

**A Framework for Document Classification  
using Machine Learning Techniques.**

*A Dissertation submitted to the  
School of Computer & Systems Sciences,  
Jawaharlal Nehru University, New Delhi  
in partial fulfillment of the requirements for the award of the degree of*

**MASTER OF TECHNOLOGY  
IN  
COMPUTER SCIENCE AND TECHNOLOGY**

**BY  
KONA SRINIVAS**

**UNDER SUPERVISION OF  
Dr. Aditi Sharan**



**SCHOOL OF COMPUTER AND SYSTEMS SCIENCES  
JAWAHARLAL NEHRU UNIVERSITY  
NEW DELHI-110067, INDIA  
JULY 2011**



# जवाहरलाल नॅहरू विश्वविद्यालय

JAWAHARLAL NEHRU UNIVERSITY

School of Computer & Systems Sciences

NEW DELHI- 110067, INDIA

## DECLARATION

This is to certify that the dissertation entitled “**A FRAMEWORK FOR DOCUMENT CLASSIFICATION USING MACHINE LEARNING TECHNIQUES**”, which is being submitted to the School of Computer and Systems Sciences, **Jawaharlal Nehru University**, New Delhi, in partial fulfillment of the requirements for the award of the degree of **Master of Technology in Computer Science and Technology**, is a bonafide work carried out by me under the supervision of **Dr. Aditi Sharan**

The matter embodied in this dissertation has not been submitted to any other University or Institution for the award of any other degree or diploma.

*K. Srinivas  
15/7/2011*

**KONA SRINIVAS**

M.TECH-CS

SC&SS, JNU,

New Delhi-110067



# जवाहरलाल नॅहरू विश्वविद्यालय

JAWAHARLAL NEHRU UNIVERSITY


School of Computer & Systems Sciences

NEW DELHI- 110067, INDIA

## CERTIFICATE


This is to certify that the dissertation entitled “**A FRAMEWORK FOR DOCUMENT CLASSIFICATION USING MACHINE LEARNING TECHNIQUES**” being submitted by **Kona Srinivas** to the School of Computer and Systems Sciences, **Jawaharlal Nehru University**, New Delhi, in partial fulfillment of the requirements for the award of the degree of **Master of Technology in Computer Science and Technology**, is a record of bonafide work carried out by him under the supervision of **Dr. Aditi Sharan**.

This work has not been submitted in part or full to any university or institution for the award of any degree or diploma.

  
15-7-2011  
Dr. Aditi Sharan.

(Supervisor)

SC&SS, JNU, New Delhi

  
(Dean, SC&SS,)

JNU, New Delhi

प्रोफेसर कर्मेशु/Professor Karmeshu  
डीन/Dean  
स्कूल ऑफ कंप्यूटर आर सिस्टम विज्ञान संस्थान  
School of Computer and Systems Sciences  
जवाहरलाल नॅहरू विश्वविद्यालय  
Jawaharlal Nehru University  
नई दिल्ली / New Delhi-110057

## ACKNOWLEDGEMENT

---

I would like to gratefully acknowledge the enthusiastic supervision of **Dr. Aditi Sharan** during this work. This work wouldn't have been possible without her constant support, valuable suggestions and comments during my whole tenure of this dissertation work. I feel privileged to work under her for my Master's dissertation. Apart from the academic guidance she has always been a great mentor of mine in encouraging me to be disciplined and well organized. I must surely say, she has given her best in providing me the infrastructure required, which led to the successful completion of my dissertation. I would take this opportunity to thank her once again for her esteemed support and I, from the bottom of my heart would like to wish her the best in all her future endeavors.

I wish to thank my colleagues Mr. Sifatullah siddiqi, Ms. Aparna Kumari and Ms. Vajenti Mala for creating a home-like environment in our lab to keep the stress away. I would also like to thank my best critics Nidhi, Shina panicker and Vineet Kumar dubey for making my dissertation an error free record of my work. Thank you guys !!

Finally I would like to thank the whole academic staff of our department for clarifying my doubts throughout this work and last but not least, the JNU administration for creating such a secular and healthy environment amongst the students.

Love you, Mom and Dad for making me feel that "I am the most luckiest kid in this world".

## ABSTRACT

Document Classification has been investigated for use in Text Mining and Information Retrieval Systems. In this era of Digitization, most of the documents are stored in the digital format. Storing of documents should be done in such a way that it can be easily retrieved while needed. This need has facilitated the importance of organizing the documents based on the content of the document. Document classification plays an important role in optimizing the time needed to retrieve the relevant document from the abundant Document Collection. This Dissertation addresses the issue of Document Classification.

To classify the documents into various categories, we can apply machine learning algorithms. In order to apply Machine Learning, Text data must be converted into a numerical format compatible with the algorithm. In this work, we first explain the preprocessing techniques required to be applied on the text data to convert the documents into a numerical format (VSM). During preprocessing, we have discussed the various phases such as Stopword Removal, Stemming, etc. in detail. We have then made an experiment on Multi-Class Classification. For this we have used a Balanced One-Versus-All SVM approach on the obtained VSM for classifying the documents. We have analyzed the behavior of the Multi-Class Classification using both an unbalanced and balanced Training Dataset. Taking the Classification percentage as the empirical measure , we have evaluated the performance.

## TABLE OF CONTENTS

Certificate.....	i
Acknowledgement.....	ii
Abstract.....	iii

### 1.Introduction.....Page No.

1.1 Introduction to Document Classification.....	1
1.2 Applications of Document Classification.....	1
1.3 Types of Document Classification.....	2
1.3.1 Document Categorization	
1.3.2 Document Clustering	
1.4 Categories of Automatic Classifiers.....	3
1.4.1 Rule Based Classifiers	
1.4.2 Linear Classifiers	
1.4.3 Example Based Classifiers	
1.5 Standard Document Classification problem.....	4
1.6 Challenges in Implementing Document Classification.....	5

### 2. Vector Space Model for Representing Documents

2.1 Introduction to Document Preprocessing.....	6
2.2 Stop-word Removal.....	6
2.2.1 Procedure to generate Stop-word List.	
2.3 Stemming.....	8
2.3.1 Categories of Stemming Algorithms	
2.3.2 Modification of Porters Rules	

2.4	VSM Representation.....	16
2.4.1	Local Factors	
2.4.2	Global Factors	
2.4.3	Normalization Factors	
3.	Machine Learning Techniques for Multi-Class Classification.	
3.1	Introduction to Machine Learning Algorithms.....	18
3.2	Binary Classification Vs Multi-Class Classification.....	18
3.3	Strategies for Multi-Class Classification.....	18
3.3.1	One-versus –all (OVA)	
3.3.2	All-versus-all (AVA)	
3.3.3	Error-Correcting Output-Coding	
3.3.4	Generalized Coding	
3.3.5	Hierarchical Classification	
3.4	Important Machine learning techniques.....	21
3.4.1	Decision Trees	
3.4.2	Decision Rules	
3.4.3	k-Nearest Neighbor	
3.4.4	Vector-based Methods	
3.5	Relevance of SVM w.r.t Document Classification.....	23
4.	Proposed Algorithm and Experimental Results.	
4.1	Dataset and Software requirements.....	24
4.2	Proposed Algorithm.....	26
4.3	Results.....	28

4.4 Analysis.....	29
5. Conclusion.....	30
References.....	31



# **Chapter-1**

## **Introduction**

### **1.1 Introduction to Document Classification:**

The basic task of document Classification is to assign an electronic document to one or more categories depending upon its content. The objective of document classification is to reduce the detail and diversity of data and the resulting information overload by grouping similar documents together.

### **Need For Document Classification:**

The trend to store the document in a digital format has become popular since it can be easily maintained. Storing the information can be done in two ways. The first approach is by printing them and organizing them into folders. The second approach is storing the information in the digital format itself, in an organized manner, which is widely used these days. In this era of digitization, every information is stored in the form of digital documents itself. So, the process of searching a relevant document from this abundant Information is quite a tedious task. So, storing of data in a digital format has created the need for a convenient way of retrieval. It makes no sense if we cannot find our document easily when we store it in a digital format. This ease to retrieve the document from a huge set of documents has lead to the need for organizing the documents in a proper way. This necessity has lead to the evolution of “DOCUMENT CLASSIFICATION”( Brucher,2002).

### **1.2 Applications of Document Classification:**

- Email Filtering
- Mail Routing
- News Monitoring
- Narrowcasting

- Content Classification

### **1.3 Types of Document Classification**

The term document Classification is often used to subsume two (Manu,2006) types of analyses.

- Document Categorization
- Document Clustering

#### **1.3.1 Document Categorization:**

Document Categorization is a supervised learning approach, in which we have a predefined classes or categories. The basic task of this approach is to assign a new document to one of the already existing predefined class.

Typically every categorization problem is undergone through two phases namely Training phase and testing phase.

**Training Phase:** In this phase class profiles are learned from sufficiently large sample of documents by training the classification algorithm using a training data set.

**Testing Phase:** In this phase class profiles are applied to the training data set in order to categorize them into the predefined classes generated by the training phase.

#### Rules in developing a Categorization Model:

- Mark the labeled data as Training set.
- Create a classifier function for each predefined class
- Apply this classifier function on the test data to check the accuracy of classifier.
- Apply the classifier for categorizing the new dataset.

#### **1.3.2 Document Clustering:**

Document Clustering is an unsupervised approach in which the data is unlabeled. The major task in this problem is to identify the similarities between the documents and to create the clusters as per the similarity measure. In this approach, a training phase may or may not be present.

### Rules in developing a Document Clustering Model:

- Create the clusters as per the similarity measures
- Deduce a function for each cluster
- Classify the new document to one of the above created clusters.

Thus a Document Categorization can be viewed as a Supervised Learning approach, where as a Document Clustering can be viewed as an Unsupervised Learning approach. Since we are basically dealing with classifying the documents into the predefined classes, we would generally use the term “Document Classification” to represent “Document Categorization” in the rest of our work.

### **1.4 Categories of Automatic Classifiers :**

Automatic Classifiers are basically divided into three (Christopher,2009) broad categories.

- Rule Based Categories
- Linear Classifiers
- Example Based Classifiers

#### **1.4.1 Rule Based Classifiers:**

These classifiers learn by inferring a set of rules from pre-classified documents. The decision rules may take the form of decision trees. Other algorithms come from work on Theorem Proving, and may be based on propositional logic or first and even second order logic.

#### **1.4.2 Linear Classifiers:**

In these algorithms, for each class a class profile is computed, a vector of weights, one for each feature, based on occurrence frequency and probabilistic reasoning. For each class and document, a score is obtained by taking an in-product of class profile and document profile. This class contains various adaptations of IR's TF.IDF weighting of terms to the learning situation, known as Rocchio's algorithm. The Naive (or Simple) Bayesian classification is based on the estimation of conditional probabilities. The Vector Support Machines compute an optimal linear classifier using a transformation of the feature space. This class furthermore comprises some heuristical learning algorithms from AI, like

the Perceptron, in which the weights are obtained in a somewhat more adventurous way in the course of the learning process. Some examples are the Sleeping Experts algorithm and the Winnow algorithm.

#### **1.4.3 Example based Classifiers:**

These classifiers classify a new document by finding the k documents nearest to it in the train set and doing some form of majority voting on the classes of these nearest neighbors.

#### **1.5 Standard Document Classification problem:**

Input: Text Documents

Output: Class labels.

To classify the documents into classes, we need to have a training set which is a labeled data. This labeled data is then used as a training set to deduce the class profiles such as classifier functions for each class. Then a test data will be used on these classifier functions to decide the class to which the document may belong to.

The basic problems in dealing with Classifying Text documents are

- 1) Most of the machine Learning methods basically work on the Structured data and since the text documents is in unstructured manner, we need to convert the document to a structured manner.
- 2) In text documents, features are generally words and since a large number of words form a document there is a high chance that the dimension of the attributes in converting it into a structured format is huge.
- 3) In order to reduce the dimensionality of the feature space, we need to preprocess the documents before making it structured.
- 4) Feature Extraction and Feature Selection techniques are then applied to reduce the feature space and to select the minimal features that best represents the documents.
- 5) A Vector Space Model (VSM), in which the documents representing the rows and words (features) representing the columns is generated to make the text collection available in the structured format.

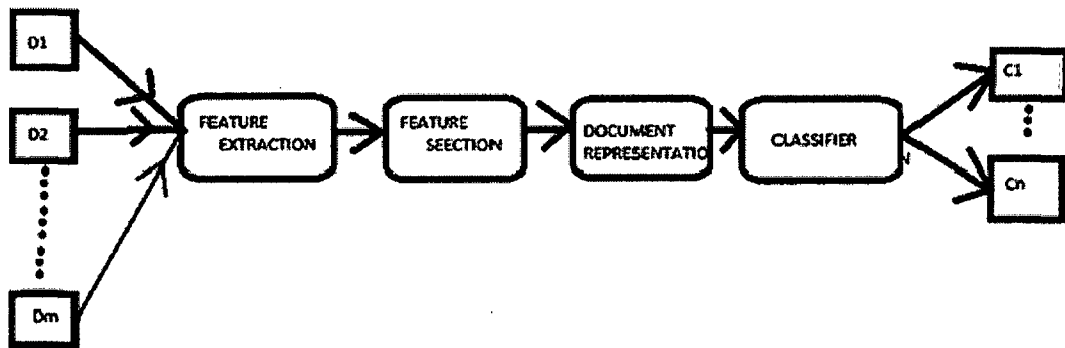


Fig.1. Flow diagram of Standard Document Classification problem.

### 1.6 Challenges in Implementing Document Classification:

- Extract the features from the whole document collection
- Select the relevant features(words) which represent the document
- Generate a VSM for the document collection
- Convert the VSM to a format compatible with the classification Algorithm
- Apply the Classifier on the training set to create the class profile
- Test the classifier with the testing set
- Evaluate the performance of the classifier

## **Chapter-2**

### **Vector Space Model for representing Documents.**

#### **2.1 Introduction to “Document Preprocessing”**

Documents are generally stored in the text format. A machine learning or data mining Algorithm is generally designed to work on the numerical data. So, to make the text data compatible with the above said algorithms, Preprocessing of data is required in such a way that the minimal features represent the majority of the text content. The major phases involved in preprocessing the documents so as to make it compatible with the Machine Learning Algorithms are

- Removing Stopwords
- Stemming
- VSM Representation.

#### **2.2 Stop-word Removal :**

Stop words is the name given to words which are filtered out prior to, or after, processing of natural language data (text). Indexing of stop words doesn't add any significance to the document. Currently a list of around 430 words is listed as stop words list in English which is iteratively updated after working-out with many different Text Corpora.

##### **2.2.1 Procedure to generate Stopword list:**

A Stop word list is created using the below procedure (Fox,1990):

- Select any large Corpus which contains around 1 million words\*(not all different).
- Set the frequency of occurrence “T” as the threshold to categorize it as a stop word.
- Some words of very high impedance may also occur more than threshold limit. So, manually scan the obtained list and undo those words from the stop word list.
- Finally, A partial Stop word list is prepared.

To make the partial Stop word list Complete, we need to update the list as per some rules and **criteria which are mentioned below.**

- Add all letters for which a word already in the list starts with the letter.
- Add any traditional stop word occurring at least 100 times that differs from a word already in list only in a single letter.
- Add words with the same prefix ending in "body," "one," "thing," or "where," provided at least one of these words, or the prefix, already appears in the list. Also add the prefix, if it is a word. For example, “no one”, “nobody”, “nowhere” are added as “nothing” and “no” are already in the list.
- Add words with the same prefix ending in "ed," "ing," or "s," provided at least one of these words, or the prefix, already appears in the list. Also add the prefix, if it is a word. For example, since "asked" appears in the list, the words "ask," "asking," and "asks" were added as well.
- Add words with the same prefix ending in "er" or "est," provided at least one of these words, or the prefix, already appears in the list. Also add the prefix, if it is a word.
- If a word in the list can take the suffix "ly," “self”, “s”, then add the suffixed word.
- Add proper prefixes of words appearing in the list.

Thus a Stop word list is generated. Moreover this is never complete. This list keeps on updating when we conduct experiments with different Corpora. A stop word list, generated by the above algorithm containing 421 words is widely accepted irrespective of the Corpus, and this list can be updated depending upon the corpus used for conducting the Document classification. Every corpus would have some words specifically used for maintaining the syntax of the data. So, these corpus specific stop words can be added to the list of 421 words to generate the required stop word list.

## 2.3 Stemming

**STEM** is the root form of a word.

Example: “**EAT**” is the stem of ate,eaten,eating.

The ability of an Information Retrieval (IR) System to conflate words allows reducing index and enhancing recall sometimes even without significant deterioration of precision. Conflation also conforms to user’s intuition because users don’t need to worry about the proper morphological form of words in a query. The problem of automated conflation(Ricardo,1990) implementation is known as “Stemming” and the algorithms used to solve this problem are widely known as Stemming Algorithms.

### 2.3.1 Categories Of Stemming Algorithms

- Brute-Force Algorithms
- Affix Removal Techniques
- Statistical Algorithms.

#### **Brute-Force Algorithms:**

Brute force stemmers (Honrado,2000) employ a lookup table which contains relations between root forms and inflected forms. To stem a word, the table is queried to find a matching inflection. If a matching Inflection is found, the associated root form is returned. To prepare this look up table, we employ a production technique which will generate all the possible inflected forms for a word encountered. Then this lookup table can be used for implementing BRUTE-FORCE ALGORITHMS.

#### Advantages:

Overcomes the challenges faced by other Stemmers when the inflected forms are just not formed by suffixing/prefixing the stem word



### Disadvantages:

- 1) Works efficiently only if the inflected form is available in the look up table.
- 2) Consumes lot of time in searching.

### Affix Removal Techniques:

#### **Trivial Algorithm:**

Simplest Algorithm is S-Stemmer, in which singulars and plural forms of a noun are conflated by using the below rules.

IF a word ends in "ies", but not "eies" or "aies"

THEN "ies" -> "y"

ELSE IF a word ends in "es", but not "aes", "ees" or "oes"

THEN "es" -> "e"

ELSE IF a word ends in "s", but not "us" or "ss"

THEN "s" -> NULL

#### **Lovins Algorithm:**

It defines 294 endings, each linked to one of 29 conditions, plus transformation rules. These rules are repeatedly applied to obtain the stem.

Example: sitting → sitt → sit. But this algorithm missed certain endings.

#### **Paice/Husk Algorithm:**

It is an iterative algorithm with one table containing about 120 rules indexed by the last letter of a suffix. On each iteration, it tries to find an applicable rule by the last character of the word. If there is no such rule, it terminates. It also terminates if a word starts with a vowel and there are only two letters left or if a word starts with a consonant and there are only three characters left. But it has the tendency to over stem.

### **Dawson Algorithm:**

Dawson algorithm can be considered as an improvement of the Lovins approach. It follows the same longest match process and has perhaps the most comprehensive list of English suffixes (along with transformation rules) – about 1200 entries. The suffixes are stored in the reversed order indexed by their length and last letter. The rules define if a suffix found can be removed (for example, if the remaining part of the word is not shorter than N symbols; or if the suffix is preceded by a particular sequence of characters). It seems that the algorithm didn't gain popularity due to its complexity and lack of a standard reusable implementation.

### **Porters Algorithm and Snowball Framework**

Porter algorithm defines five successively applied steps of word transformation.

Each step consists of set of rules in the form

<condition> <suffix> -> <new suffix>.

For example, a rule (m>0) EED -> EE means “if the word has at least one vowel and consonant plus EED ending, change the ending to EE”.

So “agreed” becomes “agree” while “feed” remains unchanged. Porter's Algorithm is very concise (around 60 rules) and efficient in terms of computational complexity. So it has become the most popular approach for Stemming.

Dr. PORTER also developed a whole Stemmers framework called “SNOWBALL”. The main purpose of this project is to develop their own stemmers for other character sets and languages. Since Porters Algorithm has been observed as an efficient means to do stemming, we would like to emphasize on Porter's Stemming Algorithm in the rest of our work. There are five rules which should be followed sequentially in order to stem a word. These five rules are listed below.

1. Step 1a
  - ✓ SSES -> SS
  - ✓ IES -> I
  - ✓ SS -> SS
  - ✓ S ->

### Step 1b

- ✓ (m>0) EED -> EE
- ✓ (\*v\*) ED ->
- ✓ (\*v\*) ING ->

If the second or third of the rules in Step 1b is successful, the following is done:

- ✓ AT -> ATE
- ✓ BL -> BLE
- ✓ IZ -> IZE
- ✓ (\*d and not (\*L or \*S or \*Z) -> single letter
- ✓ (m=1 and \*o) -> E

### Step 1c

- ✓ (\*v\*) Y -> I

## 2. Step 2

- ✓ (m>0) ATIONAL -> ATE
- ✓ (m>0) TIONAL -> TION
- ✓ (m>0) ENCI -> ENCE
- ✓ (m>0) ANCI -> ANCE
- ✓ (m>0) IZER -> IZE
- ✓ (m>0) ABLI -> ABLE
- ✓ (m>0) ALLI -> AL
- ✓ (m>0) ENTLI -> ENT
- ✓ (m>0) ELI -> E
- ✓ (m>0) OUSLI -> OUS
- ✓ (m>0) IZATION -> IZE
- ✓ (m>0) ATION -> ATE
- ✓ (m>0) ATOR -> ATE
- ✓ (m>0) ALISM -> AL

- ✓ (m>0) IVENESS -> IVE
- ✓ (m>0) FULNESS -> FUL
- ✓ (m>0) OUSNESS -> OUS
- ✓ (m>0) ALITI -> AL
- ✓ (m>0) IVITI -> IVE
- ✓ (m>0) BILITI -> BLE

### 3. Step 3

- ✓ (m>0) ICATE -> IC
- ✓ (m>0) ATIVE ->
- ✓ (m>0) ALIZE -> AL
- ✓ (m>0) ICITI -> IC
- ✓ (m>0) ICAL -> IC
- ✓ (m>0) FUL ->
- ✓ (m>0) NESS ->

### 4. Step 4

- ✓ (m>1) AL ->
- ✓ (m>1) ANCE ->
- ✓ (m>1) ENCE ->
- ✓ (m>1) ER ->
- ✓ (m>1) IC ->
- ✓ (m>1) ABLE ->
- ✓ (m>1) IBLE ->
- ✓ (m>1) ANT ->
- ✓ (m>1) EMENT ->
- ✓ (m>1) MENT ->
- ✓ (m>1 and (\*S or \*T)) ION ->

- ✓ (m>1) OU ->
- ✓ (m>1) ISM ->
- ✓ (m>1) ATE ->
- ✓ (m>1) ITI ->
- ✓ (m>1) OUS ->
- ✓ (m>1) IVE ->
- ✓ (m>1) IZE ->

#### 5. Step 5a

- ✓ (m>1) E ->
- ✓ (m=1 and not \*o) E ->

#### Step 5b

- ✓ (m > 1 and \*d and \*L) -> single letter.

### 2.3.2 Modification of Porters rules

There are few limitations with the Porters Stemming Rules. One such limitation is observed for the words ending with the letter 'y'. They are generally stemmed into a form ending with the letter 'i' which in some instances is changing the meaning of the word and the importance of the feature is being degraded. In order to retain this importance an effort is done to modify those rules ending with the letter 'y'.

#### 1) Step 1

- ✓ IES -> y
- ✓ Eliminated step 1c

#### 2) Step 2

- ✓ (m>0) ENCY -> ENCE
- ✓ (m>0) ANCY -> ANCE
- ✓ (m>0) ABLI -> ABLE
- ✓ (m>0) ALLI -> AL
- ✓ (m>0) ENTLI -> ENT
- ✓ (m>0) ELI -> E

- ✓ (m>0) OUSLI -> OUS
- ✓ (m>0) ALITI -> AL
- ✓ (m>0) IVITI -> IVE
- ✓ (m>0) BILITI -> BLE

3) Step 3

- ✓ (m>0) ICITI -> IC

4) Step 4

- ✓ (m>1) ITI ->

**Analysis Of the Modification:**

The results of the two algorithms are listed below :

=====

Results of Existing Porters Algorithm :

=====

Number of documents = 30

Number of terms = 839

Average number of terms per document (before the normalization) = 156.667

Average number of indexing terms per document = 70.8333

Sparsity = 5.2642%

Removed 199 stopwords...

Removed 141 terms using the stemming algorithm...

Removed 34 terms using the term-length thresholds...

Removed 0 terms using the global thresholds...

Removed 0 elements using the local thresholds...

Removed 0 empty terms...

Removed 0 empty documents...

=====

Results of Modified Porters Algorithm :

=====

Number of documents = 30

Number of terms = 840

Average number of terms per document (before the normalization) = 156.667

Average number of indexing terms per document = 70.8333

Sparsity = 5.25%

Removed 199 stopwords...

Removed 140 terms using the stemming algorithm...

Removed 34 terms using the term-length thresholds...

Removed 0 terms using the global thresholds...

Removed 0 elements using the local thresholds...

Removed 0 empty terms...

Removed 0 empty documents...

The dictionary generated using the Modified Porter's Algorithm is close to the English vocabulary when compared to the existing porter's Algorithm. Although there is not much significant difference in the number of words removed due to stemming in both the algorithms, but the modified Algorithm can work efficiently when the synonyms are taken into consideration for stemming since the effort is done to retain its original meaning.

For example,

Existing Porters Algorithm		Modified Porters Algorithm	
Actual word	Stemmed to	Actual Word	Stemmed to
Buy	Bui	Buy	Buy
Company	Compani	Company	Company
Companies	Compani	companies	Company
Pay	Pai	Pay	Pay

## 2.4 VSM REPRESENTATION

The whole document collection is converted to a matrix format in which rows represent documents and columns represent Features(words). A numerical value is assigned to the cell corresponding to the particular row and column. The numerical values should be weighted

using a Proper Term Weighting Scheme (Zeimpekis,2005) . Proper Weighting Schemes can greatly affect the efficiency of document Classification. A Term weighting scheme is generally composed of three different weighting schemes i.e. Local, Global and Normalization. Term weight is given by

$$"L_{ij}G_iN_{ij}"$$

Where,

$L_{ij}$  is a local factor that measures the importance of term I in document j,

$G_i$  is a global factor that measures the importance of term I in the entire collection

$N_{ij}$  is a normalization factor.

#### **2.4.1 LOCAL FACTORS**

Term frequency	$f_{ij}$
Binary	$b(f_{ij})$
Logarithmic	$\log_2(1+ f_{ij})$
Alternate log	$b(f_{ij})(1+\log_2 f_{ij})$
Augmented normalized term frequency	$(b(f_{ij})+( f_{ij} / \max_k f_{kj})) / 2$

#### **2.4.2 GLOBAL FACTORS**

None	1
Entropy	$1+(\sum_j(p_{ij} \log_2(p_{ij})) / \log_2 n)$
Inverse document frequency (IDF)	$\log_2(n/\sum_j b(f_{ij}))$
GfIdf	$(\sum_j f_{ij})/(\sum_j b(f_{ij}))$
Normal	$1/((\sum_j f_{ij}^2)^{1/2})$
Probabilistic Inverse	$\log_2((n-\sum_j b(f_{ij})) / \sum_j b(f_{ij}))$



### 2.4.3 NORMALIZATION FACTORS

None	1
Cosine	$\sum_j (g_{ij})^2)^{-1/2}$

Selection of Optimal Local and Global Values affects the efficiency of the Indexing the documents. Experimental Studies revealed that Term Frequency(TF) and Inverse Document Frequency(IDF) are proved to be efficient. So, this Scheme can be used in our future work for Document Representation through Indices.

## **Chapter-3**

### **Machine Learning Algorithms for Multi-class Classification**

#### **3.1 Introduction to Machine Learning Algorithms**

A machine Learning Algorithm is an algorithm which automatically builds a classifier by learning the characteristics of the categories from a set of classified documents, and then uses the classifier to classify documents into predefined categories.

#### **3.2 Binary classification vs Multi-Class classification:**

##### **Binary classification:**

A binary classification problem basically involves the task of classifying the document into one of the two categories.

##### **Multi-Class Classification:**

In Multi Class Classification Problem, the number of classes is more than two. The basic task of Multi-Class Classification is to assign a document to one of the existing classes. This task is basically tedious when compared to the Binary classification.

#### **3.3 Strategies for Multi-Class Classification:**

There are a few strategies (Mohamed,2005) to solve Multi-Class Classification. They are

- One-Versus-All (OVA)
- All-Versus-All (AVA)
- Error-Correcting Output-Coding
- Generalized Coding
- Hierarchical Classification

### **3.3.1 One-Versus-All (OVA)**

In this Strategy of solving a Multi-Class Classification problem of “k” classes, the problem is decomposed into “k” binary classification problems, where each problem discriminates a given class from the other  $K-1$  classes. In this approach we need “k” binary classifiers, where  $k^{\text{th}}$  classifier is trained with positive examples belonging to class k and negative examples belonging to the other  $K - 1$  classes. While testing a new example whose class is unknown, the classifier producing the maximum output is assigned to the new example.

### **3.3.2 All-Versus-All (AVA)**

In this Strategy, each class is compared to each other class .A binary classifier is built to discriminate between each pair of classes, while discarding the rest of the classes. This requires building  $K(K-1)/2$  binary classifiers. Although the number of classifiers have increased, this approach has been more effective when compared to the OVA approach.

### **3.3.3. Error Correcting Output Coding (ECOC)**

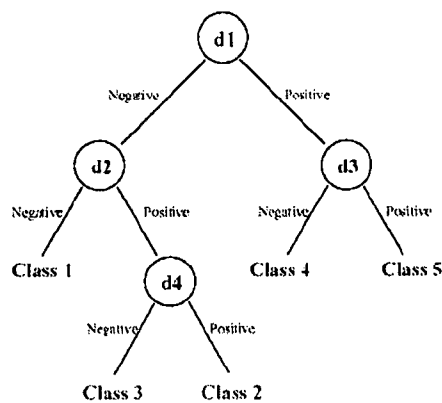
This approach incorporates the idea of error-correcting codes discussed above for neural networks . It works by training N binary classifiers to distinguish between the K different classes. Each class would be assigned a codeword of length N according to a binary matrix M. Each row of M corresponds to a certain class. Each class is given a row of the matrix. Each column is used to train a distinct binary classifier. When testing an unseen example, the output codeword from the N classifiers is compared to the given K codewords, and the one with the minimum hamming distance is considered the class label for that example. This approach has shown even better results when compared to the above two approaches.

### 3.3.4 Generalized Coding

This approach is a generalized approach of ECOC, where the coding matrix  $M$  can have any one of  $\{1,0,-1\}$  values. The value of  $+1$  in the entry  $M(k, n)$  means that examples belonging to class  $k$  are considered as positive examples to classifier  $n$ . A value of  $-1$  denotes that these examples are considered negative examples. A value of  $0$  instructs the classifier to ignore that class altogether. Clearly this scheme is the general case of the above three coding strategies. The OVA approach has a matrix  $M$  such that each column contains exactly one  $+1$  value with the rest filled with  $-1$ . The AVA scheme has columns with exactly one  $+1$  value and one  $-1$  value with the rest set to zero's. The ECOC has the matrix  $M$  filled with  $+1$  and  $-1$  values, when testing an unknown example, the codeword "closest" to the output corresponding to that example is chosen as the class label.

### 3.3.5 Hierarchical Classification

In Hierarchical Classification, a tree structure is formed by dividing the parent node into number of clusters, one for each child node. A simple binary classifier is implemented at each level to discriminate the sibling classes. If a binary Hierarchical Classification approach is used, the tree structure would resemble as shown below.



### 3.4 Important Machine Learning Techniques (Aurangzeb,2010) :

- **Decision Trees**
- **Decision Rules**
- **k-Nearest Neighbor**
- **Vector Based Methods**
  - **Centroid Algorithm**
  - **Support Vector Machines**

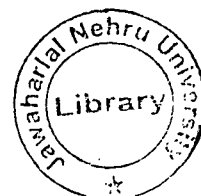
#### 3.4.1 Decision Trees:

A decision tree is generated by the training data by constructing well defined true/false-queries in the form of a tree structure. The nodes represents the questions and the leaves represent the classes. Once this structure is generated by using the training data ,the testing data is allowed to run through the qwery structure until it reaches a certain leaf. This Method is very easy to understand even for a person not familiar with the details of the model, but it has the serious disadvantage of over-fitting the training data.

#### 3.4.2 Decision Rules :

In this method, every category would have a certain rule set that best describes the category. Generally every rule would consist a category name and the dictionary corresponding to its features. Some heuristics are applied to minimize the rules for categories in order to obtain a reduced rule set per category, without effecting the categorization of the training documents. The basic advantage of this method is, we can create a local dictionaries during the feature extraction phase.(Example Bark when accompanied with dog would give a particular meaning whereas when it is accompanied with tree would give another meaning. These local dictionaries can be easily maintained.) The major disadvantage with this rule is ,it is impossible to assign a particular document exclusively to one category, because rules from different categories are applicable.

TH-20603



### **3.4.3 k-nearest Neighbor :**

k-nearest neighbor method doesn't need a training phase to categorize the documents, it categorizes on-the-fly. The categorization is performed by comparing the category frequencies of the k-nearest documents. The closeness can be calculated by measuring the angle between the feature vectors or by calculating the Euclidean distance between the feature vectors. The major disadvantage is its above average categorization time since it doesn't have any preliminary investment in learning phase.

### **3.4.4 Vector Based Approaches:**

There are two types of vector based approaches. Centroid Algorithm and Support vector machines. We will discuss in details about these two methods in the below section.

#### **Centroid Algorithm:**

An average vector called Centroid-Vector is calculated for each category at the learning phase. A new document is then categorized by finding the closest centroid vector to its feature vector. The basic disadvantage arises when the number of categories is very large and also when the clusters overlap.

#### **Support Vector Machines (SVM)**

SVM's take into consideration the positive as well as negative examples in training phase to generate a model. SVM then tries to create a decision surface which separates the positive examples from the negative examples. The document representatives which are closest to the decision surface are known as the support vectors. This model doesn't have any effect, if the data which are not support vectors are removed from the learning phase. Once a decision surface is established, this model is then applied to the testing data to categorize them into different categories. SVM generally deals with the 2-class problem, but this drawback can be easily overruled by considering the data in the required class as positive examples and the rest as negative examples.

### **3.5 Relevance of SVM w.r.t Document Classification:**

Document Classification is generally a multi-Class Classification problem, which has a large number of features. Taking into account the sparsity of the matrix (Document Vs Words), the documents can be best represented as vectors. SVM can be efficient when number of classes is more since we can easily eliminate the unnecessary classes by making them the negative examples, while generating a model in the training phase. Thus a Multi-Class SVM technique is used in my future work for categorizing the documents into classes.

## Chapter-4

### Proposed Algorithm and Experimental Results

The major objective of our algorithm is to improve the classification accuracy of the document collection. An experiment has been performed using the Balanced-OVA-SVM approach through the Reuters dataset. The Details about the Reuters dataset and the rest of our work are as follows.

#### 4.1 Dataset and Software requirements:

**Dataset: Reuters-21578** (David,2004) is a popular dataset which is widely accepted for performing Text Classification experiments. The complete reuters-21578 dataset consists of 21578 documents which are published in Reuter's newswire in 1987. Every document in Reuters dataset is maintained in a SGML format.

Every article/document starts with a tag as mentioned as below

<REUTERS TOPICS=?? LEWISSPLIT=?? CGISPLIT=?? OLDID=?? NEWID=??>  
with the corresponding entries filling up “??”.

Every document ends with a closing tag </REUTERS>.

**TOPICS** can have “YES”, “NO” and “BYPASS”, where YES indicates that \*in the original data\* there was at least one entry in the TOPICS field. NO indicates that \*in the original data\* the story had no entries in the TOPICS field. BYPASS indicates that \*in the original data\* the story was marked with the string “bypass.”

**LEWISSPLIT** can have “TRAINING”, “TEST” and “NOT-USED”. TRAINING indicates it was used in the training set, TEST indicates it was used in test set and NOT-USED indicates it was not used in the experiments which are conducted earlier by LEWIS.



**CGISPLIT** can have “TRAINING-SET” and “PUBLISHED TEST-SET” indicating whether the document was in the training set or test set while conducting experiments reported by HAYES.

**OLDID** represents the Identification number used in the earlier Reuters-22173 collection.

**NEWID** represents the Identification number used in the new Reuters-21578 collection.

There are some other tags in addition to above mentioned tags. They are listed below

- <DATE> </DATE>
- <MKNOTE> </MKNOTE>
- <PLACES> </PLACES>
- <PEOPLE> </PEOPLE>
- <ORGS> </ORGS>
- <EXCHANGES> </EXCHANGES>
- <COMPANIES> </COMPANIES>
- <UNKNOWN> </UNKNOWN>
- <TEXT> </TEXT>

## **CATEGORIES**

There are 5 different sets of content related categories in Reuters-21578 collection.

They are

- EXCHANGES with 39 categories
- ORGS with 56 categories
- PEOPLE with 267 categories
- PLACES with 175 categories
- TOPICS with 135 categories.

## **Software Requirements:**

- WINDOWS 32/64 bit Operating System
- TMG(Text Matrix Generator Version 5.06)
- MATLAB v(7.7 or above)
- MICROSOFT Excel 2007.

## 4.2 Proposed Algorithm.

Document Classification is a Multi-Class Classification problem, in which the number of classes is more than two. To solve this problem, an OVA(One Versus All) approach has been used, in which the problem is decomposed in such a way that the documents belonging to the  $i^{\text{th}}$  class are considered as positive samples and the rest of the document collection are treated as negative samples. But the major flaw in this approach arises when the samples in  $i^{\text{th}}$  class are relatively very small when compared to the rest. This creates an imbalance in the training set, which increases the dominance of negative samples, thereby increasing its probability. But, the probability of a document belonging to a particular category should be independent of the ratio of negative and positive examples to make the classification mutually exclusive. Taking this imbalance into account, an algorithm with the number of positive samples being the same as the number of negative samples has been experimented in order to evaluate the accuracy of classification. The Algorithm which we have used for implementing Balanced OVA-SVM approach is as follows.

### Algorithm :

- 1) Count the total number of classes.
- 2) For each class “i”, count the number of documents belonging to it.
- 3) For each class “i”, select the same number of random samples as negative samples from the rest of the collection.
- 4) Implement a Linear binary SVM over the above obtained samples to calculate the classifier function for each class “i”.
- 5) Test these classifiers against the Test data and compare the accuracy .

To perform the experiment, we have taken a subset of Reuters-21578 collection with six different categories of 100 documents each, where 50 are chosen as Training Set and 50 are chosen as Test Set.

The Categories we have chosen are from TOPICS set in Reuters-21578 collection are as follows

- Acq
- Corn
- Crude
- Earn
- Grain
- Interest.

The documents belonging to the above mentioned categories were collected from the Reuters dataset. A training set consisting of 300 documents was selected to build a classifier function for each category and A Testing set of 300 documents was selected for evaluating the classifier functions which are deduced from the training phase. The input of the whole framework is a text document collection. These documents were initially tokenized. Then the stop words are removed from the obtained tokens and then a Stemming algorithm (Refer 2.3) was applied over it to obtain the final list of features representing the document collection. Then this whole collection is represented in a VSM where each row corresponds to a document and the each column corresponds to a feature. The value in the cell corresponding to the particular row and the column is a TF\*IDF value which was obtained through the TMG (Text Matrix Generator). Thus finally, this numerical data is processed through the algorithm mentioned in 4.3 to collect the output i.e. class labels. The results obtained through the above algorithm are evaluated by taking the classification percentage as the performance measure. The results are as follows

### 4.3 Results:

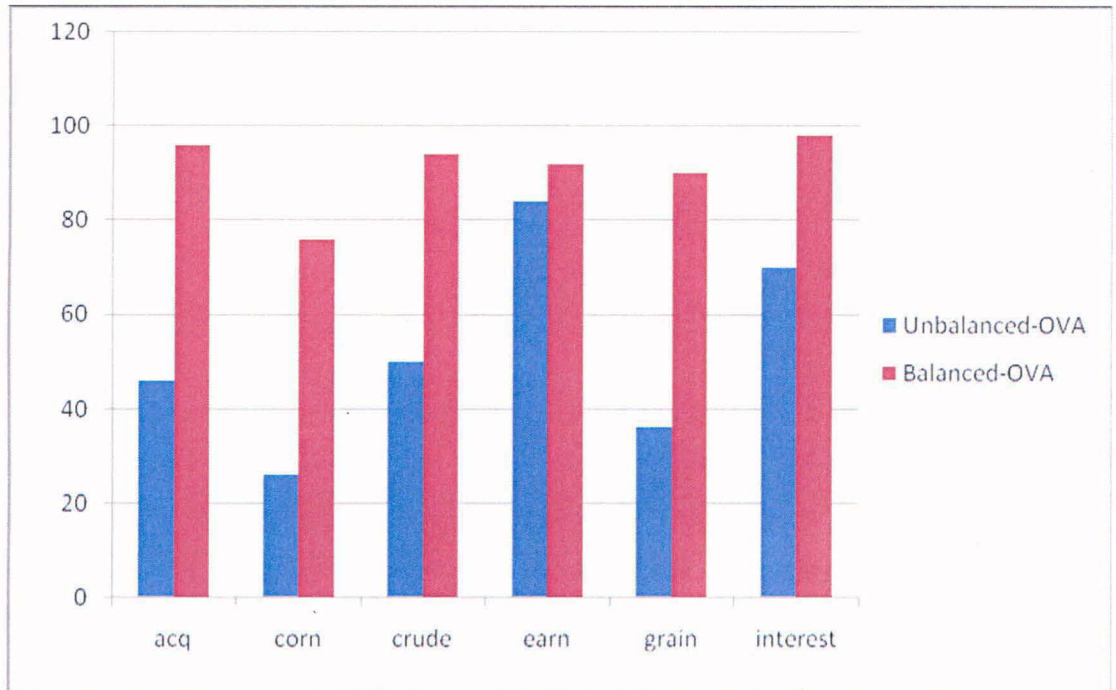
The classification accuracies as measured when a Test Set of 300 documents are applied using balanced OVA-SVM approach are as follows.

CLASS_NAME	ACCURACY_PERCENTAGE
Acq	96%
Corn	76%
Crude	94%
Earn	92%
Grain	90%
Interest	98%

where as the classification accuracies when an unbalanced OVA-SVM approach has been experimented are :

CLASS_NAME	ACCURACY_PERCENTAGE
Acq	46%
Corn	26%
Crude	50%
Earn	84%
Grain	36%
Interest	70%

This improvement can be noticed in the below diagram.



#### 4.4 Analysis:

By analyzing the result we can conclude that , if the ratio of negative to positive samples is more ,the percentage of false-positive documents is increasing when compared to the true-positive examples. This infers that the classifier accuracy in determining its exact class is low. This low classification accuracy has been observed because of the imbalance in the positive and negative samples. This imbalance has created a biasing among the classes to generate a classifier function. So, in order to maintain a mutually exclusiveness while classifying a new document it is suggested to maintain balanced samples during the training phase of a classifier.

## **Chapter-5**

### **Conclusion**

The objective our work is to study and analyze Multi-Class approach for document Classification. In our work, we have studied all the various phases through which documents have to undergo to make the data compatible with the Machine learning Algorithms. We have then studied various Machine Learning Algorithms to build a classifier function based on the training set of the document collection. This classifier function is then applied to the Test-data to study the behavior of the classifier function which is deduced from the training phase.

Our objective to work in the area of Document Classification has been motivated with the need to improve the Document Retrieval process since the digital information has been expanding in its volume because of the large amount of information available in the digital format. Document Retrieval directly depends on the way in which the documents were organized. With the balanced-OVA approach for Multi-Class Classification, we were able to achieve 91% accuracy of classification, which can surely make a positive impact while retrieving the relevant document from the available abundant information.

## References:

- Ambuj Tewari , Peter L. Bartlett, On the Consistency of Multiclass Classification Methods, The Journal of Machine Learning Research, 8, p.1007-1025, 2007.
- Aurangzeb Khan, Baharum Baharudin, Lam Hong Lee, Khairullah khan, “A Review of Machine Learning Algorithms for Text-Documents Classification”, Journal of advances in information technology, VOL. 1, NO. 1, 2010.
- Brucher H., G. Knolmayer, and M.A. Mittermayer, “Document Classification Methods for Organizing Explicit Knowledge,” technical report, Research Group Information Eng., Inst. Information System, Univ. of Bern, 2002
- Christopher D. Manning,Prabhakar Raghavan,Hinrich Schütze, “An Introduction To Information Retrieval”, Cambridge University Press, 2009.
- Damerau, Fred J. Generating and evaluating domain-oriented multi-word terms from texts. Information Processing &Management 29:433-447, 1993.
- Dasgupta A., “Feature selection methods for text classification.”, In Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 230 - 239, 2007.
- David D. Lewis, “Reuters-21578,Distribution 1.0”,  
<http://www.daviddlewis.com/resources/testcollections/reuters21578/reuters21578.tar.gz>,2004
- Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., & Lin, C.-J.. “LIBLINEAR: A Library for Large Linear Classification.” <http://www.csie.ntu.edu.tw/~cjlin/liblinear/>, 2008
- Fox, C. A stop list for general text. SIGIR FORUM, 24(1-2),19–35, 1990
- Hanuman Thota , Raghava Naidu Miriyala , Siva Prasad Akula, .Mrithyunjaya Rao , Chandra Sekhar,Vellanki ,Allam Appa Rao, Srinubabu Gedela , “Performance Comparative in Classification Algorithms Using Real Datasets”, JCSB/Vol.2 February 2009
- Harold Borko,Myrna Bernick, “Automatic Document Classification” Journal of the ACM (JACM),Volume 10 Issue 2, April 1963.
- Hodges, Julia, Shiyun Yie, Ray Reighart, and Lois Boggess. An automated system that assists in the generation of document indexes. Natural Language Engineering.2:137-160, 1996.
- Honrado, A. Leon, R. O'Donnel Sinclair, D., “A word stemming algorithm for the Spanish language” , in String Processing and Information Retrieval (Proceeding SPIRE), 2000
- LanM, Tan CL, Su J, Lu Y, “ Supervised and traditional term weighting methods for automatic text categorization”, IEEE Tran Pattern Anal Mach Intell 31(4):721–735,2009.

- M. F. Porter. "The Porter Stemming Algorithm" ,2003. machines. IEEE Transactions on Neural Networks & <http://www.tartarus.org/~martin/PorterStemmer>, 13(2):415-425, 2002.
- Manning, Christopher D., Schutze, Hinrich., "Foundations of statistical natural language processing". US: MIT Press,1999
- Manu konchady, A text book on " Text Mining Programming Applications",2006.
- Mohamed Aly. Survey on multiclass classification methods. November 2005.
- Ricardo Baeza-Yates and Berthier Ribeiro-Neto *Modern Information Retrieval*. ACM Press/Addison Wesley,1990.
- Richard O.Duda, Peter E Hart, David G Stork Pattern Classification. John Wiley & sons, 2006
- Sebastiani, F. Machine Learning in Automated Text Categorization, ACM Computing Surveys, 34, 1, pp. 1-47, 2002.
- Wei, C. P, Dong, Y. X, "A Mining-Based Category Evolution Approach to Managing Online Document Categories" in Proceedings of the 34<sup>th</sup> Annual Hawaii International Conference on System Sciences, 2001.
- Wen Zhang, Taketoshi Yoshida, Xijin Tang, "A comparative study of TF\_IDF, LSI and multi-words for text classification", Expert Systems with Applications.;38:(3):2758-2765., 2011.
- Zeimpekis D. and Gallopoulos E., "Tmg: A matlab toolbox for generating term-document matrices from text collections", Technical report, Computer Engineering & Informatics Department, University of Patras, Greece, 2005.