

Com S 573 Final Project

Weblog Explorer

Michael Fong
Dept. of Computer Science
Iowa State University

Heyong Wang
Dept. of Computer Science
Iowa State University

May 2, 2008

Contents

1	Introduction	2
1.1	Text Classification	2
1.2	Decision Tree	3
1.3	Support Vector Machine	4
2	Experimental Evaluation	5
2.1	Data Collection	6
2.2	Feature Selection	6
2.3	Classification	7
2.4	Experimental Result	8
2.5	Detecting Informative and Affective Articles	9
3	Conclusion	11
4	Acknowledgement	11

1 Introduction

Since the arise of Word Wide Web (WWW) in the early 1990's, copious of online textual information is handed everyday. Thus, text categorization has become one of the important techniques in knowledge discovery. Part of this revolution is led by the phenomenon of blogs (weblogs for short), which have become a prevalent source of information – personal articles, or professional journals, available on the Internet. Our project is to present a simple implementation to classify blog text into multi-categories.

As the internet has become people's daily necessity, more and more people share their information, ideas or opinion in their weblogs. Thus, weblog has become a prevalent source of information. In recent years, the number of weblogs has increased dramatically. According to a survey, by the end of 2005 there were 100 million blogs globally; the number of blogs in China has increased 1600 times in 4 years. Blog fever has spread out the world, in September 2005, 27% of the web users in U.S. read or wrote weblog regularly, the number increased to 39% by January 2006[11]. Thus, how to efficiently discovery knowledge from weblog space has become a very valuable study. Some research work is being conducted in this area, such as content based analysis, blog communities' evolution and weblog relevant features discovery. Weblog articles classification is the foundation part of this research.

1.1 Text Classification

The goal of text categorization is to classify a collection of documents, and assign each document to the applicable categories. However, building text classification "by hand" is time-consuming, thus it is more practical to use machine learning approach to classify from examples, which perform the category assignments automatically. Therefore, our report demonstrate one combination of among the best classifiers in an efficient way to solve this problem in the scenario of information organization and management.

Text Classification has been used as an assistance to the application in information management, and other areas. Spam detection can also apply the idea of text classification to filter the spam in email, news, or other web documents.

Many research in text classification focuses on the binary classification using Support Vector Machine (a.k.a. SVM for short) learning approach: Xiaochuan Ni and other researchers [3] have tried different machine learning

algorithms by classifying weblog articles into two categories. However, SVM is typically used to perform binary classification. In practice, binary classification is always not enough. Hence, many scientist has started working on finding the best combination of SVM with other classifiers to perform multi-class classification. One approach is to combine the decision tree with SVM[2], and our project tries to develop a simpler implementation of classification combining with Decision Tree and Support Vector Machine.

1.2 Decision Tree

A decision tree is a tree structure, whose internal nodes performs a test on one attribute and each branch from that node selects one value for that attribute. Finally, the leaf node predicts a category, based on the previously selected value for each visited attributes. Generally, each internal nodes may have more than two branching paths, depending on the number of values for each attribute.. Thus, the decision trees are not limited to binary classification at each decision-making node.

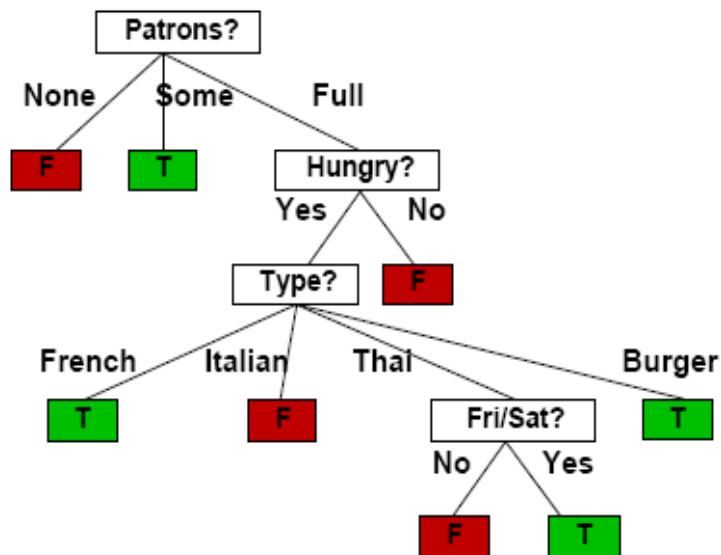


Figure 1: Example of Decision Tree from Com Sci 573 Lecture Notes[17]

1.3 Support Vector Machine

The Support Vector Machine, is a powerful learning algorithm, proposed by Vapnik[1], that typically aims to find a maximum margin separating hyperplane between two classes of data set. That is we are given training data to maximize the nearest distance between a point in one separated hyperplane and a point in the other separated hyperplane.

Many scientists working on related research has proved SVM to be one of

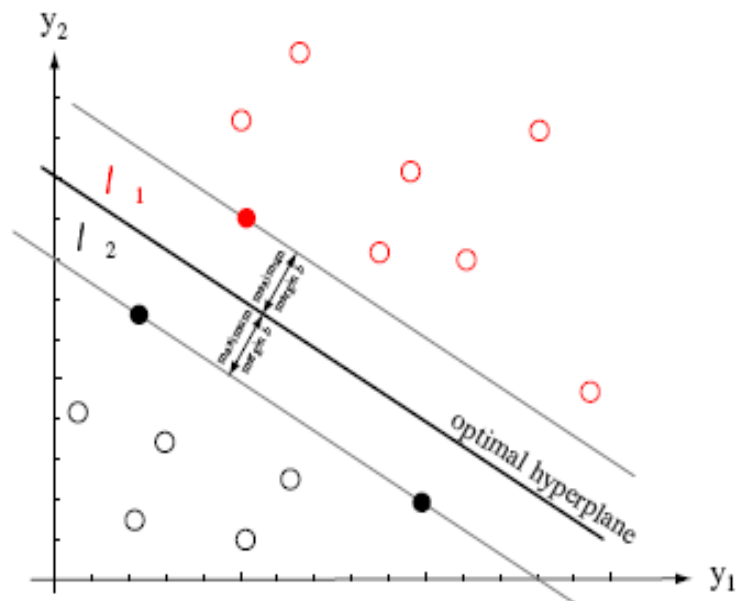


Figure 2: Picture Taken from Pattern Recognition Book, Chapter 5.11[8]

the best machine learning algorithms in binary classification[10]. Several remarkable properties of SVM listed below showed its substantial performance in text classification[5]:

- SVM is able to learn independent of the dimensionality of the feature space.
- SVM shows efficient performance, even if the size of the feature space is large.
- Document vectors are usually tended to be sparse.

Nevertheless, the original SVM is designed for binary classification and a direct solution to the multi-class problem using SVM is usually avoided, due to the various complexities. The typical approach is to use a combination different classifying methods (usually accompanied with SVM) to solve a multi-class classification problem.

2 Experimental Evaluation

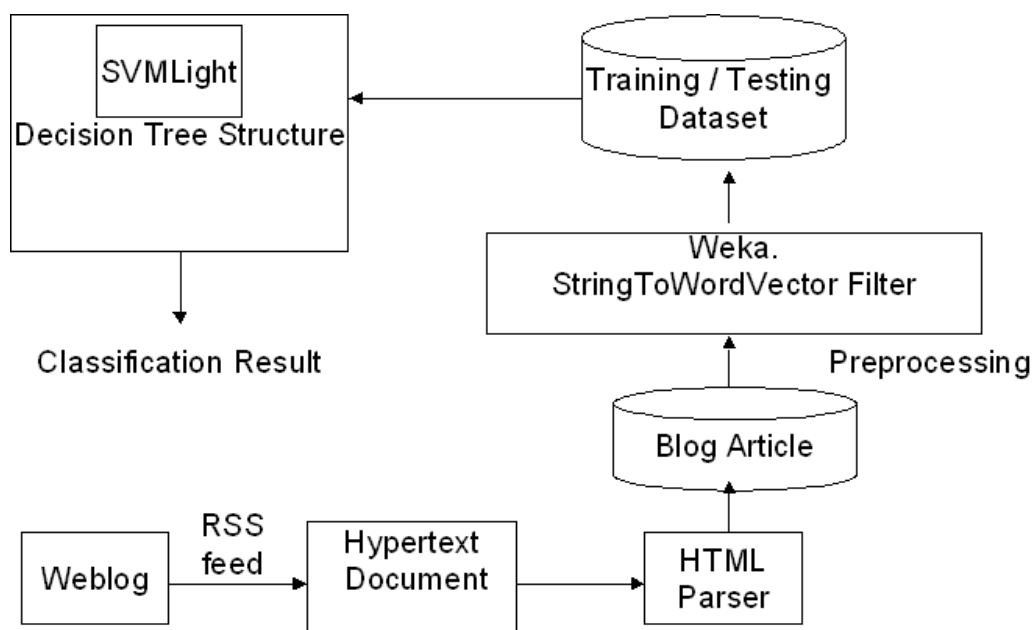


Figure 3: The Overview Scheme of the Implementation

We have implemented several program in Java for this project. In the front end, it first read the source (blog article) via Web feed, parse out the hypertext syntax, filter all relatively non-important keywords, and convert the data set into SVMLight[16][15] compatible format. Then, we use a decision-tree-like structure, implemented with SVMLight[16][15], to perform the classification on training sets and testing sets. Here is the list of libraries that we have used in this project,

- ROME (v 0.9)[12]
is a Java tools for parsing, generating and publishing RSS and Atom feeds. Formats include RSS 0.90, RSS 0.91 Netscape, RSS 0.91 User-

land, RSS 0.92, RSS 0.93, RSS 0.94, RSS 1.0, RSS 2.0, Atom 0.3, and Atom 1.0

- HTMLParser (v 1.6)[13]
is a fast Java package used to parse HTML documents. Primary features include data transformation, data extraction, filters, custom tags, and easy-to-use JavaBeans.
- WEKA (v.3.5.7)[14]
is a well-developed Java package that supports a collection of machine learning algorithms for data mining tasks. WEKA supports data pre-processing, classification, regression, clustering, association rules, and visualization.
- SVMLight (v.6.0.1)[16][15]
is an implementation of Support Vector Machine in C. The original program supports optimization, classification, regression, sparse vector representation, and ranking. In our project, we adopted a Java native interface, which support full functionalities of SVMLight. This Java interface API is programmed by Martin Theobald at Stanford University.

2.1 Data Collection

For this experiment, the blog corpus was obtained with our blog reader program, which works as follow. First, we obtained a corpus of roughly 1,000 blog articles for the pre-selected categories: Travel, Health and Sport, from various online blogging community such as BlogSpot, My Space, or Blogger. After that, we converted the document into plain text. The blog reader is implemented with ROME[12] to read the hypertext documents via RSS feed, and HTMLParser[13] to parser the HTML documents. In our experiment, we divided those 1000 documents into training sets and testing sets, around one-third (300) as testing set (roughly 100 articles for each category), and leave the rest of 700 documents as testing sets.

2.2 Feature Selection

The first step in text classification is to convert documents, represented by strings of characters, into a suitable format for the classification task. During the conversion, we determine the key phrases by viewing the word occupance in the feature space(vocabulary). We use the StringToWordVector filter in WEKA[14] to convert the original ariticle data into a new data set with

word frequency for each word. This filtering idea works as follow. Each distinct word corresponds to a feature, with the number of appearance in the data set (all documents)in the data set (all documents). The more frequent a word appears in the data set, the more important the word represents to the document. However, words with high frequency are commonly the daily words we use everyday (like 'a', 'the', etc). Those stop words (also from WEKA[14]) should be omitted, and our experiments shown stop words occupy one-fourth of the whole feature space. Last, we also filter numbers, punctuation, and etc.

2.3 Classification

In order to represent in multi-class fashion, we would like to explore both the nature of decision trees and Support vector machines. To achieve better classification for each class, we created a decision-tree-like structure, and made the SVM as decisions in the tree. For each level, we train m independent binary SVMs, where m is the number of class unclassified classes at the level. The best SVM is chosen at each level from the m SVM's trained, based on the information gain.

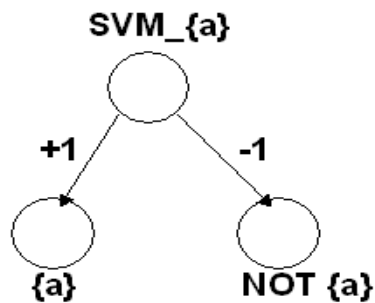


Figure 4: Each independent binary SVM

To determine the best SVM at a given node, entropy is found for each SVM, based on the number of positive instances classified by each.

$$H(\mathbb{X}) = - \sum_{i=1}^n P(\mathbb{X} = i) \cdot \ln(P(\mathbb{X} = i))$$

It is then used to calculated the information gain based on the entropy of the root node.

$$IG(\mathbb{X}, \mathbb{S}) = H(\mathbb{S}) - \sum_{v \in Values(\mathbb{X})} \frac{|\mathbb{S}_v|}{|\mathbb{S}|} H(\mathbb{S}_v)$$

where A is each decision (attribute) and S is remaining document of the node at each given level.

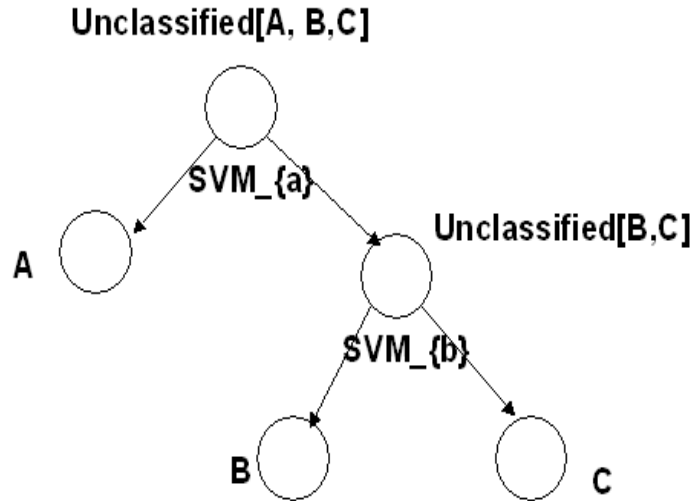


Figure 5: Example of multi-class classification in decision tree representation

In the figure 5 above, there are three classes (categories): A, B, C. At the first level, we train three different SVM, using SVMLight[16][15], one for each classes. Then, we chosen the best one among those three SVMs, by calculating the information gain for each of them. Once the best SVM is determine, we classify the instances (documents) at that level by expanding the tree with binary branches. The left child node contains the instances, which have been classified as positive document by the best SVM. To reduce the complexity of the problem, we set the left node to be leaf node, and stop investigating further on the node. Therefore, the left node may contain some false positives case, as SVM may wrongly classified for the best SVM. All the other instances (documents) are passed down to the right node where another decision is performed. We repeat the same process till all the classes (categories) are classified.

2.4 Experimental Result

We run our classification program into three categories (Travel, Health, and Sports) with 700 documents as train set, and 300 articles as testing sets. Then, we run our program with different size of feature space, and get the corresponding accuracy as follow:

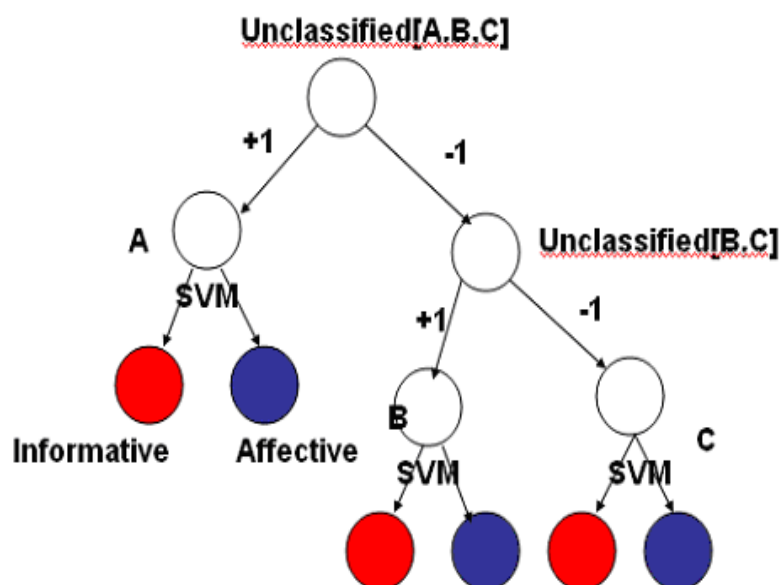
# of features	Travel	Health	Sports
73 features	52.84%	58.52%	61.02%
1000 features	85.61% (111)	81.05% (96)	72.16% (80)
3000 features	86.74% (108)	83.29% (89)	78.14% (87)

From the table, we can observe that the documents related to "Travel" was classified at the root with 85% of accuracy, but the "Sports" was classified the last, has relatively poor accuracy of 72%. The reason for this was we did not further process the instances in each left (leaf) node at each level, since there may contain some false positive case (wrongly classified) in the left child node. Also, we tried to observe the effectiveness of the feature selection to this experiment. We found out that the accuracy is dramatically increase after we increase our features, but this improvement stops when we have sufficient large enough of features. However, we were not be able to specify the precise number of features in StringToWordVector in WEKA[14], so we only could present this approximated result. Therefore, a more precise experiment on feature selection should be consider in the future tasks.

2.5 Detecting Informative and Affective Articles

A blog contains professional journal or, more often, frequently-updated views and personal remarks about a range of issues. Mood classification is increasing interest in various research communities, and is useful for various applications, such as assisting behavioral scientists and improving doctor-patient interaction[4]. Based on previously classified results, we perform additional task, and consider two genres of content in each category. Generally speaking, for each category, there are some articles the introduce some objective information while there are some articles that people express their ideas, opinion, feels etc. This approach is done illustrated by Xiaochuan Ni[3].

In order to implement this function in our project, we adapted the technique of Subjectivity Recognizing. Much work in this area has been done in Natural Language Processing. We tried to recognize the representative features in each category. In our observation, the affective articles are easier recognized. We selected 50 top representative words that most express affection in our project. The figure above describes how we judge whether an article is an informative article or affective article. We checked whether some of those 50 representative features appeared in some articles, then we label the value true (informative), false (affective) for each article.



We use the SMO classifier from Weka[14] to run the same data set. And the experimental result is showed in the following table:

genuine	feel	friend	feeling	happy
disgusted	like	hate	blessing	together
evening	today	joy	mother	emotion
remember	hope	fine	good	tomorrow
nice	day	cry	remember	hope
everyday	think	attempt	worry	sad
agree	disagree	suggest	suffer	undergo
live	believe	share	faithful	pleasant
pleased	lonely	sick	tragic	fine
nervous	scared	scare	careful	surprise

The accuracy and precision rate of the experimental results are listed below.

	Sport	Travel	Health
Accuracy	88.35%	83.4746%	84.30%
Precision	92.60%	88.2%	89.40% (80)

As we can see. The general accuracy rate for three group are from 83.4746% to 88.3459% and the precision rates are from 88.2% to 92.6%. The exper-

imental results shows that the SVM and Subjectivity Recognizing have a good performance in separating informative and affective articles.

3 Conclusion

As more and more people and enterpriser post their information in their weblogs, weblog space is of great potential in knowledge discovery. Finding an efficient way to in weblog space classification becomes very important.

In our project, we have addressed the task of separating weblog articles into 3 categories: Sport, Travel, and Health. We achieved satisfactory result in our experiment (70%-85% of accuracy).

Furthermore, in order to further investigate our article, we tried to detect the informative and affective articles in each category. We selected representative features for Informative category to judge whether an article is informative or affective and used Weka[14] SVM to test the data set, and the experimental result yield amazingly well - 85%-90% of accuracy rate.

Nevertheless, the current tree only expands to one side (right) and hence the misclassified instances(documents) are not considered for reclassification into the correct class. The accuracy of the classifier could be improved further by considering those false positives for reclassification. Moreover, we could conduct more complex combination with other classification algorithm, for instance, Naive Bayes approach, at the root and then SVM on the other nodes. So that, it would be an interesting topic to find which combination improves the efficiency of the classifier to the maximum.

Feature Selection is also worth for further investigate for this project. Many interesting studies have suggested that various different feature selection method may be the most suitable for this data source, and as a result, improvement to the performance[9].

4 Acknowledgement

The authors would like to thank Dr. Jin Tian for some advices at the beginning. And we need to give our thanks to Rafael Jordan, Lucas Bonansea, Yi Wang, who provided us very helpful advices in feature space selection.

References

- [1] Vladimir Vapnik. *The Nature of Statistical Learning Theory*, 1995
- [2] K. P. Bennett, J. A. Blue *A Support Vector Machine Approach to Decision Trees*
- [3] Xiaochuan Ni, Gui-Rong Xue, Xiao Ling, Yong Yu, Qiang Yang, *Exploring in the Weblog Space by Detecting Informative and Affective Articles*, 2007
- [4] Gilad Mishe *Experiments with Mood Classification in Blog Posts*
- [5] Thorsten Joachims, *Text Categorization with Support Vector Machines: Learning with Many Relevant Features*
- [6] Kai-Bo Duan, S. Sathiya Keerthi, *Which is the Best Multiclass SVM Method: An Empirical Study*
- [7] Christopher M. Bishop, *Pattern Recognition and Machine Learning*
- [8] Richard Duda, Peter Hart, David Stork *Pattern Recognition 2nd Edition*
- [9] Rudy Prabowo, Mike Thelwall, *A comparison of feature selection methods for an evolving RSS feed corpus*
- [10] Yang, Y. and Pedersen, J.O, *A comparative study on feature selection in text categorization. In Machine Learning: Proceedings of the Fourteenth International Conference (ICML'97), 412-420, 1997.*
- [11] M. Hodder. *Live Web Search:*
<http://www2.sims.berkeley.edu/courses/is141/f05/schedule.html>
- [12] ROME v0.9, <https://rome.dev.java.net/>
- [13] HTMLParser v1.6, <http://htmlparser.sourceforge.net/>
- [14] WEKA v3.5.7, <http://www.cs.waikato.ac.nz/ml/weka/>
- [15] SVMLight v6.01, <http://svmlight.joachims.org/>
- [16] Java Native Interface for SVM-light-6.01,
<http://infolab.stanford.edu/theobald/>
- [17] Com S 573. Machine Learning <http://www.cs.iastate.edu/jtian/cs573/>