is used as important feature for the characterization of recommender system output. Personalization methods can be categorized, e.g., as personalization by information filtering (e.g., on Lycos.com, registered users can choose their interests from given categories; on further visits, only information from the selected categories is displayed) and personalization by information supplementing (e.g., Amazon.com enhances the detail view pages of books with recommendations of additional/alternative books) where additional context specific information is provided. These personalization methods can use well structured input data from databases but are solely offered by website operators for their own websites. As not every website offers personalization, a limitation exists. If someone wishes personalized assistance independent of a special website, the help of browsing agents is an alternative. These agents can solely rely on webpages, as databases may not exist or are not accessible by the public. As long as website management does not apply robot detection technologies (cp. Bomhardt et al. (2004)) in order to prevent robots from accessing their websites, browsing agents can be used.

Browsing agents are well-known from the literature (Middleton (2001)). One example is WebWatcher (Joachims et al. (1996)) which is a tour guide for the world wide web and assists users in browsing the www. It learns from the experiences of multiple users and given keywords using term frequency/inverse document frequency (TF-IDF) similarity measures. Letizia (Lieberman (1995)) is an agent that monitors user behavior. During idle times, Letizia autonomously and concurrently explores links available at the user's current position and tries to anticipate items of interest which are displayed upon request. Letizia uses a set of heuristics and TF similarities. While WebWatcher provides tours to many people and learns to become a specialist with respect to a particular web site, Personal WebWatcher (PWW) (Mladenić (2001)) accompanies a single user and considers her/his individual interests. It doesn't ask the user for any keywords or opinions about pages but instead solely records the addresses of pages requested by the user and highlights interesting hyperlinks. During a learning phase, the requested pages are analyzed and a model is built. For speed reasons (Mladenić (1999)), PWW uses the anchor text of a hyperlink and the text near by the hyperlink as input for prediction. The agent NewsWeeder (Lang (1995)) is specialized with respect to Usenet newsgroups. It is implemented as web-based newsgroup client which stores user ratings of articles, learns preferences, and builds personalized news collections. It combines content-based filtering and collaborative filtering while using TF-IDF similarities. WebMate (Chen and Sycara (1997)) is a personal agent for browsing and searching. WebMate automatically sorts documents into categories and tracks interesting ones per category, thus building a domain-specific knowledge base. A document is recommended if it is "close enough" to an interesting reference document of the detected document category. WebMate can spider a user-defined list or URLs to compile a personal newspaper or it can feed search engines with several top key words of the current profile and rate pages found. WebMate also uses TF-IDF similarities.

As we are not aware of any browsing agent that is specialized with respect to news websites, uses support vector machines as prediction method, is designed as a single user system, and is silently augmenting browsing experience, our implementation NewsRec (News Recommender) will be described in the following. In the next section requirements, system design, and implementation details of NewsRec are discussed. The used classification methods and their evaluation measures are described in section 3. Empirical results are presented in section 4. Our findings are summarized in section 5.

2 Requirements, system design, and implementation details

A personal recommendation system for news websites should be compatible with HTML-based news websites, should be usable with any browser, should contain a user-friendly interface that annotates hyperlinks with its recommendations instead of requiring explicit requests for recommendations, should contain domain-specific state-of-the-art prediction models, should be designed for single user application, and should not lead to noticeable delay during web browsing. NewsRec fulfills all mentioned aspects.

NewsRec is implemented as HTTP proxy server. All HTTP requests and responses pass through the proxy server which manages communication between web browser and internet. The interaction between NewsRec and the user (article labeling, requesting model updates) is realized via additional embedded HTML buttons. The user configurates desired hosts for which the recommendation engine should be used. If a webpage from such a website is requested, it is processed by NewsRec's recommendation engine. Otherwise, the request is forwarded to the internet. NewsRec's recommendation engine loads a requested webpage, searches for linked documents, requests and rates the linked documents, marks the links within the original webpage as interesting (+) or uninteresting (-), adds interaction buttons, and sends the page back to the web browser.

As news websites can contain many links sequentially requesting and rating them would be too slow. We addressed this speed problem by using a thread pool which issues many requests in parallel. This approach overcomes the time consuming summation of transfer delays and timeouts that occur if pages are requested sequentially. Another important implementation detail is the usage of a recommendation cache. It stores the rating of every examined webpage. As several webpages within one website can link to the same page, this reduces the number of pages that have to be investigated. Webpages are represented using the common bag-of-words approach. Here, memory saving data structures have to be taken into consideration in order to avoid time-consuming memory swapping. A dictionary maps between words and word IDs. These mappings are used frequently. As dictionaries can easily contain more than 50000 words, fast and efficient data structures are required. We selected a Ternary Tree (Bentley/Sedgewick (1997)) as dictionary and slightly modified it in order to meet our requirements. The thread pool together with the recommendation cache, memory saving data structures, and the fast dictonary structure lead to significant performance gains that enabled the consideration of the contents of linked webpages. It should be noted that the recommendation cache has to be cleared, if the prediction model is updated.

3 Website classification and evaluation measures

A webpage is written in HTML and consists of common text, surrounded by HTML tags. The layout of a usual news webpage contains fixed elements like logos, advertisements, components for navigation, and the text of the news article. Our basic idea is to transform a webpage into common text, on which well known text classification algorithms can be applied. We extract the relevant text - that is the part of the webpage that contains the article text - remove HTML tags like *<B>* and substitute or remove HTML entities like *&uuml; &bpsp;*. Now, we have raw text for which appropriate transformations are required. The following notation is used:

n	number of documents
d_i	document i in text representation, $i = 1...n$
m	number of distinct words contained in all documents d_i $(i = 1...n)$
w_j	unique word j, $j = 1...m$
$TF(w_j, d_i)$	number of occurrences of word w_j in document d_i (term frequency)
$BIN(w_j, d_i)$	$= 1$, if word w_j is contained in document d_i, 0 otherwise (binary)
IDF_j	$= log(\frac{n}{\sum_{i=1}^{n} BIN(w_j, d_i)})$ (inverse document frequency)
R_j	$= log(n) + \sum_{i=1}^{n} \frac{TF(w_j,d_i)}{\sum_{g=1}^{n} TF(w_j,d_g)} log(\frac{TF(w_j,d_i)}{\sum_{g=1}^{n} TF(w_j,d_g)})$ (redundancy)

A document d_i can be represented as document vector $\vec{d}_i = (d_{ij})$, where each component d_{ij} of $\vec{d}_i$ either contains $TF(w_j, d_i)$ (TF-notation), or $log(1 + TF(w_j, d_i))$ (LOG-notation), or $BIN(w_j, d_i)$ (BIN-notation). This first step is called frequency transformation. Term weighting is the next step, where each d_{ij} of the document vector is multiplied by a weight factor. This factor can be 1 (NOWEIGHTS-notation), IDF_j (IDF-notation) or R_j (RED-notation). The last step comprises the normalization of the document vector $\vec{d}_i$. It can be skipped (NONE-notation), or $\frac{1}{\sum_{j=1..m} d_{ij}}$ (L1-notation), or $\frac{1}{\sqrt{\sum_{j=1..m} d_{ij}^2}}$ (L2-notation) can be used. A selection of one frequency transformation, one term weighting, and one normalization scheme describes a preprocessing setting. We have not implemented frequency transformation via BIN as it has lead to poor results in the experiments performed by Cooley (1999). Weighting via RED is expensive in terms of resource usage and, according to Paaß et al. (2004), the advantage of redundancy weighting via IDF seems to be greater for larger documents, thus, we have not included the RED weighting scheme. We selected support vector machines (SVM) (Boser et al. (1992)) for prediction, as different researchers have found out that SVMs are well suited for text classification and outperform other methods like naive bayes classifiers, C4.5, etc. (Joachims (2002), Dumais et al. (1998), Cooley (1999), Sebastiani (2002)).

Recall, precision, and the measure $F1$ ($F1 = \frac{2*recall*precision}{recall+precision}$) were selected as common evaluation measures from information retrieval. Recall is defined as the number of correctly predicted interesting documents divided by the total number of interesting documents. Precision is defined as the number of correctly predicted interesting documents divided by the total number of predicted interesting documents. Good recall values can be easily achieved in expense of poor precision values (think of predicting all documents as interesting) and vice versa. This is why the $F1$-metric is often used in practice. It is a balanced measure and is dominated by the smaller of the recall and precision values.

4 Empirical results

NewsRec was tested on the Heise news ticker (HEISE), which is maintained by the German computer magazine c't. As a first step, one of the authors used the news website during a period of 7 weeks and read and labeled 1265 articles. To avoid self fulfilling prophecies, the recommendation engine was deactivated during this time. 27% of the articles were indicated as interesting by personal inspection. The next step was the simulation and evaluation of a real-world scenario. We assume that a common user labels a certain number of articles and is then interested in the valuation of the next upcoming articles (e.g., based on achieved recall and precision values). Thus, we trained successive prediction models. The first model was trained on the first 50 documents and evaluated on the next 50 documents. The number of training documents was increased by the just evaluated 50 documents for every new model as long as there remained 50 unevaluated documents for the next application step. The achieved recall and precision values were micro-averaged in order to receive an overall prediction measure. This evaluation procedure was repeated for each preprocessing setting.

For our experiments, we fell back on the SVM implementation SVMlight by Joachims (1999) and used the linear kernel. We are aware of the fact that slightly better models based on the radial basis function (rbf) kernel (cp. Paaß et al. (2004) and Joachims (2002)) may exist, but rbf models require extensive fine tuning of model parameters. For an automatic system like NewsRec, the linear kernel turned out to be a good choice as it is less sensitive to parameter selection than the rbf kernel. Another advantage of the linear kernel is its speed.

Table 1 summarizes our findings. TF-IDF-L2 was the optimal preprocessing setting in terms of $F1$ and recall. Users which prefer high precision could select TF-IDF-NONE.

Going into more detail, table 2 contains the detailed recall and precision values for the best three preprocessing settings in terms of $F1$. Here, one can see that model selection is not trivial, e.g., TF-NOWEIGHTS-L2 and TF-IDF-L2 achieve the same recall values, if trained on the first 50 documents. If trained on the first 100 documents, TF-NOWEIGHTS-L2 outperforms TF-IDF-L2 in terms of recall. Taking a look at the next application step with 150 training documents, TF-NOWEIGHTS-L2, LOG-IDF-L2, and TF-IDF-L2 achieve the same recall, but TF-NOWEIGHTS-L2 is outperformed in terms of precision by the two other preprocessing settings.

Another important aspect is the fact that recall and precision values are not constantly increasing but are oscillating up and down, instead. This is a result of the fact that articles on new subjects may come up. Notice, e.g., that the models built on 550 training documents altogether perform very poor. TF-NOWEIGHTS-L2 did not valuate any document as interesting (although nine where contained within the evaluation set) and therefore achieved 0% recall and 100% precision, because no uninteresting document was labeled interesting. LOG-IDF-L2 and TF-IDF-L2 valuated some documents as interesting which indeed were uninteresting which lead to 0% recall and 0% precision. For the 550 training documents, the best prediction quality was achieved with the LOG-NOWEIGHTS-NONE preprocessing setting (not contained in the table): 11,1% recall, 16,6% precision and 0.13 for $F1$. On the other hand, there exist models that achieve 100% recall (LOG-IDF-L2, 650 training documents) or 100% precision (TF-NOWEIGHTS-L2, 1000 training documents). Here, one can see that model selection on the basis of table 2 is a difficult task. Thus, we recommend TF-IDF-L2 according to the overall performance mentioned in table 1.

5 Conclusions and outlook

NewsRec is easy to use and enriches conventional browsing by augmented browsing with real add-on value. The results of similar tools - if reported - cannot be compared with the ones of NewsRec, as the approaches vary and so do the datasets. Nevertheless, we report results concerning WebMate and NewsWeeder to mediate a feeling for what can be expected. WebMate achieved 31% overall precision and NewsWeeder achieved 44% (dataset a) and 59% (dataset b) precision for the highest rated positive 10% of articles. Recall was not reported for these agents. If we use these precision values as benchmarks, then NewsRec - which achieved 49.2% overall recall and 55% overall precision - seems to compete favorable. Our results fall within a reasonable range and were confirmed on another dataset. One problem in the area just mentioned is the requirement of explicit user feedback. Other researchers therefore use implicit feedback (Mladenić (1999)). We think that implicit feedback alone is a weak indicator. Instead, we will address this problem in a forthcoming paper by using a hybrid (implicit and explicit) feedback approach.

Table 1. Micro-averaged prediction quality for different preprocessing settings. Values in parenthesis denote the best results for a single set of evaluation documents

Preprocessing setting	Overall Recall	Overall Precision	Overall $F1$
TF-NOWEIGHTS-NONE	45.01% (80%)	50.01% (85.71%)	0.4738 (0.70)
TF-NOWEIGHTS-L1	12.86% (100%)	40.00% (54%)	0.1946 (0.70)
TF-NOWEIGHTS-L2	48.55% (81%)	49.18% (100%)	0.4886(0.74)
LOG-NOWEIGHTS-NONE	45.98% (80%)	44.96% (82%)	0.4546 (0.67)
LOG-NOWEIGHTS-L1	14.47% (100%)	43.27% (100%)	0.2169 (0.71)
LOG-NOWEIGHTS-L2	47.27% (82%)	48.68% (100%)	0.4796 (0.78)
LOG-IDF-NONE	37.94% (100%)	55.14% (100%)	0.4495 (0.71)
LOG-IDF-L1	21.22% (100%)	50.77% (100%)	0.2993 (0.70)
LOG-IDF-L2	44.69% (100%)	52.85% (100%)	0.4843 (0.71)
TF-IDF-NONE	37.62% (80%)	60.62% (86%)	0.4643 (0.80)
TF-IDF-L1	12.86% (100%)	40.00% (54%)	0.1946 (0.70)
TF-IDF-L2	49.20% (82%)	55.04% (89%)	0.5196 (0.73)

Table 2. Detailed recall and precision values for the best three preprocessing settings

Number of training	TF-NOWEIGHTS-L2			LOG-IDF-L2			TF-IDF-L2		
documents	Recall	Precision	$F1$	Recall	Precision	$F1$	Recall	Precision	$F1$
50	81.48%	66.67%	0.73	55.56%	83.33%	0.67	81.48%	64.71%	0.72
100	61.54%	24.24%	0.35	46.15%	35.29%	0.40	53.85%	28.00%	0.37
150	46.15%	33.33%	0.39	46.15%	46.15%	0.46	46.15%	46.15%	0.46
200	56.25%	45.00%	0.50	37.50%	60.00%	0.46	56.25%	60.00%	0.58
250	64.29%	39.13%	0.49	42.86%	37.50%	0.40	57.14%	40.00%	0.47
300	62.50%	55.56%	0.59	37.50%	60.00%	0.46	56.25%	60.00%	0.58
350	18.18%	20.00%	0.19	9.09%	16.67%	0.12	9.09%	20.00%	0.12
400	50.00%	69.23%	0.58	55.56%	55.56%	0.56	66.67%	66.67%	0.67
450	57.14%	47.06%	0.52	57.14%	53.33%	0.55	57.14%	61.54%	0.59
500	75.00%	21.43%	0.33	25.00%	10.00%	0.14	50.00%	22.22%	0.31
550	0.00%	100.00%	0.00	0.00%	0.00%	-	0.00%	0.00%	-
600	42.86%	23.08%	0.30	42.86%	27.27%	0.33	42.86%	37.50%	0.40
650	80.00%	66.67%	0.73	100.00%	55.56%	0.71	80.00%	66.67%	0.73
700	53.85%	77.78%	0.64	46.15%	60.00%	0.52	46.15%	75.00%	0.57
750	55.56%	45.45%	0.50	66.67%	50.00%	0.57	44.44%	44.44%	0.44
800	15.38%	33.33%	0.21	23.08%	42.86%	0.30	7.69%	25.00%	0.12
850	8.33%	50.00%	0.14	41.67%	83.33%	0.56	41.67%	83.33%	0.56
900	25.00%	40.00%	0.31	25.00%	40.00%	0.31	25.00%	40.00%	0.31
950	35.71%	71.43%	0.48	35.71%	71.43%	0.48	35.71%	83.33%	0.50
1000	53.33%	100.00%	0.70	46.67%	77.78%	0.58	53.33%	88.89%	0.67
1050	23.81%	62.50%	0.34	38.10%	100.00%	0.55	38.10%	80.00%	0.52
1100	52.63%	83.33%	0.65	57.89%	73.33%	0.65	52.63%	76.92%	0.63
1150	55.56%	55.56%	0.56	55.56%	45.45%	0.50	44.44%	36.36%	0.40
1200	72.73%	66.67%	0.70	72.73%	53.33%	0.62	81.82%	64.29%	0.72

References

BENTLEY, J. and SEDGEWICK, R. (1997): Fast Algorithms for Sorting and Searching Strings.
http://www.cs.princeton.edu/~rs/strings/paper.pdf

BOMHARDT, C., GAUL, W. and SCHMIDT-THIEME, L. (2004): Web Robot Detection - Preprocessing Web Logfiles for Robot Detection, to appear.

BOSER, B., GUYON, I. and VAPNIK, V. (1992): A Training Algorithm for Optimal Margin Classifiers.
http://citeseer.nj.nec.com/boser92training.html

CHEN, L. and SYCARA, K. (1997): WebMate: A Personal Agent for Browsing and Searching. *http://citeseer.nj.nec.com/chen98webmate.html*

COOLEY, R. (1999): Classification of News Stories Using Support Vector Machines.
http://citeseer.nj.nec.com/cooley99classification.html

DUMAIS, S., PLAT, J., HECKERMAN, D. and SAHAMI, M. (1998): Inductive Learning Algorithms and Representation for Text Categorization.
http://robotics.stanford.edu/users/sahami/papers-dir/cikm98.pdf

GAUL, W., GEYER-SCHULZ, A., HAHSLER, M. and SCHMIDT-THIEME, L. (2002): eMarketting mittels Recommendersystemen. *MARKETING ZFP, 24. Jg. Spezialausgabe "E-Marketing" 2002, 47–55.*

HEISE: Heise News Ticker, http://www.heise.de/ct

JOACHIMS, T. (1999): Making Large-Scale SVM Learning Practical. Advances in Kernel Methods. In: B. Schölkopf, C. Burges and A. Smola (Ed.): *Support Vector Learning*, MIT-Press.

JOACHIMS, T. (2002): Learning to Classify Text Using Support Vector Machines. *Kluwer Academic Publishers*

JOACHIMS, T., FREITAG, D. and MITCHELL, T. (1996): WebWatcher: A Tour Guide for the World Wide Web.
http://citeseer.nj.nec.com/joachims96webwatcher.html

LANG, K. (1995): NewsWeeder: Learning to Filter Netnews.
http://citeseer.nj.nec.com/lang95newsweeder.html

LIEBERMAN, H. (1995): Letizia: An Agent That Assists Web Browsing.
http://lieber.www.media.mit.edu/people/lieber/Lieberary/Letizia/Letizia-AAAI/Letizia.ps

MIDDLETON, S. (2001): Interface Agents: A Review of the Field. *Technical Report Number: ECSTR-IAM01-001.*
http://www.ecs.soton.ac.uk/~sem99r/agent-survey.pdf

MLADENIĆ, D. (1999): *Machine Learning Used by Personal WebWatcher.*
http://www-2.cs.cmu.edu/afs/cs/project/theo-4/text-learning/www/pww/papers/PWW/pwwACAI99.ps

MLADENIĆ, D. (2001): Using Text Learning to Help Web Browsing.
http://www-ai.ijs.si/DunjaMladenic/papers/PWW/hci01Final.ps.gz

MOBASHER, B., COOLEY, R. and SRIVASTAVA, J. (1999): Automatic Personalization Based on Web Usage Mining.
http://citeseer.nj.nec.com/mobasher99automatic.html

PAAß, G., KINDERMANN, J. and LEOPOLD, E. (2004): Text Classification of News Articles with Support Vector Machines. In: S. Sirmakessis (Ed.): *Text Mining and its Applications.* Springer, Berlin, 53–64.

SEBASTIANI, F. (2002): Machine Learning in Automated Text Categorization. *ACM Computing Surveys, 34(1), 1–47.*