Novel Frequent Sequential Patterns based Probabilistic Model for Effective Classification of Web Documents

Hammad Haleem[1], Pankaj Kumar Sharma[2] and M M Sufyan Beg[3].

{ hammadhaleem, sharmapankaj1992}@gmail.com, mmsbeg@eecs.berkeley.edu
Department of Computer Engineering
Jamia Millia Islamia
New Delhi, India

Abstract

Web page classification has been one of essential tasks in web information retrieval such as delivering content specific search results, focused crawling and maintaining web-directory projects like DMOZ, etc. This paper presents a novel probabilistic web page classification scheme that utilizes the occurrences of frequent sequential patterns to determine the class of the document. As being suggested by many previous works in the field of text mining, patterns possess more relevant information about the document than individual words. This paper is an attempt to successfully make use of this hypothesis for classification of web documents. After testing this novel approach on RCV1 dataset, we were able to obtain classify the test documents with 88% accuracy.

1. Introduction

With the expansion of the World Wide Web (WWW) from early 2000s, Internet has shown a rapid growth within last decade of its existence. It has been estimated that currently more than 2.4 billion people have access to Internet through their mobile, desktop and tablet devices [21]. During the last decade Internet observed a very high growth rate.Internet World Statistics reveals that the world Internet usage growth has increased by 480.4% during the period 2000-2011 [1]. With the increasing size of Internet and connection of more and more networks to Internet, Internet is home to more than 35% of data available in worldwide.

According to Netcraft's January 2012 survey [1], more than 584 million web sites exist on the Internet and out of which, nearly 175.2 million are active. Today there are several different tools available for an Internet user to locate and identify relevant information on the Internet. These tools can be broadly classified as Crawler based search engines [22] e.g., Google, Bing, Yahoo, Duck-Duck-Go etc., Meta search engines [24] e.g. Meta-crawler, Clusty etc. and Subject Directories [25] like DMOZ (Directory Mozilla), Librarians Internet Index (LII), etc. The crawler based search engines and subject directories maintain their own data repositories, whereas meta search engines do not maintain any such data repositories, instead they depend on indices of other search engines and subject directories to answer user queries. The database maintained by any crawler based search engine is quite large and have considerably larger amount of data indexed in their databases as compared with subject directories. Directory Mozilla (DMOZ) [2] i.e., dmoz.com has 93,446 editors for 1.2 million categories and has indexed 5.26 million websites, which is only 2.5% of the total active web sites available on the Internet today.

Majority of the web directories are edited and maintained by human effort [2]. Subject directories are popular due to proper classification of data in several categories. A large website directory could be quite helpful in improving the quality of search results, filtering web content, developing knowledge bases, building vertical (domain specific) search engines. Hence need for automating the process of classification of websites based on their content arisen recently. This paper presents a sequential pattern based naive Bayesian (NB) probabilistic model for the automatic classification of web-pages. Naive Bayes probabilistic model is one of the most effective and straightforward model for text document classification and has exhibited good results in previous studies conducted for data mining. The model is quite optimized and has quite effectively worked to classify websites based on their content in real-time. Also it can be scaled for large databases. The model presented in this paper has given accuracy around 88% or the test data under consideration.

Classification lies under the domain supervised learning in machine learning. In supervised learning, a set of labeled data, each labelled by one or more

978-1-4799-6758-2/14/$31.00 ©2014 IEEE

class labels, is used to train a classifier and subsequently, the trained classifier is used to label future examples. In general, the problem of web page classification can be subdivided into multiple sub problems like subject classification, functional classification, sentiment classification, etc. [3]. The Subject classification problem is concerned about the subject or topic of a web page. For example, to classify whether the web page is about “economics”, “politics” or “sports” is an instance of subject classification. Functional classification cares about the role that the web page plays. For example, deciding a page to be a “personal homepage”, “course page” or “admission page” is an instance of functional classification. Sentiment classification focuses on the opinion that is presented in a web page, i.e., the author’s attitude about some particular topic. In this paper our main topic of interest would be subject classification of web-document even though the proposed technique could be extended to other types of classifications. Here onwards, the word “Classification” will refer to “Subject Classification”.

In order to extract useful information from unstructured data such as text documents and web pages many data mining has been proposed. A significant amount of research work has already been done on Information Retrieval and classification of unstructured data sources. These techniques makes use of the concepts of association rule mining [4], frequent itemset mining [29], sequential pattern mining [28], maximal pattern mining [30], closed pattern mining, etc. Most of these methods were introduced for development of efficient state of art methods for extracting patterns from documents in a reasonable amount of time. Still due to large number of generated patterns from these techniques it remains an open research issue on how to effectively use and update such patterns. This paper focuses on developing a model that uses the properties of closed frequent patterns to perform classification of web pages.

The paper is divided into 10 sections. Section 2 describes related work in the area of text mining. Section 3 describes classification of web pages and related problems. Section 4 talks more about sequential patterns and their use in classification. Section 5 explains about the naive Bayes classifier. Section 6 describes pattern deploying methodology used in the training and testing phases. Section 7 and 8 provide information about the experimental setup and results. Finally section 9 presents a conclusion followed by references.

2. Related work

A number of text content based classifier has been developed in the past for classification of web pages. Most of the previous works based upon k-nearest neighbors [5, 6, 7, 8, 31], support vector machine [9], etc. Most of these previous techniques make use of word frequencies in the documents for text based classification. As we stated in our hypothesis and also supported by certain other works, as done in[10], words present in frequently occurring sequential patterns [28] provide more information related to the document than any other those obtained by other methods like TF-IDF [11]. Under above stated circumstances, our study aims to develop a new technique to classify Web-page automatically by supervised machine learning using a Naive Bayesian probabilistic model [12] from the closed sequential patterns of the training documents. Further as frequent patterns could be large in number if the minimum threshold support is less, we make use of closed sequential patterns of each document to obtain the words’ frequencies. The model we develop is certainly an improvement over today’s classifications’ probabilistic model [13,14], which uses TF-IDF scheme [11] for the training.

For the extraction of closed sequential patterns, we have used BIDE [15] algorithm and further apply algorithm 1 (presented in Section 6)for obtaining D-patterns (DP) [16] related to the documents. Further, a modified Naive Bayesian Classifier is being used. To train the classifier, we used algorithm 2. For the testing part we extract frequent patterns from each of testing documents and use these patterns to calculate the probability of each document and find out where they fit best, as given in algorithm 3.

3. Classification of Webpages

The idea of creating a web page classifier can be thought to lie directly under the context of machine learning methodology, specially the techniques developed for text classification [8, 9, and 17] which could be effectively used for classification of web pages. Though the problems of text document classification and web page classification being similar in nature, web page classification has its own pluses and minuses. The web pages provide more structured information with the use of HTML. On the negative side, the amount of noise (content such as advertisement banner and navigation bars) in HTML page is also very prevalent and could be intervening if not taken into account. Another major issue in web pages is that web pages can largely variable number of words. Some of the pages may contain thousands of words and while others may have very low words count, i.e., there is no uniformity in size of web pages.

Instead of using all the words in a web page to find the category to which document belongs, representative features from the web pages may be used for classification. In this paper we maintain the hypothesis that words occurring in frequently appearing patterns could be more informative than all the words combined. This strategy is already a prevent way of Knowledge Discovery from text databases, as used in [16].

As this strategy is a pure-text based classification technique, if we directly apply it for web pages classification, noisy context will cause a bias, making it possible to get biased towards the noisy content instead of the main topics and important content of the web-page. Thus, we require employing an intelligent pre-processing technique to extract the text content of the web page.

As the pre-processing step, all HTML tags are needed to be removed from the web pages, including punctuation marks. Stop words removal is also a must in the extracted data, as stop words are common to all documents and does not contribute much in real classification process [18]. A stemming algorithm [19] is required to be applied to reduce words to their basic stem. One such frequently used stemmer is the Porter's stemming algorithm [23]. More information about the preprocessing stage is provided in upcoming sections. Each text document obtained after pre-processing is represented as a list of words in a set of paragraphs, in this paper, dp(d) represents set of paragraphs for a given document, d. BIDE [15] algorithm is then applied on each of these documents for the purpose of obtaining d-patterns (explained in upcoming section). For training the respective classifier, documents are fed to Algorithm 1.

4. PTM Pattern Taxonomy Model

In this paper we used the PTM model to represent the information obtained from the web pages. For this section we would consider that we are not mining on the website data but normal text data. This section assumes that all documents are split into paragraphs. So a given document d yields a set of paragraphs ps(d). D is the training set of documents, which consists of a set all documents used for training the classifier. T = {t1, t2... tm} is the set of terms (or keywords) which can be extracted from the set of training documents, D. The sections below briefly describes about the sequential patterns and closed sequential patterns in specific. The Pattern mining algorithm is also explained well in the sections below.

4.1. Why sequential patterns mining

In the recent times many different ways to identify patterns have been proposed. But mainly three types of patterns discovery algorithms are popular. These include Text based approaches, Phrase based approach and Sequential patterns approach for mining frequent patterns. The text based approach is a quite old and established approach as compared to other two. The text based methods for patterns discovery have been studied quite thoroughly in many recent years. The major problem with any text based approach, are the issue of Polysemy and synonymy. The phrase based approaches though said to contain a lot more information than any other methods have failed to generate any good results. This paper used a modified popular sequential pattern discovery technique described in [15]. The pattern discovery algorithm presented here has a better execution time than original algorithm and hence is useful for using it in a classifier with real-time classification capabilities. But the sequential pattern based techniques for pattern discovery also fails sometimes due to low frequency and misinterpretation of patterns. The PTM model has produced quite effective results when applied on the text document and even in the website classification the results are quite encouraging.

4.2. Frequent patterns and related terminologies

In this section we describe some important terms in brief due to space constraints, which will be frequently used and referenced in this text.

4.2.1. Termset: A term set t, for a document d is defined as a set of terms given by {t_1, t_2… t_n}. Where t_n belongs to the Document D.

4.2.2. Covering set: Covering set of a termset X may be defined by set of paragraphs in which X appears. Covering set of X is denoted by $\ulcorner X \urcorner$ is defined as

$$\ulcorner X \urcorner = \{DP \mid DP \in PS\ (d), X \subseteq DP\}.$$

Where, PS (d) returns set of paragraphs in document d.

4.2.3. Absolute Support: Absolute support of termset X, may be defined as number of paragraphs in which X appears, i.e., $sup_a(X) = |\ulcorner X \urcorner|$.

4.2.4.Relative Support: Relative support of termset may be defined as ratio of paragraphs in which it appears to total number of paragraphs, i.e.,
$sup_r(X) = sup_a(X) / |PS\ (d)|$.

4.2.5.Frequent Pattern: A termset X is called a frequent pattern if $sup_r(X) \geq$ min_sup, where min_sup is a minimum threshold support, provided by the user.

4.2.6. Closed Frequent Pattern: A frequent pattern is called closed if none if its subset having same support is a closed pattern.

4.3.Closed Sequential patterns

4.3.1. Sequential Pattern: A sequential pattern is an ordered list of terms. A sequential consisting of terms is represented as $s = <t_1, t_2, t_3… t_n>$.

4.3.2. Subsequence: A sequence x is said to be sub-sequence of sequence y, represented by $x \sqsubseteq y$ iff x could be obtained by removing zero or more characters from y. More formally we can define, if $x = <x_1, x_2 … x_n>$ and $y = <y_1, y_2… y_j>$ then $x \sqsubseteq y$ iff (x1 = yk), (x2 = yk+1)… (xn = yk+n-1), for some n ∈ [1, j] and k ∈ [0, n].

4.3.3.Covering set of a sequence: Covering set of sequence X is denoted by $\ulcorner X \urcorner$ is defined as :$\ulcorner X \urcorner = \{ dp \mid dp\in PS(d), X \sqsubseteq dp\}$. PS (d) returns set of paragraphs in document d.

4.3.4.Frequent Sequential Pattern: A sequence X is called a frequent sequential pattern if $sup_r(X) \geq$ min_sup, where min_sup is a minimum threshold support, provided by the user.

4.3.5.Closed Frequent Sequential Pattern: A frequent sequential pattern is called closed, if none if its sub-sequences having same support is closed.

Note: Support (absolute and relative) are defined in the same way as given in section 4.2. From now onwards in this paper we will reference pattern in place of sequential patterns sequential patterns.

4.4. Pattern deploying method

To store the semantic information related to the document, we use D-patterns [16]. D-patterns are calculated by applying the algorithm given below. The algorithm assumes that for each document d, we have as set of closed sequential patterns. These patterns are calculated by use of BIDE algorithm [15].

let sp be the set of closed sequential patterns for d.

```
Then
    DP = {}
    foreachsp∈CSP( d ) do
        DP = DP ⊕ sp
    end
```

Where, CSP (d) returns closed sequential patterns for document d. ⊕, the composition operation is defined equation 1, detailed description in [2]. We have further elaborated more on the technical details for this algorithm in the section 7 of the paper. Let p1 and p2 be sets of term-number pair's p1 ⊕p2 is called the composition of p1 and p2 which satisfies:

$$\mathrm{P1} \oplus \mathrm{P2} \;=\; \{(\mathrm{t}, \mathrm{x1}+\mathrm{x2}) | (\mathrm{t}, \mathrm{x1}) \in\; \mathrm{p1}, (\mathrm{t}, \mathrm{x2}) \in\; \mathrm{p2}\,\}$$

$$\cup :\{(\mathrm{t}, \mathrm{x}) | (\mathrm{t}, \mathrm{x}) \in\; \mathrm{p1} \cup\; \mathrm{p2}\,, \mathrm{not}((\mathrm{t}, \_) \in\; \mathrm{p1} \cap \mathrm{p2}\,\} \qquad \text{- (1)}$$

where _ is a wild card entry.

5. Naive Bayes Classifier

Naive Bayes classifier is preferred over other classifiers because of its simple and straightforward approach and ability to apply for a wide variety of domains. Theoretically a classifier based on Bayes theorem will have minimum error rate in comparison to all other classifiers. Also naive Bayes classifier can be easily used for any number of features without much difference in performance. It can be scaled for larger databases without rise in execution time in comparison with other classifiers.

5.1.Bayesian Classifiers: These are statistical classifiers which predict the class membership probabilities of tuples, which means probability of a given tuple to be in a particular class. Studies comparing classification algorithms have found that a simple Bayesian classifier known as the naive Bayesian classifier is comparable in performance with decision tree and selected neural network classifiers. Bayesian classifiers have also exhibited high accuracy and speed when applied to large databases. In theory Bayesian classifiers have the minimum error rate in comparison to all other classifiers. However, in practice this is not always the case owing to inaccuracies in the assumptions made for its use, such as class-conditional independence, and the lack of available probability data.

6. Pattern Deploying :

This section discusses more about the pattern deploying method. We have calculated the D-patterns for each of the documents. We use the BIDE algorithm presented in [15] to find D patterns from the document. These patterns were recorded and then composition operation (as described in section 4) is applied to formulate a pair of word and its corresponding weight in the document. This approach has helped us to increase the speed of classifier compared to a normal Naive Bayes classifier that operated on list of words. Rather than having a full list of words we now had a smaller but a more important wordlist for the document. That improves the overall accuracy as well as the required computation cycles for execution.

We used the BIDE algorithm for computing closed sequential patterns. The computation of closed pattern for any document can be a really computation intensive task but the BIDE algorithm used here has help certainly by pattern pruning method applied. Because of computational speedup achieved it’s now possible to apply this algorithm in real time calculation of D-patterns from any document and hence resultant real time category calculation of web classifier.

```
Patterns Deploying algorithm
(Pattern(d, min_sup) )

INPUT: web Document, d; minimum support, min_sup
OUTPUT: d-pattern of document d, dp
let PS(d) returns set of paragraphs for document, d;
SP = Bide(PS(d))
dp = {}
foreachpatterns $p \in SP$ do
        dp = dp$\oplus$p
end
```

Algorithm 1: Pattern deploying algorithm

7. Experimental Setup

This section explains the setup of the entire experiment. We used the standard Reuters Corpus dataset Volume 1 (RCV1) [20] for testing as well as training of data. The Reuters corpus has a total of 806,791 pages classified under 90+ categories. Then popular data cleaning and data extraction techniques were applied to extract useful data from the dataset. Once the data extraction process is finished we trained our classifier according to k-fold strategy [26] for various values of k and classifier accuracies for different values of k are obtained. All these processes are briefly discussed in upcoming sections.

7.1.About the Dataset

The data set used here is a collection of documents related to pre-defined categories. We used 7 pre-defined categories used. Number of documents in each category, i.e., number documents used for train set and number of document used for test set is given in Table 1. The documents were taken from two different sources, firstly we used the Reuters corpus version 1 (RCV1) [20] and secondly on our own crawled data set. We compiled a dataset where most of the documents were taken from popular news networks like BBC, CNN, and Reuters belonging to our predefined categories. In this experiment we have only considered the web pages in English language. Any multimedia or non-English text content was removed from the crawled documents. The dataset consisted of total of 14157 documents, divided into 7 categories, having a distribution as given below:

Categories	**Train**	**Test**	**Total**
Economics	1616	405	2021
Sports	1611	403	2014
Religion	1580	395	1975
Entertain	1620	405	2025
Health	1626	407	2033
Politics	1640	411	2051
Education	1628	407	2035
Total	11321	2833	**14154**

Table 1: Division of dataset for k-fold analysis with k=5.

7.2. Pre-processing the Dataset

We have used a combination of two kinds of data sets, first the standard pre-formatted RCV1 and second a set of web pages collected by our crawler. The python regular expression capabilities were extensively use to identify and extract information from web-pages. We also used the BeautifulSoup module to read web-pages and extract information from them. . Beautiful Soup is a Python library designed for easy and effective extraction of information from web-pages based upon HTML and XML tags. The title of the documents is used as individual paragraphs as they also provide important information related to the document.

During the initial phase we removed any ASCII character or special symbol from the data. CSS and JavaScript were also removed as they don't have any semantic significance. Images have been used quite frequently in buttons or links for navigation and display name of the organization, however this experiment only deals with the textual content on the web page hence the images were also removed.

During the final phase of preprocessing, we utilized the standard Stop-Word list defined in NLTK [27]and BOW. Along this we used the standard stemming and lemmatization techniques like porter stemming algorithm and Wordnetlemmatizer was used to process and reduce each word to its lowest forms. Furthermore we used the pos-tagger present in NLTK was used to identify the form of current word (like noun, adjective or verb) and further processed by the wordnet-lemmatization to convert words in lowest forms. We divided each paragraph of text in lines. These lines were processed and the order of words was preserved for subsequent application of sequential pattern discovery algorithm on the text.

7.3. Vocabulary generation :

The words that occurred commonly in most of the web pages were considered as stop words. A custom list of such words is present in Table 2. These common words were considered as Stop-Words and removed directly from the document. Sometimes webmaster inflated the Meta and text descriptions causing incorrect classification, we normalized such repeating keywords to reduce the impact of site promotion techniques applied by webmasters. This step was performed during the cleaning phase.

Table 2: Web-page specific Stop Words
login, view, browser, website, web, online, search, keyword, designed, copyright, rights, reserved, click, search, welcome, email, click, contact, developed, mail, home, page, feedback, webmaster

7.4. Testing and training

We followed the k-fold cross validation strategy (with k=10) to decide the number of training and testing examples. The documents in each category are divided into 5 equal partitions say D_1, D_2, D_3... D_5. Training and testing will be performed k times. In iteration I, partition Di in each category will be reserved as a training set, and the remaining partitions will be collectively used to train the model. That is, in the first iteration, subsets D_2... D_5 collectively serve as the training set to obtain a first model, which will be tested on D_1 of each category; the second iteration will be trained on sets D_1, D_2, D_3... D_5 and will be tested on D_2. And so on. Prior probability of each category will be same since we have taken equal number of documents in each category. Training set was stored in the form of hash tables of hash tables. Each hash table in first level stands for category. And hash tables in second level will be containing the frequency of words in that particular category. The posterior probability with

Laplace's correction was calculated using the formula [29] [30]:

$$\mathbf{P\ (wk|c) = (n_k +1) / (n + |Vocabulary|)} \quad - (11)$$

Where N_k stands for number of occurrences of word W_k in category C, N is total number of words in given category and |vocabulary| stands for number of words in training set. In order to classify a document say X, the probabilities of each word of document in a given category were calculated from hash table , then they were multiplied together .The probability values obtained for individual categories were sometimes going below the floating point limit of Python (in order of $x*10^{-300}$) . We were able to find a solution for this problem by taking the Log_2 (P_x), of the obtained probability and summing them up. The obtained number was then further multiplied by -1, to get overall positive value.

$$\mathbf{C=arg\ Max\ (-log_2\ (P_c) - \sum log_2\ (P_{wk}\ |\ C))} \quad - (12)$$

The calculated values were compared to find the minimum of all. The category with minimum log of probability was selected as the classified class for a document. The equation 12 helps to clearly understand the solution of the mentioned problem. The Naïve Bayes algorithm [32] to train and test the classifier is given below:

```
Probabilistic Model Generating Algorithm
( PromoGen ( D, C, min_sup) )

INPUT: Set of Documents, D; Set of categories, C;
minimum support for closed patterns, min_sup
OUTPUT: Probabilistic hash-set, PC
PC = {}
foreachcategory, $c \in C$ do
    PC[c] = {}
    let CS(D, c) gives list of documents belonging to
                                        category c;
    foreachdocument $d \in CS(D, c)$ do
        dp = DPattern(d, min_sup)
        PC[c] = PC[c] $\oplus$dp
    end
end
```

Algorithm 2: **Probabilistic Model Generating Algorithm**

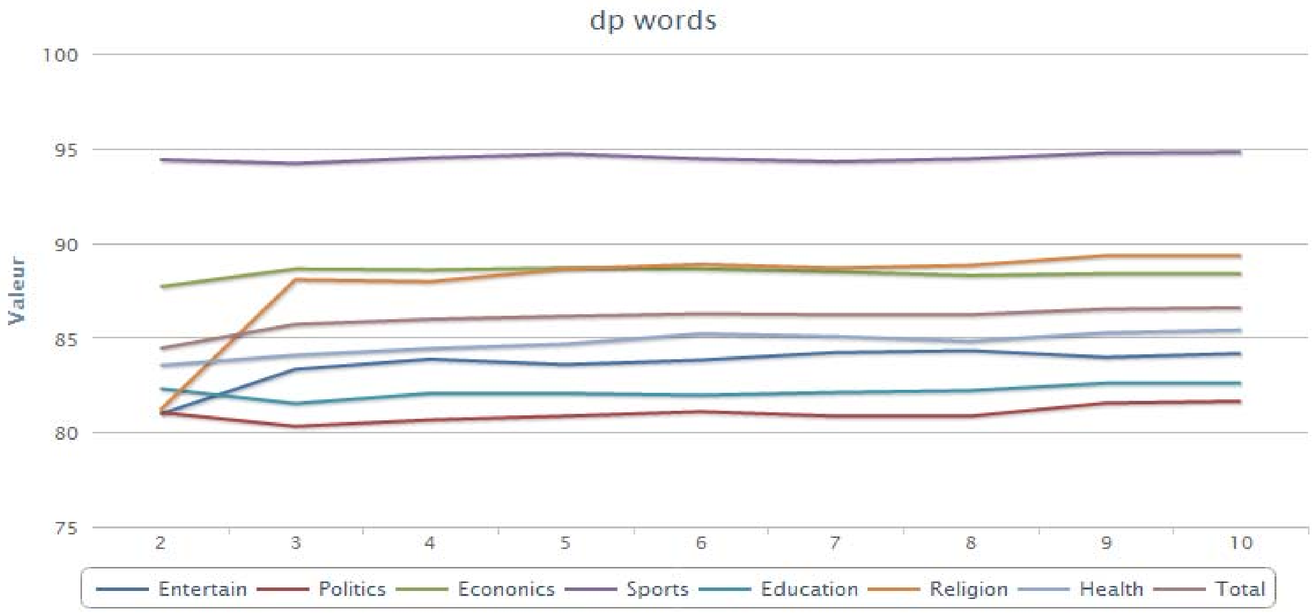
```
Testing Algorithm

INPUT: Test document, d; Set of categories, C;
Probabilistic has-set, PC
OUTPUT: Most probable category for document d,
category let WS(d) returns a list of words in
document d;
max_prob = 0.0
foreachcategory $c \in C$ do
        pc = PC[c]; i.e, Words frequencies related
to category c
        ci = 1.0
        foreach word $w \in WS(d)$ do
                ci = ci * (pc[w]/word_count)
        end
        if ci >max_probthen
                max_prob = ci
                category = c
        end
end
```

Algorithm 3: Testing Algorithm

8. Results

The classifier was executed for the given dataset with 14000 examples in seven categories for various values of k. With each value of K, We were able to record corresponding accuracy Graph 1,2,3 depicts the relation between the increasing value of K and its effect on the accuracy of classification of documents for the range 2<=k<10. We have considered all the three scenarios for classification. For finding accuracy for each value of K



Graph1: Classification results for the DP-Word relation

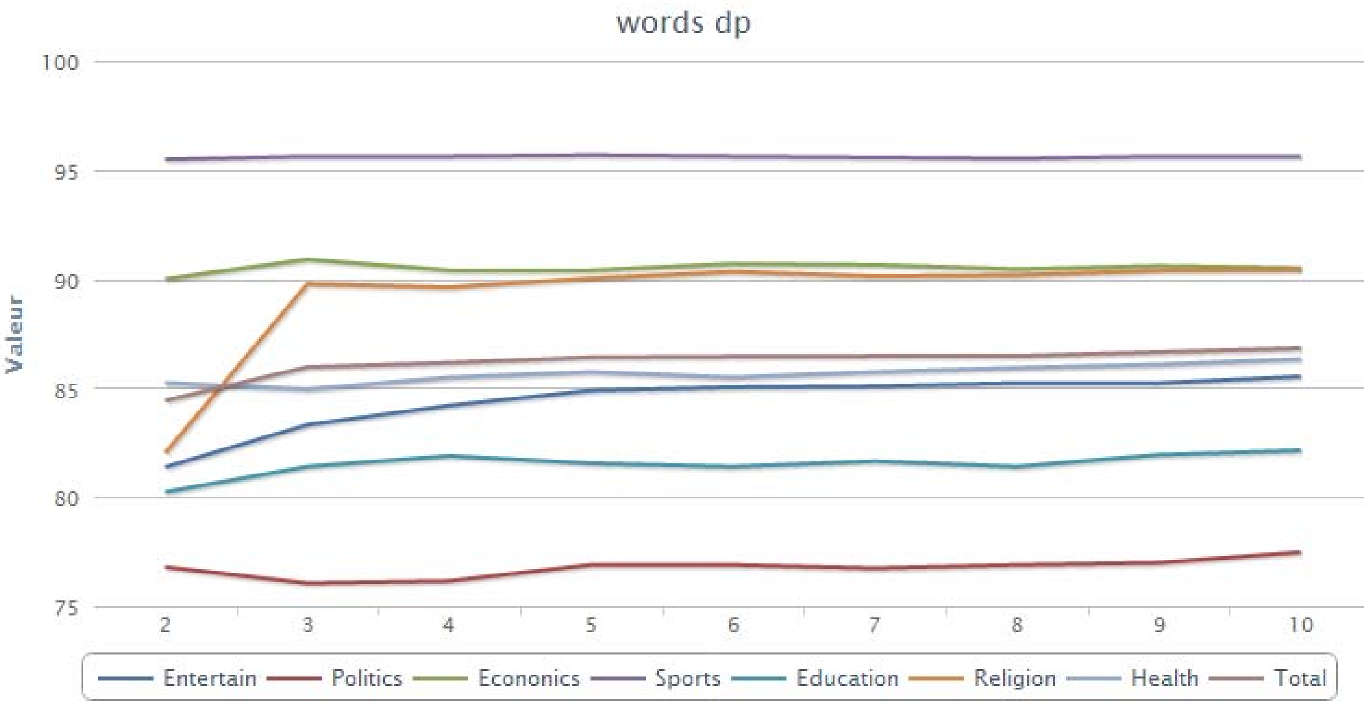
	economics	**Sports**	**religion**	**entertain**	**health**	**politics**	**education**
economics	1754	1	28	2	65	135	26
sports	10	1913	8	40	12	25	6
religion	15	2	1753	31	19	152	3
entertain	69	33	113	1609	59	107	35
health	58	2	41	22	1666	78	166
politics	202	8	196	25	66	1547	7
education	84	8	10	60	183	36	1670

Table 3: Confusion matrix for the DP-DP relationship graph.

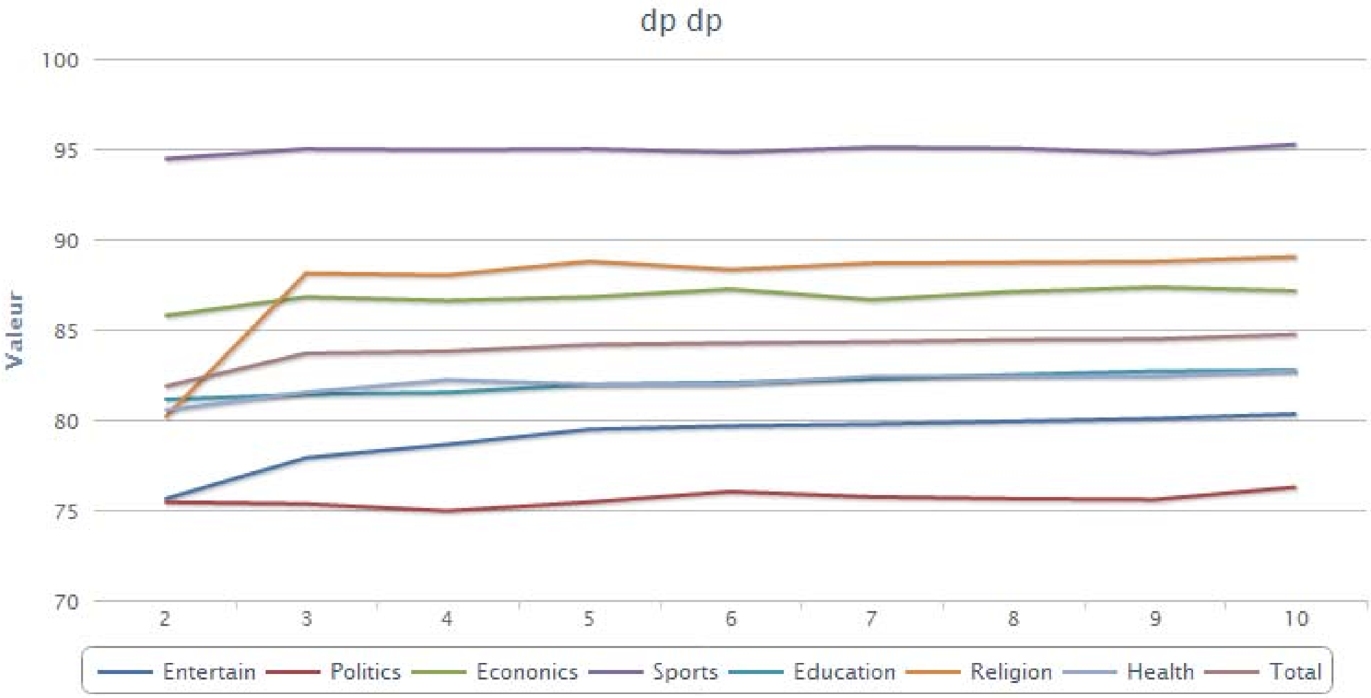
We have considered three scenarios for comparisons. First scenario is when we trained the dataset on the D-patterns and the classifier was tested on words. Performance in this scenario is presented in graph1. As the number of words present in document far exceeds the number of words in a set of frequent patterns hence the execution time of this step was greater than others.

Second scenario is when we trained the dataset on the words and the classifier was tested on D-patterns. Performance in this scenario is presented in Graph 2. As the number of words present in document far exceeds the number of words in a set of frequent patterns hence the training time of this step was greater than others.

Third and final scenario presented here is about the DP-DP, in this case the training as well as testing was performed on the D-patterns and hence the computational time required was comparatively less. The results were also similar for this also. The results can be explained as initially there is a rise in classification accuracy with increase in number of folds which can be explained in terms of better training. Then there is a slight fall in accuracy which can be explained in terms of over training of classifier. I.e. classifier is trained too much and is not able retrieve a result.



Graph2: Classification results for the Word-DP relations



Graph 3: Classification results for the DP-DP relation

It is observed that the classifier was able to achieve maximum accuracy which is above 87% for K = 5, i.e. when 4/5th of the documents were used as training set and rest for test set in Dp-Dp relationship. Using our approach we were able to get a good accuracy. The table 5 depicts confusion matrix. A classifier is good if majority of the documents lie along the diagonal of a confusion matrix. This trend is observed in table 5. This confusion matrix can be further used to find most common classification measures like accuracy, recall, error rate, specificity and f-measures etc. Based on each of the K value we generated confusion matrix for each of the category. We calculated the accuracy, recall and precision for the classifier. Confusion matrix for DP-DP relationship is given in table 3. These values were used to calculate values of precision, recall and accuracy. The achieved accuracy for the proposed technique is 85% for DP-DP classifier, 87% for DP-Word Classifiers and 88% for the Word-DP classifiers. Results are quite encouraging even when the classifier has data from overlapping categories and a single document falling into multiple categories.

In this experiment, not a too direct correlation between the value of K and the accuracy has been seen. Though a larger value of k led to a somewhat decreased accuracy. Initially increasing the value of K the accuracy increases many fold and then stabilized for the time and then again decreases for the remaining values of K. This relation could be used to find an optimal value of K for testing and training the classifier. Here K=5 has been able to generate best results.

9. Conclusion and Future Scope

The Classification Model proposed and implemented during the course of this paper utilizes the features of naive Bayes classifiers and the PTM pattern discovery model. The naive Bayes document classification algorithm discussed in this paper intelligently exploits the richness of content of webpage for effective classification. The algorithm has been utilized to categorize data into a very broad set of categories. We intelligently utilized the plus point of the PTM model proposed earlier, i.e. finding best frequent patterns from a text document quickly. The quality of frequent patterns that they represent main idea of the text and can be utilized for ranking of documents has been used to implement a Naive Bayes classifier. This classifier gave sufficiently good result on the Reuters Corpus Volume 1 Dataset. Achieved accuracy of more than 87% was quite encouraging. This approach could be utilized by the search engines and other online directory projects like DMOZ [2] for effective categorization of web pages to build an automated web directory based on the content available on the web page and type of organization in the real time. The computationally fast algorithm can help to increase the speed of classification.

Although in this experiment, only nonhierarchical and distinct categories are considered the above algorithm can also be used to classify the pages into more specific categories (hierarchical classification) by changing the feature set e.g. a website that is e-commerce may be further classified into electronics, clothes or a book selling website. Also there can be improvements in the part of pattern discovery algorithms. Though we have use the sequential pattern discovery algorithm it can be obviously better if Phrase based method can be developed for pattern mining. As phrases offer better advantages like a better meaning, for the discovered pattern.

References

[1] NetcraftInternet survey for august 2012 http://news.netcraft.com/archives/2013/08/09/august-2013-web-server-survey.html

[2] Open Directory Project: ODP http://www.dmoz.org/

[3] Xiaoguang Qi and Brian D. Davison, “Web Page Classification: Features and Algorithms”, Technical Report LU-CSE-07-010, Department of Computer Science and Engineering, Lehigh University, Bethlehem, PA

[4] R. Agrawal, T. Imielinski, and A. Swami. “Mining association rules between sets of items in large databases” In Proc. of the ACM SIGMOD Conference on Management of Data, Washington, D.C., May 1993

[5] Keller, James M., Michael R. Gray, and James A. Givens. "A fuzzy k-nearest neighbor algorithm."*Systems, Man and Cybernetics, IEEE Transactions on* 4 (1985): 580-585.

[6] Denoeux, Thierry. "A k-nearest neighbor classification rule based on Dempster-Shafer theory."*Systems, Man and Cybernetics, IEEE Transactions on* 25.5 (1995): 804-813.

[7] Beyer, Kevin, et al. "When is “nearest neighbor” meaningful?." *Database Theory—ICDT’99*.Springer Berlin Heidelberg, 1999.217-235.

[8] Soucy, Pascal, and Guy W. Mineau. "A simple KNN algorithm for text categorization."*Data Mining, 2001.ICDM 2001, Proceedings IEEE International Conference on*.IEEE, 2001.

[9] Tong, Simon, and Daphne Koller. "Support vector machine active learning with applications to text classification." *The Journal of Machine Learning Research* 2 (2002): 45-66.

[10] Wu, Sheng-Tang, et al. "Automatic pattern-taxonomy extraction for web mining." *Web Intelligence, 2004. WI 2004. Proceedings.IEEE/WIC/ACM International Conference on*.IEEE,

2004.

[11] Salton, Gerard, and Michael J. McGill. "Introduction to Modern Information Retrieval." (1983).

[12] Kononenko, Igor. "Semi-naive Bayesian classifier."*Machine Learning—EWSL-91*.Springer Berlin Heidelberg, 1991.

[13] Yu, Hwanjo, Jiawei Han, and KC-C. Chang. "PEBL: Web page classification without negative examples." *Knowledge and Data Engineering, IEEE Transactions on* 16.1 (2004): 70-81.

[14] Qi, Xiaoguang, and Brian D. Davison. "Web page classification: Features and algorithms." *ACM Computing Surveys (CSUR)* 41.2 (2009): 12.

[15] Wang, Jianyong, and Jiawei Han. "BIDE: Efficient mining of frequent closed sequences." *Data Engineering, 2004.Proceedings.20th International Conference on*.IEEE, 2004.

[16] Zhong, Ning, Yuefeng Li, and Sheng-Tang Wu. "Effective pattern discovery for text mining."*Knowledge and Data Engineering, IEEE Transactions on* 24.1 (2012): 30-44.

[17] McCallum, Andrew, and Kamal Nigam. "A comparison of event models for naive bayes text classification." *AAAI-98 workshop on learning for text categorization*.Vol. 752. 1998.

[18] Silva, Catarina, and BernardeteRibeiro. "The importance of stop word removal on recall values in text categorization." *Neural Networks, 2003.Proceedings of the International Joint Conference on*.Vol.3.IEEE, 2003.

[19] Lovins, Julie B. *Development of a stemming algorithm*. MIT Information Processing Group, Electronic Systems Laboratory, 1968.

[20] Lewis, David D., et al. "Rcv1: A new benchmark collection for text categorization research." *The Journal of Machine Learning Research* 5 (2004): 361-397.

[21] INTERNET USAGE STATISTICS “The Internet Big PictureWorldInternet Users and Population Stats” , June 2012 http://www.Internetworldstats.com/stats.htm

[22] Brin, Sergey, and Lawrence Page. "The anatomy of a large-scale hypertextual Web search engine."*Computer networks and ISDN systems* 30.1 (1998): 107-117.

[23] Porter, Martin F. "An algorithm for suffix stripping." *Program: electronic library and information systems* 14.3 (1980): 130-137.

[24] Joachims, Thorsten. "Optimizing search engines using clickthrough data." *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*.ACM, 2002.

[25] Piontek, Sherry, and Kristen Garlock. "Creating a world wide web resource collection."*Collection Building* 14.3 (1995): 12-18.

[26] Kohavi, Ron. "A study of cross-validation and bootstrap for accuracy estimation and model selection."*IJCAI*.Vol. 14.No. 2. 1995.

[27] Bird, Steven. "NLTK: the natural language toolkit." *Proceedings of the COLING/ACL on Interactive presentation sessions*.Association for Computational Linguistics, 2006.

[28] Srikant, Ramakrishnan, and Rakesh Agrawal. *Mining sequential patterns: Generalizations and performance improvements*. Springer Berlin Heidelberg, 1996.

[29] Han, Jiawei, Jian Pei, and Yiwen Yin. "Mining frequent patterns without candidate generation."*ACM SIGMOD Record*.Vol. 29.No. 2.ACM, 2000.

[30] Srikant, Ramakrishnan, and Rakesh Agrawal. *Mining sequential patterns: Generalizations and performance improvements*. Springer Berlin Heidelberg, 1996.

[31] Guo, Gongde, et al. "AnkNN model-based approach and its application in text categorization." *Computational Linguistics and Intelligent Text Processing*.Springer Berlin Heidelberg, 2004.559-570.

[32] A. McCallum and K. Nigam, “A comparison of event models for Naïve Bayes text classification”, in AAAI/ICML-98 Workshop on Learning for Text Categorization, 1998, pp. 41-48.