Using Text Learning to help Web browsing

Dunja Mladenić
J.Stefan Institute, Ljubljana, Slovenia
Carnegie Mellon University, Pittsburgh, PA, USA
Dunja.Mladenic@{ijs.si, cs.cmu.edu}

Abstract

Web browsing is gaining popularity with the growing number of Web users, especially for a casual usage of the Web, when the user does not have a precise query in mind. By observing the user's behavior when browsing, we build a model of promising hyperlinks and use it to highlight hyperlinks on the requested Web pages. In order to do that, we propose text learning methods for handling high dimensional problems (having severals tens of thousands of features) with highly unbalanced class distribution (more than 90% of examples having majority class value). Extensive experimental results were performed on a related problem of modelling Web document content category by using hyperlink to the document. The results show that when modelling by Naive Bayesian classifier, it is highly important how we select the features to be used in the model. Namely, the best performing feature selection in our experiments on Personal WebWatcher data is when the features are scored according to *Odds Ratio* and only a small number of the best features is used for learning.

1 Introduction

Large amount of information available of the Web is attracting many users that are trying to find interesting Web pages. When having an idea about the goal of their search, users typically go to some of the existing search engines and issue a query hoping the target information will be found on some of the top ranked pages. A number of researchers are trying to improve the results of a search engine by addressing the problems such as better query handling by query expansion or stronger query language (eg., request for excluding terms in [AltaVista]), improving search engine's algorithms (eg., hyperlink structure taken into account by [Google]), post processing of the search results by reordering or clustering them (eg., as performed in a research search engine [Manjara] or [Vivisimo]). On the other hand, when the user is not certain what to look for, s/he rather browses the Web by mainly following hyperlinks on the requested Web pages. By processing the requested documents and analyzing their hyperlinks, we can highlight promising hyperlinks and help the user in browsing the Web.

In order to be able to judge the hyperlink we need a model of promising hyperlinks. By promising, we mean here hyperlinks the user is likely to click on, so we want to point them to the user in advance. The model we are using in our approach is based on the content analysis of requested documents and hyperlinks. The methods we use to build this model are text learning methods. The problem of building a model of promising hyperlinks can be seen as a problem of classifying hyperlinks as promising or non-promising. This problem turned out to be non-trivial, by its high dimensionality (ie., a large number of different words that occur in the documents) and a small proportion of positive examples (the proportion of promising hyperlinks among all the hyperlinks is low). To handle this problem, we investigated different ways to reduce a large number of words (perform feature subset selection).

2 Domain description

Machine learning problem is here defined as predicting clicked hyperlinks from the set of Web documents visited by the user. This is performed on-line while user is sitting behind some Web browser and waiting for the requested document. Our prototype system named Personal WebWatcher [5] uses text-learning on this problem, learning separate model for each user and highlighting hyperlinks on the requested Web documents. All hyperlinks from the visited documents are used as machine learning examples. Each is assigned one of the two class values: positive (user clicked on the hyperlink) or negative. We use machine learning to model the function $User_{HL} : HyperLink \rightarrow \{pos, neg\}$. Our hope is that this function is also some approximation of interesting hyperlinks (user clicked on hyperlinks that she/he is interested in and skipped all other hyperlinks, that is of course not always true!). We represented each hyperlink as a small document containing underlined words, words in a window around them and words in all the headings above the hyperlink. Our documents are represented as word vectors (using so called bag-of-words representation commonly used in information retrieval) and learning was performed using Naive (simple) Bayesian classifier as commonly used on text data (for overview of text-learning approaches see [6]). For each word position in the document, a feature is defined having a word as its value [4].

Experiments are performed using personal browsing assistant `Personal WebWatcher` that observes users of the Web and suggests pages they might be interested in. It learns user interests from the pages requested by the user. The learned model is used to suggest hyperlinks on new HTML-pages requested by and presented to the user via Web browser that enables connection to “proxy”.

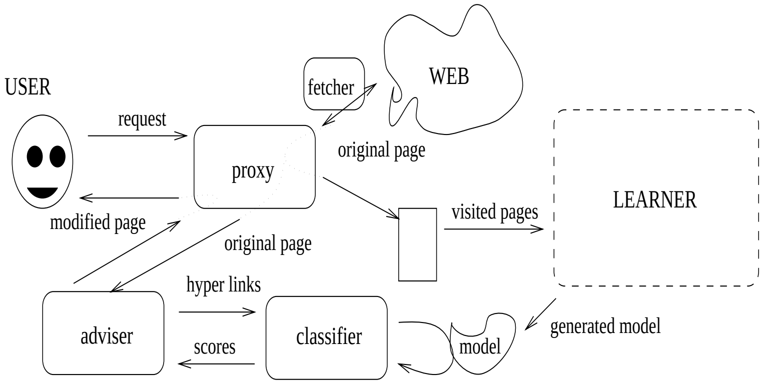

Figure 1: Structure of browsing assistant Personal WebWatcher.

Helping the user browsing the Web is performed here by highlighting interesting hyperlinks on the requested Web documents. We assume that the interesting hyperlinks are the hyperlinks that are highly probable to be clicked by the user. Our problem is defined as predicting clicked hyperlinks from the set of Web documents visited by the user. All hyperlinks that are present on the visited documents are used for constructing machine learning examples.

The data was collected for users participating in the HomeNet project [1]. Results for two users are described in Section 4 with the data characteristics are given in Table 1. For each user approximately 4000 different words occurred in documents resulting here with 4000 features.

3 Feature selection on text data

The usual way of learning on text defines a feature for each word that occurred in the training documents. This can easily result in several tens of thousands of features. Methods for feature subset selection that are used on text are very simple compared to the methods developed in machine

Domain (user id.)	Positive class probability	Number of examples	data entropy
usr150101	0.104	2 528	0.480
usr150211	0.044	2 221	0.259
usr150202	0.053	4 798	0.301
usr150502	0.100	2 498	0.468

Table 1: Domain description for data collected from four HomeNet users. It can be seen that we are dealing with unbalanced class distribution, since 10 % or less examples are positive, all other are negative.

learning. Basically, some evaluation function that is applied to a single feature is used. All the features are independently evaluated, a score is assigned to each of them and the features are sorted according to the assigned score. Then, a predefined number of the best features is taken to form the solution feature subset.

Scoring of individual features can be performed using some of the known measures, for instance some measure used in machine learning, such as $Information\ gain$ used in decision tree induction [9]. $InfGain(W) = P(W) \sum_i P(C_i|W) \log \frac{P(C_i|W)}{P(C_i)} + P(\overline{W}) \sum_i P(C_i|\overline{W}) \log \frac{P(C_i|\overline{W})}{P(C_i)}$ In our comparison several feature scoring measures were comapre. Information gain was included as the well known measure successfuly used in some text-learning experiments. Very simple frequency measure proposed in [12] were reported to work well on text data $Freq(W) = TF(W)$. *Odds ratio* is commonly used in information retrieval, where the problem is to rank out documents according to their relevance for the positive class value using occurrence of different words as features [10]. $OddsRatio(W) = \log \frac{P(W|pos)(1-P(W|neg))}{(1-P(W|pos))P(W|neg)}$ Our experiments show that this measure is especially suitable to be used in a combination with the Naive Bayesian classifier for our kind of problems. We propose some variants of *Odds ratio*, to test if the results are sensitive to some modifications in the formula, eg., $FreqLogP(W) = \log \frac{P(W|pos)}{P(W|neg)}$ $ExpP(W) = e^{P(W|pos)-P(W|neg)}$. As a baseline method we used random scoring method defined to score each word by a random number.

Scoring measure	Accuracy					Information score				
best features	10	100	200	500	1000	10	100	200	500	1000
user150101										
ExpP	**94.35**	**94.49**	**94.52**	**94.49**	**94.19**	**0.037**	**0.042**	**0.043**	**0.038**	**0.021**
FreqLogP	**94.32**	**94.49**	**94.52**	**94.50**	**94.18**	**0.036**	**0.041**	**0.042**	**0.037**	**0.019**
OddsRatio	94.08	94.27	94.04	93.68	93.26	0.030	0.036	0.027	0.016	-0.002
InfGain	94.24	92.62	92.27	92.16	92.09	0.011	-0.064	-0.073	-0.074	-0.071
FreqOddsRatio	92.44	92.35	92.33	92.15	91.87	0.048	-0.045	-0.044	-0.047	-0.06
Freq	91.72	90.85	90.75	90.73	91.74	0.264	-0.242	-0.227	-0.210	-0.197
Random	93.39	93.37	93.37	93.23	92.73	0.005	-0.012	-0.020	-0.035	-0.059
user150211										
ExpP	**96.61**	**96.63**	**96.60**	**96.42**	**95.97**	**0**	**0.001**	**-0.003**	**-0.014**	**-0.039**
FreqLogP	**96.60**	**96.62**	**96.56**	**96.41**	**95.97**	**-0.001**	**0**	**-0.005**	**-0.017**	**-0.042**
OddsRatio	**96.61**	**96.64**	**96.51**	**96.22**	**95.76**	**0.002**	**0.005**	**-0.008**	**-0.027**	**-0.055**
InfGain	94.31	94.01	93.66	93.29	93.09	-0.226	-0.294	-0.320	-0.334	-0.337
FreqOddsRatio	95.76	95.52	95.43	95.25	94.86	0.130	-0.136	-0.138	-0.145	-0.16
Freq	94.49	92.60	92.05	91.74	91.62	-0.374	-0.40	-0.420	-0.428	-0.427
Random	96.57	996.46	96.38	96.18	95.77	**0.003**	-0.015	-0.035	-0.066	-0.103

Table 2: Comparison of different feature scoring measures giving the average classification accuracy and the average information score of 10 hold-out testing repetitions on the two data sets. The results are given for different number of the best features selected according to each of the scoring measures (10, 100, 200, 500, 1000). The results show here represent a subset of the classification results plotted in Figure 2.

4 Experimental results

We measure classification accuracy, defined as a percent of correctly classified examples and calculated over all classes. We used hold-out testing with 10 repetitions using 30% randomly selected examples as testing examples and reported average value and standard error. Feature selection and learning was performed on training examples only. For each data set we observed the influence of the number of the best features selected for learning to the system performance. Since we have unbalanced class distribution (see Table 1), Classification accuracy can give misleading results. For such domains more appropriate measure is Information score [3]. In the experimental results presented in Figure 2 Classification accuracy and Information score are used to estimate model quality. For both domains the highest Classification accuracy and the highest Information score are achieved by the measures based on Odds ratio: $ExpP, FreqLogP, OddsRatio$ (see Table 2) and Figure 2). For these measures the best vector size is approximately between 60 and 200 best features. This means that the selected feature subset includes just 2% - 5% of all features. The similar reduction (up to 90%) in the number of features used in text-learning was observed in [12]. The other three measures ($InfGain, Freq, FreqOddsRatio$) for most vector sizes perform about the same or even worse than $Random$. Closer look to the words sorted according to Information gain showed that the most best words are characteristic for negative class value (their probability estimated for positive documents is 0). This results were also confirmed by experiments on document categorization into hierarchical structure of Web documents, Yahoo! [7]. This means that in classification, a new positive hyperlink is represented with a word vector almost full of zeros, since it contains very few of the selected best words. In our experiments we didn't remove any common or frequent words. That resulted with html-tags and other common words beeing the most frequent, contributing to the poor performance of the Frequency measure $Freq$. Our explanation for the poor results achieved by the combination of Frequency and Odds ratio $FreqOddsRatio$ is that the value of Frequency is standing out in this combination. Odds ratio has most values between 1 and 20 while Frequency has values between 1 and 1000, resulting in their. combination having values between 1 and 2000. In case of the combination of Frequency with logarithm of probability ratio $FreqLogP$, the logarithmic part is standing out and the measure achieves better results.

References

[1] Kraut, R., Scherlis, W., Mukhopadhyay, T., Manning, J., Kiesler, S., The HomeNet Field Trial of Residential Internet Services, *Communications of the ACM* Vol. 39, No. 12, pp.55—63, December 1996.

[2] Joachims, T., A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization, *Proc. of the 14th International Conference on Machine Learning ICML97*, pp. 143—151, 1997.

[3] Kononenko, I. and Bratko, I., Information-Based Evaluation Criterion for Classifier's Performance, *Machine Learning 6*, Kluwer Academic Publishers, 1991.

[4] Mitchell, T.M., Machine Learning, The McGraw-Hill Companies, Inc., 1997.

[5] Mladenić, D., Personal WebWatcher: Implementation and Design, *Technical Report IJS-DP-7472*, October, 1996. http://www-ai.ijs.si/DunjaMladenic/papers/PWW/

[6] Mladenić, D. (1999). Text-learning and related intelligent agents. IEEE EXPERT, Special Issue on Applications of Intelligent Information Retrieval, May-June 1999.

[7] Mladenić, D. & Grobelnik, M. (1999). Feature selection for unbalanced class distribution and Naive Bayes, *Proceedings of the 16th International Conference on Machine Learning ICML-99*, Morgan Kaufmann Publishers, San Francisco, CA. pp. 258-267.

[8] Pazzani, M., Billsus, D., Learning and Revising User Profiles: The Identification of Interesting Web Sites, *Machine Learning 27*, Kluwer Academic Publishers, pp. 313—331, 1997.

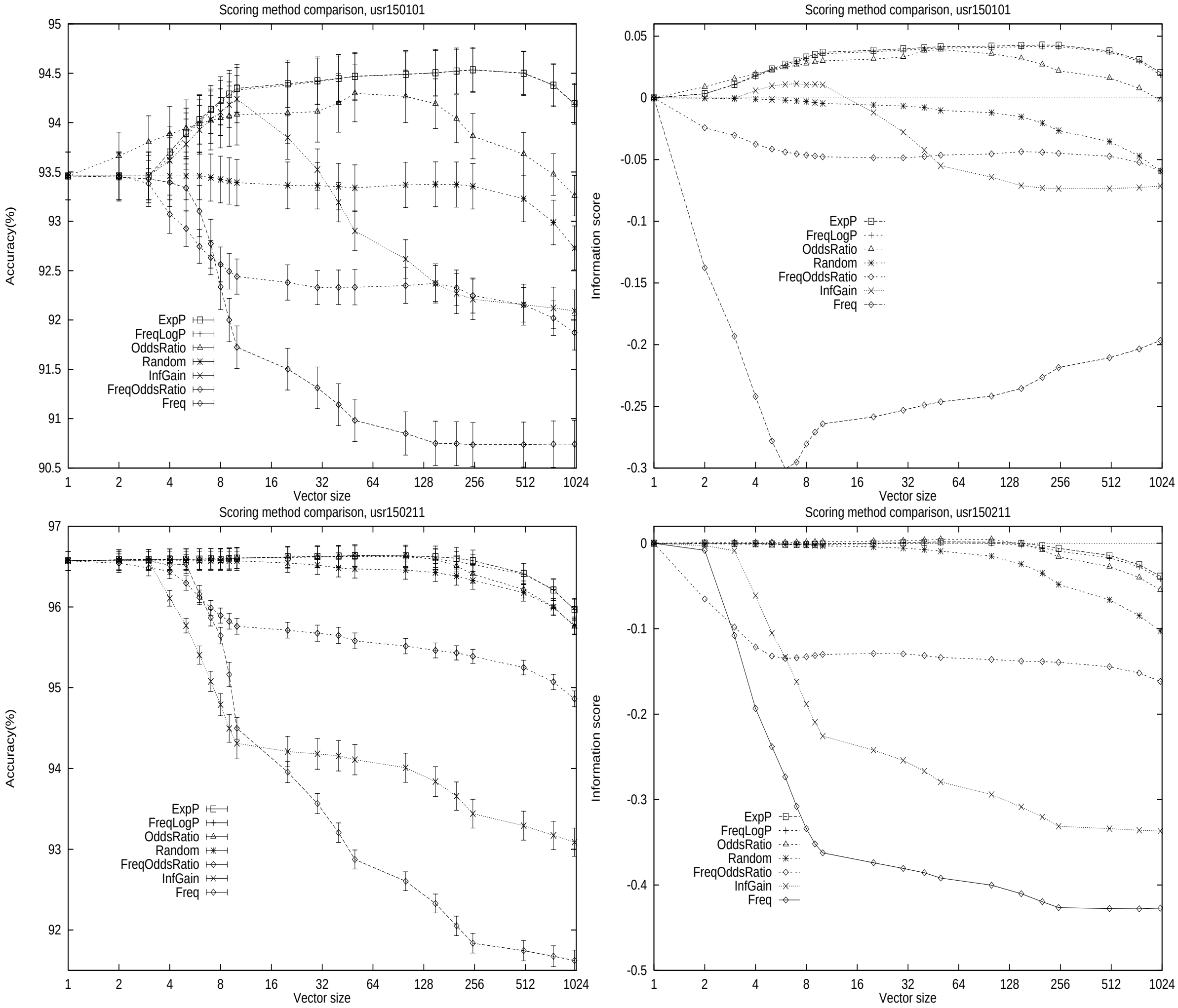

Figure 2: Influence of vector size to Classification accuracy and Information score on data for Home-Net usr150101 (upper) and usr150211 (lower). Notice that curve names are sorted according to the values at the end.

[9] Quinlan, J.R. (1993). Constructing Decision Tree. In *C4.5: Programs for Machine Learning*. Morgan Kaufman Publishers. pp. 17-26.

[10] van Rijsbergen, C.J,. Harper, D.J., Porter, M.F., The selection of good search terms, *Information Processing & Management*, 17, pp.77—91, 1981.

[11] Shaw Jr, W.M., Term-relevance computations and perfect retrieval performance, *Information Processing & Management*, 31(4), pp.491—498, 1995.

[12] Yang, Y., Pedersen, J.O., A Comparative Study on Feature Selection in Text Categorization, *Proc. of the 14th International Conference on Machine Learning ICML97*, pp. 412—420, 1997.