Feature selection for classification based on text hierarchy

Dunja Mladenić and Marko Grobelnik
Department of Intelligent Systems, J.Stefan Institute,
Jamova 39, 1111 Ljubljana, Slovenia
Phone: (+386)(61) 1773 272, Fax: (+386)(61) 1258-158
E-mail: Dunja.Mladenic@ijs.si, Marko.Grobelnik@ijs.si

Abstract

This paper describes automatic document categorization based on large text hierarchy. We handle the large number of features and training examples by taking into account hierarchical structure of examples and using feature selection for large text data. We experimentally evaluate feature subset selection on real-world text data collected from the existing Web hierarchy named Yahoo. In our learning experiments naive Bayesian classifier was used on text data using feature-vector document representation that includes word sequences (n-grams) instead of just single words (unigrams). Experimental evaluation on real-world data collected form the Web shows that our approach gives promising results and can potentially be used for document categorization on the Web. Additionally the best result on our data is achieved for relatively small feature subset, while for larger subset the performance substantially drops. The best performance among six tested feature scoring measure was achieved by the feature scoring measure called *Odds ratio* that is known from information retrieval.

1 Introduction

Growing interest in the usage of machine learning on text data is further stimulated with the intensive growth of World Wide Web that can be seen as a widely accessible source of text data. The problem of automatic document categorization is well known in information retrieval and usually tested on publicly available databases (eg. Reuters, MEDLINE). Here we use machine learning for document categorization using one of the existing Web hierarchies named Yahoo [3]. In this way, the current situation on the Web captured in the Web hierarchy is used for automatic document categorization. The proposed approach is not limited to Web hierarchy and can be applied on other hierarchies like for instance, thesaurus. The idea is to divide a categorization problem into subproblems and learn a separate classifier for each category. The hierarchical structure of data is taken into account when forming training examples for the subproblems. The result of document categorization is a list of categories each assigned a probability as a result of consulting the corresponding subproblem classifier.

We use naive Bayesian classifier on text data using feature-vector document representation that includes word sequences instead of just single words. It is well known in machine learning that features used to describe examples are not necessarily all relevant and beneficial for inductive learning. Additionally, a high number of features may slow down the induction process while giving similar results as obtained with a much smaller feature subset. Our experiments show the importance of feature subset selection in our domain when learning is performed using naive Bayesian classifier. The best performance among six tested measures was achieved for relatively small feature subset by Odds ratio.

2 Feature subset selection in text-learning

Selection of a subset of features to be used in inductive learning has already been addressed in machine learning. According to John et al. [6] there are two main approaches used in machine learning to feature subset selection: filtering approach and wrapper approach. Most of the feature subset selection approaches used in machine learning are not designed for the situations with a large number of features.

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. Most methods for feature subset selection that are used on text are very simple compared to the methods developed in machine learning. Basically, some evaluation function that can be used on a single feature is used. All the features are independently evaluated, a score is assigned to each of them and features are sorted according to the assigned score. Then, a predefined number of the best features is taken to form the solution feature subset. The number of features to select is an experimental issue that we address in our experiments.

Scoring of individual features can be performed using some of the methods used in machine learning for feature selection during the learning process, for example, *Information gain* used in decision tree induction [12]. *Expected cross entropy* used in text-classification experiments [7] is similar to Information gain. The difference is that instead of calculating average over all possible feature values, only the value denoting that word occurred in a document is considered. Our experiments show that this means an important difference in the resulting performance. *Mutual information* used in text-classification experiments [14] is similar to Cross entropy used on text data with the latter additionally taking into account word probability. We propose a new feature scoring measure called *Weight of evidence for text* that is based on the average absolute weight of evidence used in machine learning [8]. *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 [13]. Additionally we used a very simple measure based on word frequency that is shown to work well in the text-classification domains [14]. As a baseline method we used random scoring method defined to score each word by a random number.

- $InfGain(F) = P(W)\sum_i P(C_i|W)\log\frac{P(C_i|W)}{P(C_i)} + P(\overline{W})\sum_i P(C_i|\overline{W})\log\frac{P(C_i|\overline{W})}{P(C_i)}$
- $CrossEntropyTxt(F) = P(W)\sum_i P(C_i|W)\log\frac{P(C_i|W)}{P(C_i)}$
- $MutualInfoTxt(F) = \sum_i P(C_i)\log\frac{P(W|C_i)}{P(W)}$
- $WeightOfEvidTxt(F) = \sum_i P(C_i) \times P(W) \times |\log\frac{P(C_i|W)(1-P(C_i))}{P(C_i)(1-P(C_i|W))}|$
- $OddsRatio(F) = \log\frac{P(W|pos)(1-P(W|neg))}{(1-P(W|pos))P(W|neg)}$
- $Freq(F) = TF(W)$

Where F is the feature that represents word W, $P(W)$ is the probability of that word W occurred, $\overline{W}$ means that word doesn't occur, $P(C_i)$ in the probability of the i-th class value, $P(C_i|W)$ is the conditional probability of the i-th class value given that word W occurred, $P(W|C_i)$ is the conditional probability of word occurrence given the i-th class value, $P(W|pos)$ is the conditional probability of word W occurring given the class value 'positive', $P(W|neg)$ is the conditional probability of word W occurring given the class value 'negative', $TF(W)$ is term frequency (the number of occurrence of word W).

3 Learning from large text hierarchy

Hierarchically classifying documents was recently addressed in [7], where the Reuters dataset was used and three hierarchical structures were manually constructed each having a 3-level hierarchy that is based on up to 1,000 documents and the biggest having 12 nodes. The general idea of using hierarchical instead of 'flattened' approach is the same as used here. Its realization using machine learning techniques as well as the actual nature and the size of the used hierarchy is quite different. We use the existing Web hierarchy named Yahoo as example domain for the generation of classifier from large text hierarchy. The categories in the Yahoo hierarchy are human constructed and designed for human Web browsing making the hierarchy biased toward human knowledge areas that are represented in Web documents. For example, one of the most general categories named 'Bussines and Economy' is over-represented including more than a third of all documents, while the other category that appears at the same level of hierarchy named 'Social Science' includes less than 1% of all documents. The Yahoo hierarchy itself (without a huge category 'Regional') is currently built on approximately 900,000 Web documents located all around the Internet. We refer here to these documents as *actual Web documents*. Hyperlinks to that documents are organized in about 50,000 *Yahoo Web documents*. Each Yahoo document represents one of the included categories. These documents are connected with hyperlinks forming a hierarchical structure with more general categories closer to the root of the hierarchy.

In order to apply machine learning on text data we represent documents as feature-vectors using the bag-of-words representation as commonly used in learning on text data. We also reduce the number of features by removing words contained in the "stop-list" and removing infrequent words. Our document representation includes not only single words (unigrams) but also up to 5 words (1-grams, 2-grams, ... 5-grams) occurring in a document as a sequence (eg. 'machine learning', 'world wide web'). Since we have previously removed words contained in "stop-list", some features that represent word sequences can actually capture longer sequences like for instance, 'Word for Windows' that is represented as a 2-gram 'Word Windows' or 'winners will be posted at the end of each two-week period' that is represented as a 5-gram 'winners posted end two-week period'. In this way we can capture some characteristic word combinations but also increase the number of features (eg., in the whole Yahoo hierarchy from 69,280 to 255,602 features). The process of feature generation is performed in n passes over documents, where i-grams are generated in the i-th pass. At the end of each pass over documents all infrequent features are deleted (here we check for frequency ≤ 3). Each new pass generates features of length i only from the candidate features (of length *i-1*) generated in the previous pass. This process is similar to the large k-itemset generation used in association rules algorithm [1].

Learning algorithm used in our experiments is based on using naive Bayesian classifier on text data as described in [5] and [9]. In their approach documents are represented as word-vectors where a feature is defined for each word position in the document having a word at that position as a feature value. In machine learning setting this document representation can be seen as a fixed size vector having a feature for each word from the domain containing word frequency in a document. The assumption about feature independence used in naive Bayesian classifier is clearly incorrect. According to [2] this does not necessary mean that classifier will have poor performance because of that.

We use the Yahoo hierarchy to learn document categorization. For a new document, the classifier returns probability distribution over categories included in the hierarchy. One possibility is to use 'flattened' approach and generate one huge classifier with many class values, each value corresponding to one category requiring a large number of features to cover all the variety of the large number of documents included in different categories. As pointed out in [7], a better idea is to somehow split the whole problem into subproblems. We use the hierarchical structure to decompose the problem into a set of subproblems corresponding to individual categories. For each of the subproblems, a classifier is constructed that predicts the probability that a document is a member of the corresponding category. A set of positive and negative examples for each subproblem is constructed from the given hierarchical structure. We define a set of negative examples as examples from the whole domain hierarchy. The set of positive examples is constructed from examples in the corresponding category node and all the examples from its subtrees, if any. When propagated to the higher levels, examples are assigned weight proportional to the size of node they appear in. We are using items (hyperlinks and their description) on Yahoo documents that contain hyperlinks to actual Web documents as training examples instead of using actual Web documents. Our assumption here is that an item contains words representative for the document it points to. Words in each item represent description of the document it points to carefully chosen by the document author. In this way we reduce the time and space needed to collect and store training data. The actual Web documents are used as testing examples selected randomly from the actual Web documents (not Yahoo Web documents that are used in learning) accessible from the hierarchy domain. The final result of learning is a set of specialized classifiers.

4 Experimental results

Our experiments are performed using our recently developed machine learning system **Learning Machine** [4] that supports usage of different machine learning techniques on large data sets with especially designed modules for learning on text and collecting data from the Web. In our experiments we observe the influence of different number of features (vector size in feature-vector document representation) and probability thresholds to classification results on independent set of (300 for *'Computers and Internet'* and 200 for the two smaller domains) testing examples selected randomly from the actual Web documents. We get our categorization results from the set of independent classifiers, each potentially having different number of features. Thus we express the vector size as a factor of the number of features in positive examples, since the set of negative examples is the same for all classifiers. In this way, a classifier for a larger category is using more features than a classifier for some smaller category, while

both classifying the same testing example. For each testing example we observe a list of categories each assigned a probability as a result of consulting the corresponding subproblem classifier. Sorting categories according to that probability gives ranking that we use to get the rank of the correct category. In case of several categories having the same probability, the ranking of the middle one is taken. For each testing example we report rank and probability assigned to the correct category. To get summary results over testing examples, we give mediana with lower and upper quartile, since some of the testing examples are rather non-typical of their category, containing eg., welcome page or only one sentence asking for language preference or error message or page giving redirection.

In order to enable operational usage of the system on larger domains we include the following pruning mechanisms. When classifying testing example only classifiers that share at least 3 highly scored features with the example are considered. More specifically, a classifier is required to assign a non-zero conditional probability given the positive class value $P(W|pos) > 0$, to at least 3 highly scored features that are contained in the testing example. Namely, this conditional probability is used in naive Bayesian classifier when calculating probability that the testing example should be assigned a category corresponding to the given classifier. In case that non of the features contained in the testing examples is assigned non-zero probability by a classifier, a prior probability is returned (in our case for most classifiers the prior probability of positive class value is $<< 0.1$). We tested different pruning levels (1,2,... 15) on several domains [11] and here use 3 as one of the best performing.

Results of naive Bayesian classifier for six feature selection measures are given on domains representing three of the top fourteen Yahoo categories. *'References'* having 129 categories and 923 (697 1-grams+196 2-grams+27 3-grams+3 4-grams+0 5-grams) features, *'Education'* having 349 categories and 3,215 (1,928+1,067+186+28+6) features and *'Computers and Internet'* having 2,652 categories and 7,631 (5049+2276+261+38+7) features. We test our approach on three domains that are parts of the Yahoo hierarchy, since they are hierarchies themselves with the same problem characteristics as the whole Yahoo hierarchy but smaller.

Domain name	Scoring measure	Correct category rank	Correct category probability	Vector size
References	**Odds ratio**	**2**	**> 0.99**	**1.5-3**
	Cross entropy	2	>0.99	1
	Term frequency	3	>0.99	0.5-1
	Weight of evidence	5	<0.01	0.5-1
	Mutual information	30	0.03	1.0
	Random	25-35	<0.01	3-15
	Information gain	44-46	<0.01	2-30
Education	**Odds ratio**	**3**	**> 0.99**	**2-5**
	Cross entropy	4	>0.99	1
	Term frequency	5	>0.99	1
	Weight of evidence	7	<0.01	0.5
	Mutual information	57-66	>0.012	0.5-1.5
	Random	90-96	<0.01	5-10
	Information gain	133-141	<0.01	0.5-30
Computers and Internet	**Odds ratio**	**3-4**	**> 0.98**	**6-8**
	Term frequency	4	>0.99	1
	Cross entropy	6	>0.99	1
	Weight of evidence	5	0.96	0.5
	Mutual information	540-803	< 0.01	0.5-15
	Random	570-720	<0.01	10-20
	Information gain	950-990	< 0.01	1.5-30

Table 1: Results of experiments on three domains formed from Yahoo text hierarchy. Feature scoring measures are sorted according to their performance in each domain.

Figures 1, 2 give mediana of rank and probability assigned to the correct category on three domains we used here. In all domains the best performance is achieved when only a small number of features is used and features are selected using Odds ratio as scoring measure (see also Table 1). This is consistent with the results reported in text-learning on the problem of predicting clicked hyperlinks from the set of Web documents visited by the user [10]. The best performance in rank and probability is achieved

here by Odds ratio on feature subset having relatively small number of features *'References'*: 1.5-3, *'Education'*: 2-5, *'Computers and Internet'*: 6-8 times the number of features in positive class. More specifically, on *'References'* is the mediana of the correct category rank 2 and the mediana of the correct category probability over 0.99, ie. the half of the testing examples are assigned rank 1 or 2 and probability > 0.99 for the correct category. On *'Education'* the mediana of the correct category rank is 3 and the mediana of the correct category probability over 0.99, ie. the half of the testing examples are assigned rank 1, 2 or 3 for the correct category and probability > 0.99. On *'Computers and Internet'* the mediana of the correct category rank varies between 3 and 4 while the mediana of the correct category probability is over 0.99, ie. the half of the testing examples are assigned rank 1, 2, 3 or 4 for the correct category and probability > 0.99. On *'References'* for the most vector sizes the lower quartile for rank and probability is 1, meaning that one fourth of the testing examples are assigned the highest rank to the correct category and the probability 1. On *'Education'* and *'Computers and Internet'* the lower quartile for rank is 1 or 2 and for probability 1.

All other tested measures perform significantly worst, some achieving comparable rank and/or probability but not consistently over domains and showing higher sensibility in performance to changing vector size (eg. Cross entropy). The second best result on 'References', 'Education' and 'Computers and Internet' is achieved for vector size 1 (the number of selected features equals the number of features in positive class examples) by Cross entropy (rank 2, 4 and 6 respectively and probability over 0.99) and a simple Frequency measure (rank 3, 5 and 4 respectively and probability over 0.99). The new measure Weight of evidence achieved on all three domains good rank (5, 7, and 5) for vector size 0.5 but low probability on 'References' and 'Education' (< 0.01). Mutual information performed slightly better than Random, while Information gain performed worst than Random. The poor performance of Information gain is caused by assigning the highest score to the features that are either characteristic for positive or negative class with the latter making majority. In this way, the most of the features highly scored by Information gain are characteristic for negative class and thus assigned the lowest score by Odds ratio. Observation of mean value and standard error on all three domains confirms that the best performing Odds Ratio is significantly better and more stable in rank and probability than the other tested measures.

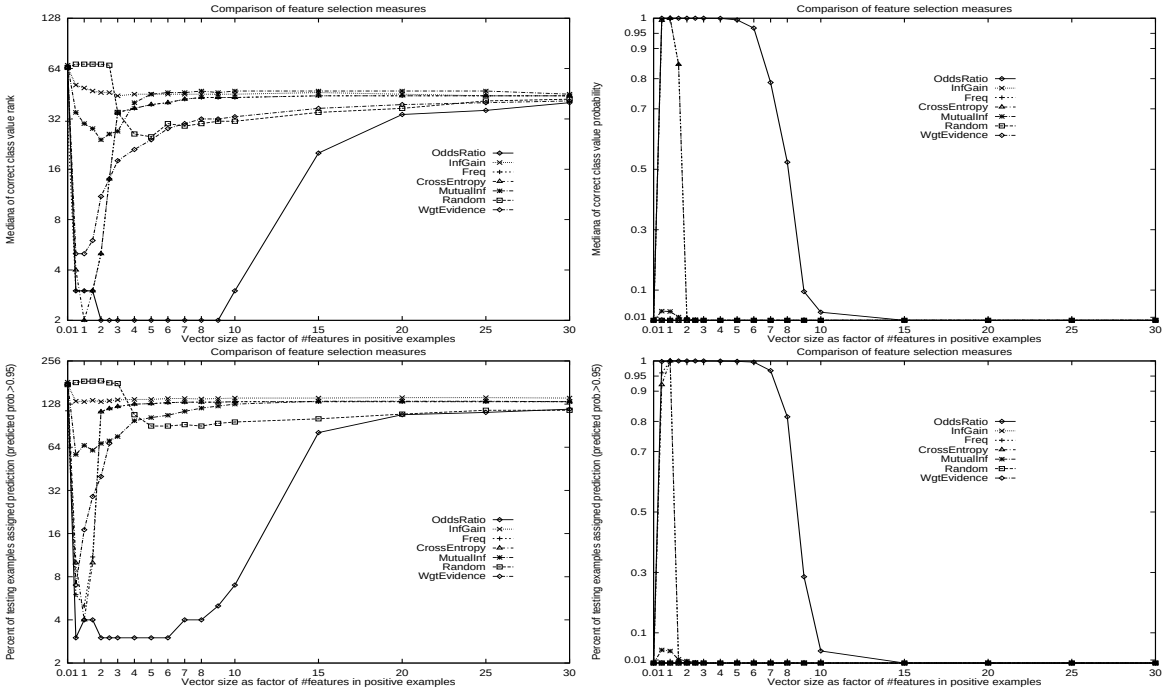

Figure 1: Comparison of feature selection measures on 'References' (upper) ans 'Education' (lower) (mediana of: correct category rank, probability of correct category).

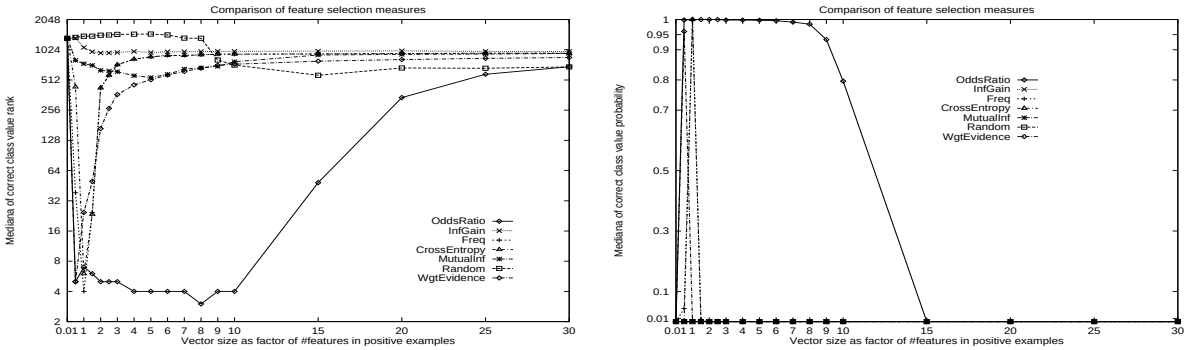

Figure 2: Comparison of feature selection measures on 'Computers and Internet' (mediana of: correct category rank, probability of correct category).

5 Acknowledgments.

This work was financially supported by the Slovenian Ministry for Science and Technology. Part of this work was performed during the authors stay at Carnegie Mellon University. Authors are grateful to Tom Mitchell and his machine learning group at Carnegie Mellon University for generous support, suggestions and fruitful discussions.

References

[1] Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., Verkamo, A.I., 1996. Fast Discovery of Association Rules, In Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R. (eds.), *Advances in Knowledge Discovery and Data Mining* AAAI Press/The MIT Press, pp. 307—328.

[2] Domingos, P., Pazzani, M., 1997. On the Optimality of the Simple Bayesian Classifier under Zero-One Loss, *Machine Learning 29*, Kluwer Academic Publishers, pp. 103—130.

[3] Filo, D., Yang, J., 1997. Yahoo! Inc., 1997. http://www.yahoo.com/docs/pr/

[4] Grobelnik, M., Mladenić, D., Learning Machine: design and implementation, *Technical Report IJS-DP-7824*, Department for Intelligent Systems, J.Stefan Institute, 1998.

[5] Joachims, T., 1997. 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.

[6] John, G.H., Kohavi, R., Pfleger, K., Irrelevant Features and the Subset Selection Problem, *Proc. of the 11th International Conference on Machine Learning ICML94*, pp. 121—129, 1994.

[7] Koller, D., Sahami, M., Hierarchically classifying documents using very few words, *Proc. of the 14th International Conference on Machine Learning ICML97*, pp. 170—178, 1997.

[8] Kononenko, I. On biases estimating multi-valued attributes. *Proc. of the 14th International Joint Conference on Artificial Intelligence IJCAI-95*, pp. 1034-1040, 1995.

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

[10] Mladenić, D., Feature subset selection in text-learning, *Proc. of the 10th European Conference on Machine Learning ECML98*, 1998.

[11] Mladenić, D., Grobelnik, M., 1998. Efficient text categorization, *Text Mining workshop on the 10th European Conference on Machine Learning ECML98*, (submitted).

[12] Quinlan, J.R., Constructing Decision Tree in *C4.5: Programs for Machine Learning*, pp.17—26, Morgan Kaufman Publishers, 1993.

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

[14] 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.