1.ABSTRACT

World Wide Web (www) is a large repository of information which contains a plethora of information in the form of web documents. Information stored in web is increasing at a very rapid rate and people rely more and more on internet for acquiring information. Internet World Stats reveal that world internet usage has increased by 480% within the period 2000- 2011. This exponential growth of the web has made it a difficult task to organize data and to find it. If we categorize data on the internet, it would be easier to find relevant piece of information quickly and conveniently. There are some popular web directories projects like yahoo directory and Mozilla directory in which web pages are organized according to their categories. According to a recent survey, it has been estimated that about 584 million websites are currently hosted on the internet. But these internet directories have only a tiny fraction of websites listed with them. The proper classification has made these directories popular among web users. However these web directories make use of human effort for classifying web pages and also only 2.5% of available webpages are included in these directories. Rapid growth of web has made it increasingly difficult to classify web pages manually, mainly due to the fact that manually or semi-automatic classification of website is a tedious and costly affair. Because of this reason web page classification using machine learning algorithms has become a major research topic in these days. A number of algorithms have been proposed for the classification of web sites by analyzing its features. In ou project we would like use a fast, effective, probabilistic classification model with a good accuracy based on machine learning and data mining techniques for the automated classification of webpages into different categories based on their textual content.

2.INTRODUCTION

The World Wide Web (WWW) started in the year 1991 and has shown a rapid growth within last two decades. It is estimated that today more than 3.5 billion people are using internet. The number of people using the Internet is rapidly increasing. Internet World Statistics reveals that the world Internet usage growth has increased by 480.4% during the period 2000-2011 [1]. It was also observed that more than 35% data available in whole world is stored in internet. According to Netcraft's January 2012 survey, 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 average internet user to locate and identify relevant information on the internet. These tools can be broadly classified as 1.Crawler based Search Engines (SE) e.g., Google, Bing, Yahoo, Duck-Duck-Go, etc., 2.Meta Search engines e.g. Metacrawler, Clusty etc. and 3.Subject Directories 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 don't 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) [1] 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. Subject directories are popular due to proper classification of data in several categories. A larger 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. Manually classifying data is really expensive to scale as well as quite labor intensive and in this paper we present a technique to reduce manual effort significantly and hence in this paper we have analyzed an existing model for web page classification and have tried to develop a cost effective way for categorizing data.

In this project we use Naïve 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 used in our project has given accuracy around 94% for the custom created data set.

3.RELATED WORK

In this section we briefly try to review previous works in the field of text classification with a special emphasis on classification of web pages. In the starting days, task of classification of documents was generally carried out manually by experts of that domain. But very soon, ways were identified to carry out the classification in semi-automatic or automatic ways. Now we have a lot of effective ways to carry out machine assisted classification of documents. Many techniques have been researched and worked upon lately. Some of the approaches for text-categorization include statistical and machine leaning techniques like k-Nearest Neighbor approach [2][3][4], Bayesian probabilistic models [5][6], inductive rule learning [7], decision trees [8],[9], neural networks [10],[11] and support vector machines [12],[13]. While most of the learning methods have been applied to pure text documents, there are numerous approaches dealing with classification of web pages. Pierre [12] discusses various practical issues in automated categorization of web sites. Machine and statistical learning algorithms have also been applied for classification of web pages [13]-[17]. In order to exploit the hypertext based organization of the web page several techniques like building implicit links [17], removal of noisy hyperlinks[18], fusion of heterogeneous data[19], link and context analysis[20] and web summarization[21] are used. An effort has been made to classify web content based on hierarchical structure [22].

4.CLASSIFICATION OF WEB PAGES

Web page classification is different from normal text classification in some aspects. The uncontrolled nature of web content presents additional challenges to web page classification as compared to traditional text classification. The web content is semi structured and contains formatting information in form of HTML tags. A web page may contain hyperlinks to point to other pages. This interconnected nature of web pages provides features that can be of greater help in classification. In the first step of classification all HTML tags are removed from the web pages, including punctuation marks. The next step is to remove stop words as they are common to all documents and does not contribute much in classification. In most cases a stemming algorithm is applied to reduce words to their basic stem. One such frequently used stemmer is the Porter's stemming algorithm [23]. Each text document obtained by application of procedures discussed above is represented as frequency vector. Machine learning algorithms are then applied on such vectors for the purpose of training the respective classifier. Then this classifier is used on a test data set to predict the category to ensure that it has enough accuracy for working with previously unseen web pages. In order to rank high in search engine results, site promoters pump in many relevant keywords. This additional information can also be exploited.

5.BAYES ALGORITHM

a) Bayesian Classifiers

Bayesian classifiers are statistical classifiers which predict the class membership probabilities of tuples. It means probability of a given tuple to be in a particular class. Bayesian classifiers are based on Bayes theorem which will be explained in next section. 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[24].

b) Bayes Theorem

This theorem was named after Thomas Bayes who did early work in probability and decision theory during the 18^{th} century. Let X a data tuple. In Bayesian terminology, X is termed as 'evidence'. X is described by n attributes. ie, $X=\{x_1, x_2, x_3, x_4.....x_n\}$. Let H be some hypothesis that the data tuple X belongs to a particular class C. For classification problems we want to find P(H|X), the probability that hypothesis H holds when attributes set/'evidence' X is given. That means we are looking for the probability that the tuple X belongs to class C, given that we know the attribute description of X. Bayes theorem helps us to determine the probability P(H/X) in terms of P(H),P(X/H) and P(X).

Bayes Theorem is:

$$P(H/X)=\frac{P(X/H)P(C)}{P(X)} \quad \text{--(1)}$$

Where P(X/H) gives the probability for tuple to occur when it is given that the hypothesis holds. P (H) is the probability for the hypothesis H to hold. And P(X) is the probability for the attribute set X to occur.

c) Naive Bayes Classifier

Naive Bayes Classifiers are Bayesian Classifiers which assume class conditional independence. I.e., it assumes that the effect of an attribute value on a given class is independent of the values of the other attributes. This assumption simplifies the calculations a lot. Naive Bayesian classifier works as follows.

Let D be a training set of tuples and their associated classlabels. Each tuple is represented by n dimensional attribute vector in the form (x_1 ,x_2 ,x_3 ....x_n) depicting n measurements made on the tuple from n attributes, respectively, A_1 , A_2 ,..A_n

$$P(Ci/X)=\frac{P(X/Ci)P(Ci)}{P(X)} \quad \text{--(2)}$$

As P(X) will be constant for all classes, we have to consider numerator term only. If we are taking equal number of training tuples for each class, then P(C i) will be same for all classes. In such cases we have to maximize P(X/C i) only. Otherwise we have to maximize P(X/C i)*P(C i). Where P(C i) can be calculated as follows

$$P(Ci)=\frac{\text{Number of training documents in class Ci}}{\text{Total number of training documents}} \quad \text{--(3)}$$

When we have data sets with several number attributes, it is computationally expensive to calculated P(X/Ci). But when we assume class conditional independence, computational cost can be reduced significantly. With this assumption P(X/Ci) can be calculated as follows.

$$P(X/Ci)=\prod_{k=1} P(xk/Ci)=P(x1/Ci)P(x2/Ci)P(x3/Ci)...P(xn/Ci) \quad \text{--(4)}$$

If attribute A k is categorical, then P (x k /C i) is given as follows

$$P(xk/Ci)=\frac{\text{Number of tuples in Ci having value xk for attribute Ak}}{\text{Total number of tuples in class Ci}} \quad \text{--(5)}$$

If attribute x_k is continuous valued, then the following function can be used for finding P(x_k /Ci).

$$g(x,\mu,\sigma)=\frac{1}{\sqrt{(2\pi)}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad \text{--(6)}$$

so that,

$$P(xk/Ci)=g(xk,\mu Ci,\sigma Ci) \quad \text{--(7)}$$

To predict the class label of X, P(X/C i)P(C i) is evaluated for each class C i . The classifier predicts that the class label of tuple X is the class C i if and only if P(C i /X) > P(C j /X) for 1 <=j <= m and i!=j.

d) Modification on Naive Bayes for document classification

For document classification problems calculation of P(x_k/Ci) will be modified. It is required since attributes are not either categorical or continuous valued in case of document classification problems.

In document classification problem we have the relation,

$$P(X/Ci) \propto P(Ci) \prod_{1<k<nd} P(Tk/Ci) \quad \text{--(8)}$$

Where n_d is number of tuples in class C_i. P (t_k/C_i) can be calculated as follows:

$$P(Tk/Ci) = \frac{Tci}{\sum_{t' \in V} Tct'} \quad \text{--(9)}$$

Where T_{ci} is number of occurrences of word T_k in category C_i. V is the vocabulary, all words in training set.

e) Laplacian correction:

To avoid zero computational value In order to avoid the zero probability value , when a word in testing tuple do not come in training set, we use Laplacian correction. Laplacian corrected formula is Where V is number of terms in vocabulary.

$$P(tk/Ci) = \frac{Tci+1}{\sum_{t' \in V} (Tct'+1)} = \frac{Tci+1}{\sum_{t' \in V} Tct' + |V|} \quad \text{--(10)}$$

6.Experimental Setup

This section explains the setup of the entire experiment. We collected different web pages belonging to predefined categories. Then popular data cleaning and data extraction techniques were applied to extract useful data from collection of webpages. Once the data extraction process is finished we trained our classifier and used k-fold strategy and random sub sampling method to decide the accuracy of classifier. All the processes are briefly discussed below.

a) Creation of Data Set

The data set is collection of webpages within predefined categories. There were 6 predefined categories. Number of document in each category, number documents used for train set and number of document used for test set is given in table 1. Most of the documents were taken from popular news networks like BBC, CNN, and Reuters since they are already classified into various categories. We designed a crawler to automatically scan these web documents and to store these documents into different local directories corresponding to their category. In this experiment we have only considered the data in English language. Since we were only interested in the text content we simply removed all multimedia contents on the web page like Videos, Images, Flash plugin content and JAVA applets. Other than these, pages with relatively less content were also removed from the collection. The dataset consisted of total of 3183 documents in six categories.

b) Cleaning HTML Documents

We used the Python3 regular expression module and BeautifulSoup module to extract the information from web pages. Beautiful Soup is a Python library designed for easy and effective scraping of webpages and provides a few simple

methods and Pythonic idioms for navigating, searching, and modifying a parse tree [24]. Rather than extracting information from specific tags we removed all the HTML tags present in the document with the exception of title tag because it offers considerable amount of information regarding the category of webpage.

Table 1

Number of training and testing documents

Category	Total	Train	Test
Education	529	432	97
Entertainment	537	429	108
Politics	508	406	102
Religion	536	428	108
Sports	530	424	106
Technology	543	434	109
Total	**3183**	**2553**	**630**

All non ASCII characters, special symbols and numbers were removed from the data. CSS and JavaScript content present in the HTML documents were also cleaned during this phase. In many webpages images were used as in place of hyperlinks or images were used to display some important information. Such information is a very important feature for classification purpose, however our experiment concentrates on text based retrieval so such graphical contents were ignored. The standard Stop-Word list used in Bow [27] was used along with a comprehensive Stop-word list provided In the NLTK[28] (Natural language processing

toolkit) library in python. Along this we used the Porter stemming[25] technique to reduce words to their least forms. Porter's stemmer provided in NLTK library was used for stemming purpose. After all the above processing the HTML document was converted into a list of words in their most basic forms. All words having length less than 3 were removed. Further these words were converted into lowercase and then they were stored in a hash table along with their respective frequencies. All words having frequency less than 2 were ignored.The whole process of web page cleaning can be described by following algorithm.

Algorithm 1

Algorithm for getting data out of a web document in required form

INPUT:

filename->name of input file

OUTPUT:

freq-> Hash table containing frequency list of relevant words in given web page.

```
Function preprocess(filename):
   data=content_of("filename")
   pure_text=remove_html_tags(data)
   text=replace_nonascii_digitis_with_space(pure_text)
   text=replace_special_symbols_with_space(text)
   words=split_words_by_space(text)
   words=convert_to_lower(words)
   words=remove_stop_words(words)
   words=remove_words_with_length_lessthan3(words)
   freq=count_freq_of_words(words)

   return freq
```

c) Vocabulary Generation

Following algorithm was used for vocabulary generation

Algorithm 2
Algorithm for vocabulary generation

INPUT:
train_set->set of training documents in the format {name,category}
categories->A list containing available categories
OUTPUT:
database-> a 2d hash array listing frequency of each word in each category

```
Function vocabulary_generation(train_set)

  database={}  // define database as a hash function

  for each category in categories:
     database[category]={} //define each item in database
                           //as hash function

  for each document in train_set:
    freq=preprocess(document.name)
    for each word in freq:
      if word in database[document.category]:
        database[document.category][word]+=freq[word]
      else:
         database[document.category][word]=freq[word]

  return database
```

The words that occurred commonly in most of the webpages were considered as stop words. A list of such words is present in Table2.

Table 2
Web–page specific stop words

Login mail home website webpage web online search designed developed copyright rights reserved click welcome email contact page feedback webmaster help support

These common words were considered as Stop-Words and removed directly from the document. It was also observed that webmasters inflated the title,

Meta description and keyword tags with multiple keywords. We normalized such repeating keywords to reduce the impact of site promotion techniques applied by webmasters. This step was performed during the cleaning phase. All the words with the occurrence less than two were removed from the Documents and also the words with length less than three were also discarded from the Data. The frequency of each of the words was stored in a Hash table along with their category, for faster and efficient access in the later stages of experiment. The keywords that appear in two sample categories are given in table 3 and table 4

Table 3 **Key words Related to Education**
student university offer alumni degree program calendar study experience graduate state academics admission business college campus sport education inform library common event institute watch aid recreate history faculty graham athlete finance school center Wheaton art census online office form student day scholarship

Table 4 **Keywords Related to Religion**
god worship religion Hinduism prayer will Muslim symbol john people good practice order source church refer pope power influence belief include life faith doctrine religion follow century origin great scripture book Christian ecclesiast reform history state universe yantra India Iranian Buddhism Zoroastrian adhere culture tradition pupil jump Indian Islam Abraham create retrieve Zoroaster Mazda text evil

d) Testing and Training the Classifier

For testing the efficiency of classifier, we used two strategies namely k-fold cross validation strategy and random sub sampling method. We used k fold strategy for the values of k ranging from 2 to 10. In this method documents in each category will be divided into k equal partitions say D_1, D_2, D_3... D_k. Training and testing will be performed k times. In iteration i, partition Di in each category will be reserved as a testing set, and the remaining partitions will be collectively

used to train the model. That is, in the first iteration, subsets D_2... D_k 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_3... D_k and will be tested on D_2. And so on. In random sub sampling method we randomly pick k number of documents as training documents and remaining documents will be used for testing. And this procedure will be repeated n times. Overall accuracy will be average of accuracy values obtained each time.

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[26][31]:

$$\mathbf{P(w_k/C) = (n_k+1) / (n + |Vocabulary|)} \quad \text{--(11)}$$

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 (order of 2^{1023}) . We were able to find a solution for this problem by taking the initial probability as large number say 10^{2000} . Here we can assign such a large value to probability because, we are considering relative probability not absolute one. So category of a document will be decided as follows

$$\mathbf{C=Ci \text{ where } P(Ci/X) \text{ is maximum}} \quad \text{--(12)}$$

The calculated values were compared to find the maximum of all. The category with maximum relative probability was selected as the class for a document. The equation 12 helps to clearly understand the solution of the mentioned problem. The Naïve Bayes algorithm for deciding the category of a webpage is given as follows:

Algorithm 3
Algorithm for webpage classification

INPUT
freq-> word frequency list of test web page.
Database-> 2d hash table containing list of word frequencies in each category
categories->List of available categories for classification
OUTPUT
category->category of given web page

```
probability_model(freq_list,database):
    v=total_number_words_in_database
    pc={}

    for category in categories:
        attributes=database[category]
        n=total_number_of_words_in_category

        pc[category]=MAX_VAL
        for word in freq_list:
            if word not in attributes:
                pc[category]=pc[category]*(1.0/(n+v))
            else:
                pc[category]=pc[category] *
                                ((1+attributes[word])/(n+v))
    Category=category for which pc[category] is maximum
    return Category
```

The result obtained from both k-fold cross validation strategy and random sub sampling method was plotted in graph format using python module pylab which is defined in matplotlib library[29].

7.Experimental Results

The classification task was performed for the given dataset with 3183 examples in six categories. In k-fold cross validation strategy we used the value of k ranging from 2 to 10. The result obtained in this phase is depicted in Figure 1.

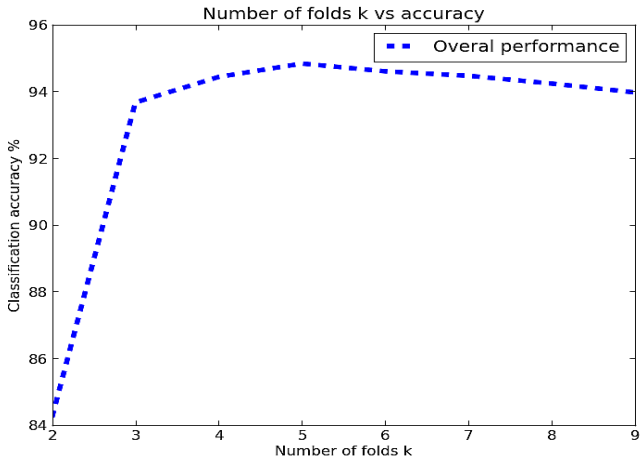


Figure 1

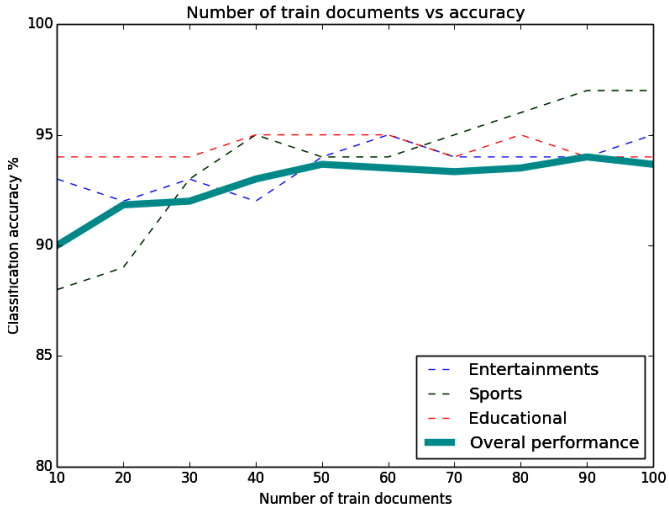
Average accuracy vs. number of folds

Initially there is a rise in classification accuracy with increase in number of folds which can be explained in terms of better training. Then the accuracy remains almost as a constant which is because of some inherent limitation of classifier. It is also observed that the classifier was able to achieve maximum accuracy which is above 94% for K = 5, i.e. when $4/5^{th}$ of the documents were used as training set and rest for test set.

The table 5 depicts confusion matrix obtained when k=5, ie when classifier give highest value of accuracy. 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.

Table 5 Horizontal category labels–Predicted category Vertical category labels–Actual category						
	P	**En**	**S**	**Ed**	**R**	**T**
P	470	5	10	0	14	1
En	7	481	6	0	6	0
S	11	3	485	0	1	0
Ed	17	0	0	475	6	2
R	1	0	0	0	499	0
T	64	0	0	1	0	235
P: Politics**, En:** Entertainment**, S:** sports **Ed:** Educational **R:** Religion **T:** Technology						

In random sub sampling method, we used values of k in the range 10 to 100 with an interval of ten. For each value of k, sampling was performed 5 times and the average accuracy is considered. The result obtained after performing random sub sampling method is depicted in Graph 2.



Graph 2

Number of training documents vs accuracy

note: some of the curves has removed for the sake of simplicity.

In this graph also, initially there is a rise in classification accuracy with increase in number of train documents. Then it become a almost constant around an accuracy of 94%. This trend is because of some inherent limitations of classifier. Maximum accuracy was obtained when k=90, ie 94.57%.

8.Area of Applications

Proper classification of web sites has numerous number of application in day to day life.

- Classification of web sites allow to make web directory projects (services that allow us to browse through different categories) automatic and to make the process faster and cheaper. This will help search engines to provide more relevant piece of results in response to user searches.
- Web site/page classification allow machine learning systems to collect necessary information from internet in a more easy way. For example if a system want to collect all news regarding stock exchange,then it can make use web page classification system.
- Younger generation may loss the rhythm/order of their life because of technology on finger tip. Classification of web sites will allow to prevent youth/children from misuse of technology. Some of the morally progressed nations like modern Turkey impart strict restrictions on internet usage to save youth from moral crazes.

9.Conclusion and Future Scopes

The probabilistic model of web page classification method based on naive bayes theorem used in this paper exploits effectiveness of textual content of a web page in deciding it's category. The algorithm used in our experiment classify webpages into a very broad set of categories. And we were able to achieve a classification accuracy around 94.5% after proper pre processing of data. It is also observed that the classification accuracy is proportional to number of documents used for training the classifier upto a certain limit. The results are quite encouraging. This approach can be utilized by the search engines and other online directory projects like DMOZ[32], Yahoo Directory for effective categorization of web pages to build a web directory with reduced human effort. This method can also be used in automatic crawlers for collecting web pages related to a particular category without human intervention.

10.References

[1] http://www.sciencedaily.com/releases/2013/05/130522085217.htm

[2] http://www.worldwidewebsize.com/

[3] http://www.thecultureist.com/2013/05/09/how-many-people-use-the-internet-more-than-2-billion-infographic/

[4] Automated Classification of Web Sites using Naive Bayesina Algorithm. Ajay S. Patil and B.V Pawar IMECS 2012.

[5] A decision-tree-based symbolic rule induction system for text categorization. DE. Johnson, FJ Oles, T Zhang, T Goets 2002

[6] Automated learning of decision rules for text categorization. Chidanand and Fred Damerau ACM 1994

[7] A statistical learning model of Text classification for Support Vector Machines. Thorsten Joachims ACM 2001

[8] Text categorization with support vector machines: Learning with many relevant features. Thorsten Joachims

[9] Support vector machines for text categorization. A. Basu, C. Watters, and M. Shepherd IEEE 2002

[10] Automated text classification using a dynamic artificial neural network model M. Ghiassi,M. Olschimke, B.Moon,P. Arnaudo Elsevier 2012

[11] Study a text classification method based on neural network model Jian Chen,Hailan Pan, Qinyum Ao Springer 2012

[12] Autmatic text classification using artificial neural network. Springer volume 172 2005

[13] Naive Bayes text classificaton - http://nlp.stanford.edu/IR-book/html/htmledition/naive-bayes-text-classification-1.html

[14] Naive Bayesian text classification - John Graham-cuming 2005

[15] A survey of text classification algorithms – Charu C Aggarwal, ChengXiang Zhai

[16] An improved K-nearest neighbour algorithm for text categorization. Li Baoli, Yu Shiven, Lu Qin ICCPOL 2003

[17] Recent research in web page classification-A review . Alamelu Mangai, Santhosh Kumar, Sugumaran IJCET 2010

[18] Data Mining: Practical machine learning tools and techniques, Ian H. Witten and Eibe Frank 2nd Edition, Morgan Kaufmann, San Francisco, 2005.

[19] Fast categorizations of large document collections Shanks, V. and H. E. Williams. SPIRE 2001

[20] Simple and accurate feature selection for hierarchical categorization. Wibowo, W. and H. E.Williams. ACM 2002

[21] Computational Approaches to Analyzing Weblogs- Mihalcea, R. and H. Liu. A corpus-based approach to finding happiness. In N.

Nicolov, F. Salvetti, M. Liberman, and J. H. Martin (Eds.) AAAI 2006

[22] Graph-based text classification: Learn from your neighbors Angelova, R. and G. Weikum SIGIR 2006

[23] S. Dumais and H. Chen, "Hierarchical classification of web content", in Proc. of the 23rd annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, Athens, Greece, 2000, pp. 256 - 263. [23] M.F. Porter, "An algorithm for suffix stripping", Program, Vo.14, no. 3, pp. 130-137, Jul. 1980.

[24] Python beautiful soup library.

http://www.crummy.com/software/BeautifulSoup/

[25] http://tartarus.org/martin/PorterStemmer/index.html Porter Stemming algorithm, with various implementations.

[26] http://nlp.stanford.edu/IR-book/html/htmledition/naive-bayes-text-classification-1.html Stanford NLP , naïve bayes text classification

[27] The BOW or libbow C Library [Online].

Available: http://www.cs.cmu.edu/~mccallum/bow/

[28] Bird, Steven, Edward Loper and Ewan Klein (2009). Natural Language Processing with Python. O'Reilly Media Inc. Python NLTK

[29] www.matplotlib.org Python based open source mathematical analysis toolkit.

[30] T. M Mitchell. (1997). Machine Learning McGraw-Hill Companies, Inc.

[31] Data Mining concepts and techniques Han Kamber Lee Morgan Kaufman publications. 3rd Edition 2012.

[32] DMOZ open directory project. [Online]. Available: http://dmoz.org/