Global Journal of Computer Science and Technology: C
Software & Data Engineering
Volume 14 Issue 9 Version 1.0 Year 2014
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

Nomenclature and Benchmarking Models of Text Classification Models: Contemporary Affirmation of the Recent Literature

By Venkata Ramana. A & Dr. E. Kesavulu Reddy
S.V.University, India

Abstract- In this paper we present automated text classification in text mining that is gaining greater relevance in various fields every day. Text mining primarily focuses on developing text classification systems able to automatically classify huge volume of documents, comprising of unstructured and semi structured data. The process of retrieval, classification and summarization simplifies extract of information by the user. The finding of the ideal text classifier, feature generator and distinct dominant technique of feature selection leading all other previous research has received attention from researchers of diverse areas as information retrieval, machine learning and the theory of algorithms. To automatically classify and discover patterns from the different types of the documents [1], techniques like Machine Learning, Natural Language Processing (NLP) and Data Mining are applied together. In this paper we review some effective feature selection researches and show the results in a table form.

GJCST-C Classification : *H.1, E.4, H.2.1*

Strictly as per the compliance and regulations of:

Nomenclature and Benchmarking Models of Text Classification Models: Contemporary Affirmation of the Recent Literature

Venkata Ramana. A [α] & Dr. E. Kesavulu Reddy [σ]

Abstract- In this paper we present automated text classification in text mining that is gaining greater relevance in various fields every day. Text mining primarily focuses on developing text classification systems able to automatically classify huge volume of documents, comprising of unstructured and semi structured data. The process of retrieval, classification and summarization simplifies extract of information by the user. The finding of the ideal text classifier, feature generator and distinct dominant technique of feature selection leading all other previous research has received attention from researchers of diverse areas as information retrieval, machine learning and the theory of algorithms. To automatically classify and discover patterns from the different types of the documents [1], techniques like Machine Learning, Natural Language Processing (NLP) and Data Mining are applied together. *In this paper we review some effective feature selection researches and show the results in a table form.*

I. INTRODUCTION

Research on text categorization has emerged into a new level with the speed in advancement of internet technology. Various techniques were developed such as Machine Learning, Support Vector Machines (SVMs), KNN, Neural Network, Boosting and Naive Bayes variants [2] etc. Machine learning technique has evolved into a foremost model of text categorization [3].

Text classification is used in diverse fields for managing documents stored as texts in databases. The information today is composed of a large set of documents from multiple sources, such as news, articles, books, digital libraries, e-mail messages and web pages. Applications of text classification are being used in areas such as; in news delivery for classifying articles automatically into subjects and made available to users based on their search profile or interests.

In content management, grouping documents into many-sided categories is to simplify searching and browsing. In identifying spam mail where thequestionable mails are flagged as suspected spam and separated for batch deletion. In e-commerce for item descriptions in shopping and auction web sites where short texts are used for classification. In call centers offering support services, the text notes of call logs are classified with respect to defined criteria to identify trends periodically. These are but a few examples of how text classification is finding its way into applications and text classification systems.

The text classification process involves various steps like indexing, feature generation, feature filtering and feature selection.

a) Document Preprocessing

Document preprocessing step indexes the documents to minimize the complexity in documents.

b) Feature Selection

Feature selection methods are a preprocessing step. The selected features from the training set are then used to classify new incoming documents. The popular feature selection methods are document frequency, term frequency, chi-square statistic and Accuracy. Feature selection consists of the following steps;

i. Preprocessing - In the preprocessing stage, the various steps are;
   a. Feature Extraction - To generate text features based on the occurrence of the words in the document.
   b. Feature Filtering - To eliminate irrelevant or noisy features and non-discrimination in the data [4] effectively reducing the size of feature set.
   c. Document Representation - The document is transformed from a full text version to a vector space representation.

ii. Classifier building - Is performed by a score based strategy where words are scored with respect to their occurrence in the given document. This is predefined by the measure of weight of the word to decrease the high dimensionality space.

The selected features are then used to apply feature selection using different policies for text classification.

In this paper we discuss feature selection methodologies in text classification and review some effective methods for text classification and how performance can be increased [5, 6].

Author α: Research Scholar, Department of Computer Science S.V.University,Tirupat-India-517502. e-mail: avr_rdg@yahoo.co.in
Author σ: Assistant Professor, Department of Computer Science S.V.University,Tirupat-India-517502.
e-mail: ekreddysvu2008@gmail.com

II. Nomenclature of the Feature Selection Strategies for ext Classification

a) Text Classification

Text classification is the process of classifying the documents into predetermined categories. The engineering methodologies define a group of consistent rules for accurately classifying documents into a categorical set. The nature of classifying the documents can be of 3 types: i) Unsupervised, ii) Supervised and iii) Semi Supervised. Automatic text classification widely studied since the last few years has rapidly evolved due to fast development of internet technology.

1. Document Preprocessing - Document preprocessing step indexing of documents to minimize the complexity in the documents.
   The documents are classified into 2 types based on the class:
   (i) Single Label and
   (ii) Multi-Label.
   Single label document are fit for only a single class whereas multi label documents may be fit for single and multiple classes.
2. Feature Selections - Feature selection involves generation of filters focusing on relevant and informative data. The feature filtering process and feature selection are used for selecting features useful for text categorization with respect to scalability, efficiency as well as accuracy.
3. Applying Feature Selection: To apply feature selection in text categorization there are two major policies;
   (i) Local Policy: In the local policy a varied set of features are chosen from each class not dependent on other classes providing identical weight to each class and optimizing the performance by choosing the most important features for each class.
   (ii) Global policy: In the global policy a single set of features are chosen from all classes, providing a global view of the whole dataset and a single global score from the local scores [8, 9].
   E.g.: The automatic labeling of every inbound news item with a predefined subject like "media", "politics" or "sports" or "art". First a training set D = (d1….. dn) of documents that are already labeled with a class C1, C2 (e.g. politics, sport) is selected. Next a classification model that is competent enough for labeling a new document d of the vertical with the right class is determined. The most commonly used document representation is known as vector space model (SMART) [7]. *In this paper we focus on classification of single label document.*
4. The challenges related with text classification come from many fronts and are mostly of three types;
   (i) Selection of a suitable data structure to represent the documents.
   (ii) Selection of right objective functions to prevent high formal dimensionality of the data and consequent algorithmic issues. It mainly leads to over-fitting where a classifier fits the training dataset well but performs inadequately on cases exterior to the training dataset that result in high computational overheads and increased training period.
   (iii) Text classification problems at times have only very limited training data present that poses a high difficulty for learning and the classification.

b) Text Feature Generators and Feature Extraction

The process of Text Feature Generation is to come up with an array of feature generators that make the feature selector choose powerful predictive features.

First it is important to ascertain what qualifies to be counted as a word or a term. Difficulty arises in instances like 'HP-UX' qualifying as single word or couple of words and how to ascertain the type of term '650-857-1501'? In programming, a simple solution requires contiguous sequence of alphabetic characters; or alphanumeric characters including identifiers like 'ioctl32', that are occasionally helpful. By means of the Posix regular expression \p{L&}+ we evade breaking 'naıve' in two and also several accented words in French, German, etc. Difficulty arises in case of words like 'win 32', 'can't' or words that may be hyphenated over a line break. Similar to several other data cleaning techniques, the list of exceptions is never-ending and we have to limit the expectation and expect for an 80%-20% exchange. An advantage is semantic errors in word parsing are generally observed by the core learning algorithm. So their statistical properties are of importance not its readability or intuitiveness to people. Major feature generators that are useful differ according to the domain text qualities. The typical feature generators applied are;

i. *Word Merging*

Merging is a technique of decreasing the size of the feature space considerably by merging different word variants and treating the new entity as a single feature. This significantly improves the predictive value of certain features.

Force conversion of all letters to lowercase is a generally accepted technique. Letters at the start of a sentence, which does alter the word's meaning, and helps lessen the dispersion. In case of proper nouns, it frequently conflates other word meanings, e.g. 'Bush' or 'LaTeX.'

Several word stemming algorithms could be used for merging multiple related word forms. For

instance, 'cat,' 'cats,' 'catlike' and 'catty' can all be merged into a common feature. Stemming normally is advantageous for recall however affects the precision. As in the example above if one is searching for 'catty' and the word is considered the same as 'cat,' then essentially a definite amount of precision is lost. In case of exceedingly skewed class distributions, this loss may greatly affect precision.

Also stemming algorithms give errors of both over-stemming and under-stemming; however, the semantics occupy less significance compared to feature's statistical properties. Stemmers have to be individually designed for every natural language and there are several fine stemmers existing for Roman languages whereas for other languages such as Hebrew and Arabic the stemming process becomes difficult. Further for some applications of text classification, there occurs mixing of multiple natural languages together, at times even inside only one training case. This necessitates a language recognizer to determine the type of stemming algorithm that needs to be used for each case or each sentence. However this level of complexity and slowdown is not desired. Stemming by merely considering the initial few characters of every word may give equal classification accuracy for several classification problems. Misspellings that commonly occur in technical texts or blogs and resolving with an automatic spelling correction step provided in the processing pipeline is occasionally proposed to ease classification but the errors revealed may overshadow the supposed advantage. A familiar problem with the spell checker is that out-of-vocabulary (OOV) words are forced to the nearest known word that can have totally different meaning. This is generally seen with technical terms that can be crucial predictors. Common misspellings may give frequent misspelled form and appear as a useful feature, e.g. 'volcano.'

Out-of-vocabulary (OOV) words are mostly words like abbreviations and acronyms found in governmental or technical texts. In case if glossaries can be referred, the short and long forms can be merged into a single term. Though several acronym dictionaries exist online, there are many types of short acronyms that are extremely document and even domain-specific. A few of the researches have shown success identifying acronym definitions in text, such as '(OOV)' above, that gives a locally clear-cut definition for the term. Online thesauruses may as well be used to merge together dissimilar words, e.g. to resolve the 'color' vs. 'hue' problem however the technique hardly ever helps, as multiple meanings exist for many words that distorted their final meanings. To disambiguate word meanings correctly would require a much deeper understanding of the text than is needed for text classification. However, this problem is overcome by using domain-specific thesauruses of synonyms. For instance in representing a huge set of part numbers corresponding to a common product line a single feature to represent all proves very beneficial.

While merging related words with each other can turn out features with additional frequent occurrence (characteristically with greater recall and lower precision).

ii. *Word Phrases*

Identifying multiple word phrases as a single term can generate rarer, highly specific features (which regularly aid precision and have lower recall), e.g. 'John Denver' or 'user interface.' However instead of using a dictionary of phrases as in the above case an easy technique is to consider all successive pairs of words as a phrase term and let feature selection decide which are helpful for prediction. Modeled on the new technology of online searching, the recent trend to eliminate spaces in proper names, e.g. 'SourceForge,' gives the specificity of phrases devoid of any particular software deliberations. Also the same can be applied to phrases having three or more words with intermittently more specificity and also with strictly decreasing frequency. Maximum advantage is gained with two-word phrases [10] to some extent as the parts of the phrase may previously have the identical statistical properties, e.g. the phrase with four words 'United States of America' is previously enclosed by the two-word phrase 'United States.' Also, the extent of a two-word phrase can be extended by removing general stopwords, e.g. 'head of the household' turns into 'head household.' However the stop word lists are language specific and have limitations. The main advantage of classification lies in increasing the extent of phrases, instead of removing frequently useless words that could be already removed in a language-independent fashion with maximum feature selection techniques.

iii. *Character N-grams*

The word identification techniques discussed previously do not succeed in some cases and cannot succeed in spotting some good features. For instance, languages like Chinese and Japanese do not employ a space character. Segmenting such text into words is difficult, however approximately comparable accuracy might be gained just by means of every set of adjoining Unicode characters as features $n-grams$. Definitely several of the variants will be worthless; however feature selection is able to identify the maximum predictive features. In case of languages that utilize the Latin character set, 3-grams or 6-grams may be right. For example, $n-grams$ would obtain the real meaning of common technical text patterns such a $HP-UX$ 11.0','while(◁▷){',#!/bin/',and':)' .Phrases of two adjoining $n-grams$ simply equate to $(2n)$ grams. The number of potential increase exponentially with where in reality it is merely a little fraction of the

possibilities that arise in actual training examples and only a small part of those could be predictive.

The general Adobe PDF document format, records the position of each character on the page and does not clearly records spaces. Software libraries to pull out the text from PDF utilize heuristics to determine where to output a space character. Due to this reason text extracts either occasionally overlook spaces amid the words or have a space character placed among every pair of letters. Obviously, such issues will cause chaos with a classifier that relies on spaces to recognize words. A more strong technique is for the feature generator to remove all whitespace that gives n-grams from the resulting sequence.

iv. *Multi-Field Records*

Multi-field records are regularly used as maximum applications have multiple text (and non-text) fields with respect to each record, though in most applications of text classification research deal with training cases as a single string. These fields in document management are usually title, author, abstract, key-words, body and references. In the field of technical support, these may be title, product, keywords, engineer, customer, symptoms, problem description, and solution. Additionally classifying long strings, e.g. arbitrary file contents, the first few kilobytes are usually taken as a separate field and that is generally adequate for generating desired features without the need to deal with huge files like star or zip archives.

The simplest approach is to concatenate all strings together. However, supposing the classification goal is to separate technical support cases by product type and model, then the most informative features may be generated from the product description field alone, and concatenating all fields will tend to water down the specificity of the features.

Another uncomplicated strategy is to provide every field with its own separate bag-of-words feature space. For example the word 'OfficeJet' given as the title field would be addressed as if it were not connected to a feature for the similar word in the product field. Occasionally multiple fields are required to be combined, and at the same time set aside as separate and while the rest are isolated. Such choices are done manually and an automated search improves computation time for the search and essentially reduces the expert's time, and it may identify better options not possible in manual search.

v. *Other properties*

In case of certain classification issues, text properties other than words or $n-grams$ generate the key predictors for high accuracy. Certain kinds of spam use deceptions such as '4*ree* v!@gr@4 u!' 'to prevent word-based features though these might easily be found by features identifying their abnormal word lengths and the density symbols. Similarly identifying Perl or awkcode, is done with specific alphanumeric identifiers, less specific in occurrence than the distribution of particular keywords and special characters. Information of formatting like the amount of whitespace, the word count, or the average number of words per line can be key features for specific tasks.

Here task-specific features created are usually extremely expensive like parsing particular XML structures that hold name-value pairs. The features being task-specific, it is especially difficult to provide common useful comments about their generation or selection. The insufficient information available regarding task-specific features in literature of text classification overrides their true importance in many practical applications.

vi. *Feature values*

Next with the determination of the word that could be considered as a feature term, the significance of the numerical feature must be ascertained. For certain purposes a binary value is enough to represent if the term actually occurs. This depiction is employed by the Bernoulli formulation of the Naive Bayes classifier [11]. A lot of other classifiers utilize the term frequency $tf_{t,k}$ (the word count in document k) directly as the feature value, e.g. the Multinomial Naive Bayes classifier [11].

The support vector machine (SVM) has demonstrated to be extremely effective in text classification. In such kernel techniques, the distance between two feature vectors is normally calculated as their dot product (cosine similarity), which is dominated by the dimensions with larger values. In order to avoid a situation where the extremely frequent but non-discriminative words (such as stop-words) dominate the distance function, we can either use binary features or weight of the term frequency value $tf_{t,k}$ inversely to the feature's document frequency df_t in the corpus (the number of documents in which the word appears one or more times). In this way, very common words are downplayed. This technique commonly known as TF.IDF has a many variants, one form being $tf_{t,k} \times \log\left(\frac{M+1}{df_t+1}\right)$, where M is the number of documents. Though this technique necessitates greater computation and storage per feature compared to binary features, it may further offer superior accuracy for kernel methods. The document length typically varies according to the document type short or long word counts with the same topic. To make these feature vectors more comparable, the $tf_{t,k}$ values can be normalized to make the length (Euclidean norm) of every feature vector equals to 1.

c) Feature Filtering

Feature selection process scores each possible feature with respect to a specific feature selection metric and selecting the most excellent k features. As discussed in the previous sections, a wide array of feature generation approaches, we now concentrate on feature filtering.

Selection of the best feature differs extensively from job to job where a number of values ought to be used. Keywords are extracted and examined according to criteria such as the incidence of repeated words or frequently used terms not definite to any category to eliminate irrelevant and noisy information. Feature filtering with respect to the training class labels scores each feature independently. The scoring counts the number of feature examples in training positive- and negative-class separately and next over these it computes a function. Increase in the number of features results in increase in the time necessary for initiation and training time affecting the accurateness of the classifier. To achieve dimensionality reduction the two most common approaches in machine learning or data mining are the filter and the wrapper [8, 12].

1. The filter chooses a subset of features by filtering based on scores assigned by specific weighting and is independent of any learning algorithm.

i. First in the process, the a) rare words and b) common words are removed;

a. The rare words can be eliminated, since they may not have any presence in future classifications. For instance, words with presence less than two times can be eliminated. Word frequencies characteristically pursue a *Zipf* distribution: the frequency of each word's incidence is proportional to where rank is its rank among words sorted by frequency, and is a fitting factor close to 1.0 (Miller 1958) [13]. Since of the total distinct words, a part equal to half of the total number can appear only once, so deleting terms below a specific low rate of incidence generates great savings. The meticulous choice of threshold value may affect the accuracy. If we remove rare words with respect to the count of the entire dataset prior to splitting off a training set, it will result in leaking a part of the information regarding the test set to the training phase. Avoiding major resource allocation for cross-validation studies, the research creates feasibility since avoiding class labels of the test set.

b. Excessively common words, or regular words such as conjunctions, prepositions and articles such as 'a' and 'of', may also be eliminated because of their high frequency of occurrence so as to not discriminate any specific class. Common words can be recognized either by a threshold on the number of documents the word occurs in, e.g. if it occurs in over half of all documents, or by supplying a *stop word* list. Stop word are language-specific and often domain-specific. Depending on the classification task, they may run the risk of removing words that are essential predictors, e.g. the word 'can' is discriminating between 'aluminum' and 'glass' recycling.

ii. Second in the process it is stated that the commonprocess of *stemming* or *lemmatizing—merging* various word forms such as plurals and verb conjugations into one distinct term—also reduces the number of features to be considered and which however is a feature engineering option. Suffix stripping Suffix stripping is used to stem words having a common stem and similar meanings can be merged into one term. Example, "invent," "invented," "inventing,""inventive," "invention," and "inventions" can be combined into the same term "invent" by removing the suffixes.

iii. Attribute selection. Other than the simple steps of stop-word removal and suffix stripping, attribute selection is a important step that can usually lessen significantly the attributes count. The attributes are typically term weights (determined by an indexing method) and the attribute space results in high computational overhead and increased training times making its removal necessary. It is depicted as a vector of features in a vector space model [5] or "bag-of-words" in a probabilistic model; features are the components in a vector or "words".

The filtering process is usually chosen because it is easily understood and has independent classifiers. In the filter approach, the attributes are evaluated based on some relevance measure, independent of any learning algorithm. In this paper, we use the term "relevance" informally to refer to the degree to which an attribute is relevant to the prediction of the class. For a proper definition of "relevance," please refer to Avrim and Pat (1997) [14]. The relevance measure is designed to measure the dependency between the class and an attribute and the attributes most applicable for predicting the class are selected. Since the attributes required to be evaluated only one time, the filter method is computationally efficient.

However, the attributes selected are not particularly trained on the learning algorithm used as it is actually not used in building the classifier. Also as the attributes are mostly individually assessed, the selected attributes, when considered as a set, may not be the most excellent possible subset.

In automatic classification, feature size reduction using simple filtering methods like stop words deletion or words stemming gives inadequate results. So a feature selection technique or algorithms ought to be used to optimize the performance of classification systems for visual detection.

2. The wrapper - The wrapper approach fundamentally depends on the learning algorithm. There are two major components in the wrapper approach: the (i) Performance Evaluation Method and (ii) Search Method.

   Cross-validation has been shown to be an effective performance evaluation. Cross- validation used to select feature generator is also used to tune the parameter automatically on the training data for selecting parameters for the induction algorithm, like the popular complexity constant C used in the SVM model. Optimizing each parameter in its own nested loop is the easiest to program, however, with each successive nesting a lesser fraction of the training data is provided to the induction algorithm. For instance if nested 5-fold cross-validation is used to decide on the feature generator, the number of features and also the complexity constant then the inner-most loop trains with only half of the training set: $\frac{4}{5}\times\frac{4}{5}\times\frac{4}{5}=51\%$. However the small training set fails in comparison to the full size training set for determining the optimal parameter values. So a 10-fold cross-validation, in spite of the high computing cost, is typically preferred to 2-fold cross-validation. As an alternative, a single loop of cross-validation must be combined with a multi parameter search strategy. The easiest way of programming is done by measuring the cross validation accuracy (or F-measure) at each point on a simple grid, and then deciding on the top parameters. There has been a huge research done on multi-parameter optimization and the methods though more complex to program are much more efficient. In the wrapper approach, the subset of features is chosen based on the Accuracy of classifiers. Exhaustively trying all
3. the subsets is not computationally feasible [15]. Technically, the wrapper is relatively difficult to implement, especially with a large amount of data.

   An optional feature selection process is the depiction of feature value. Usually for most of the cases it is adequate if a Boolean indicator for the word occurrence in the document. Additional options are; the number of times in the document the word occurs, the frequency of its incidence normalized by the length of the document, the count normalized by the inverse document frequency of the word. In cases where there are wide variations of document length, it can be essential to normalize the counts. In our study the datasets of most documents considered are short, that does not require any normalization. Also the words in short documents most probably do not repeat, resulting in Boolean word indicators to be as informative as counts. The result is of vast savings in training resources and in the search space of the induction algorithm. If not it may attempt to discrete each feature optimally, searching over the number of bins and each bin's threshold. In our study, we had chosen Boolean indicators for each feature that enlarges the selection of FS metrics that would be considered, e.g. Odds Ratio deals with Boolean features, and was reported by Mladenic and Grobelnik (1999) to perform well [16].
4. A final alternative in the FS strategy is if we can remove out all negatively correlated features. Some think that classifiers built from positive features alone will be efficient in the particular cases wherever the background class may shift and retraining is not required, which however has to be proved. Further, certain classifiers work basically with positive features, e.g. the Multinomial Naïve Bayes model and results prove it to be superior compared to previous Naïve Bayes model (McCallum & Nigam, 1998) [11], though significantly mediocre to some induction methods for text classification (e.g., Yang & Liu, 1999; Dumais et al., 1998) [17, 35]. Negative features are abundant due to the large class skew however rather important in practically: For example, while scanning a catalog of Web search results intended for the author's home page if many of results on George Foreman the boxer are shown they could be removed from the search with the terms 'boxer' and 'champion,' which is no concern to the author.

d) Feature Selection

i. *Prologue*

Feature selection refers to the selection of those features that are more important for relevant and informative data useful for text categorization. It enhances the scalability, efficiency as well as accuracy of a text classifier and plays a very important role in later steps influencing overall system performance. As many systems are large scale in various areas of data collection, feature selection is an important and widely grown. Some of basic applications of feature selection are Image Recognition, Clustering, Text Categorization, System monitoring, Rule Induction and Bioinformatics. (Jensen 2005) [48].

Feature selection consists after preprocessing and feature filtering selects the best features depending on the highest scores.

In document categorization or text classification, various methods of feature selection are used and they are;

Filter methods evaluate each feature independently and determine a ranking of all the features, from which the top ranked features are selected [4]. They can also be used as a pre-processing step to reduce the feature dimensionality to enable other, less scalable methods.

Wrapper methods search for the 'best' subset of features, repeatedly evaluating different feature subsets via cross validation with a particular induction algorithm. Wrapper methods have traditionally sought specific combinations of individual features from the power set of features, but this approach scales poorly for the large number of features inherent with classifying text. The wrapper methods have higher time complexity and accuracy compared to filter methods.

Embedded methods build a usually linear prediction model that tries to maximize the goodness-of-fit of the model and at the same time minimizes the number of input features [18].

Cross-validation method is used to select the best among feature generators and optimize other parameters, is somewhat like a wrapper method, but one that involves far fewer runs of the induction algorithm than typical wrapper feature selection.

Some variants build a classifier on the complete dataset where the classifier deletes the features it finds no application iteratively [19] as they are minimally scalable. However in case of large feature spaces, the memory may be insufficient for representing all the potential features and vectors. In this study such methods are not considered.

Text Classifiers Evaluation: Performance evaluation of the classifiers is the last stage of text classification. It is an experimental evaluation and not an analytical one. It is based on the capability and effectiveness of a classifier in taking the right categorization decisions rather than the Efficiency issues. The performance is measured with the help of many techniques like precision, recall [4], fallout, error, accuracy etc.;

(i) Precision w.r.t. ci (Pri) is defined as the as the probability that if a random document dx is classified under ci, this decision is correct.

(ii) Recall w.r.t. ci (Rei) is defined as the conditional that, if a random document dx ought to be classified under ci, this decision is taken, where TPi–The number of document correctly assigned to this category.

(iii) FN - The number of document incorrectly assigned to this category. FPi - The number of document incorrectly rejected assigned to this category. TNi -The number of document correctly rejected assigned to this category. Fallout = FNi / FNi + TNi

(iv) Error = FNi +FPi / TPi + FNi +FPi +TNi

(v) Accuracy = TPi + TNi

For obtaining estimates of precision and recall relative to the whole category set, methods such as i) Micro-averaging, ii) Macro-averaging are mostly used. Other measures like iii) Break–even point, iv) F-measure and v) Interpolation [7] are also used.

ii. *Feature Selection Methods and metrics*

The central design of Feature Selection (FS) is the selection of a subset of features from the original documents. In data mining or machine learning the methods that are regularly used for feature selection are: *Filter methods and Wrapper methods [12].*

Filtering methods: Filter methods use statistical techniques and are independent of the learning algorithm for FS. The filter method is usually chosen because it is easily understood and has independent classifiers. The various filtering metrics researched are; Document Frequency (DF), Term Frequency (TF-IDF), Chi Squared CHI2, Information Gain (IG), Accuracy (Acc2), Mutual Information (MI) [2], Association Word Mining [21], Expected Cross Entropy, Odds Ratio, Sampling Method, Gini Index etc. [22]. *The filter method is appropriate to treat very large feature space, is also the most scalable and is the focus of study in this paper;*

All features are evaluated independently with respect to the class labels in the training set to establish a ranking and the top ranking or scoring features are chosen [18] for classification. *A few filtering techniques metrics or scoring schemes are studied in this paper as they can be applied for most of the texts classification problems.* These filter metrics use a term goodness criterion threshold to attain a preferred degree of term purging in the entire terminology of a document. They are; (i) DF, (ii) TF-IDF (ii) Chi-square (iv) IG and (v) Acc2.

(i) DF: Document frequency is an easiest way of assessing feature significance; it simply determines in how many documents a word occurs where choosing regular words will advance the probability of the features presence in the next test cases. The DF of a specific term simply corresponds to the number of documents in a class containing that term [2, 4, 23]. It is computed independent of class labels and the total test set can also be included in the computation.

(ii) TF-IDF: Term Frequency method associates maximum scores to the terms that appear in some documents with a max frequency. That is a term occurring more number of times in a document means it is more discriminative whereas if it occurs in the majority of the documents, then it is less discriminative for the content.

(iii) Chi2Max: Chi-Squared is a statistical test that is widely used. It calculates the independence of 2 events between feature occurrence and class value [24] and the deviation from the expected distribution based on the assumption of actual independence.

(iv) IG: Information Gain measures in how much data the occurrence or nonexistence of a term or it

measures the decrease in entropy when the feature is given vs. absent that helps in deciding the correct classification choice for any class [2, 25]. IG reaches its highest value if a term is a perfect sign for class association, that is, if the term is occurs in a document and if and only if the document belongs to the respective class.

(v) Acc2: Accuracy considers only the number of documents in which the term occurs, without taking into account the number of actual documents.

Wrapper methods: Wrapper methods employ learning algorithm as the appraisal function. Classic AI search methods-such as simulated-annealing-to or greedy hill-climbing [19] explore for the 'best' subset of features and repetitively appraise different feature subsets with cross validation using a specific induction algorithm (Nejad et al., 2013) [20].

(i) Sequential Forward Selection (SFS),
(ii) Sequential Backward Selection (SBS),
(iii) Neural Networks (Dave, 2011, Eyheramendy and Madigan, 2005) [26],
(iv) Genetic Algorithm (GA) based selection.

The fourth method or Genetic Selection (GS) is a new FS employs the genetic algorithm (GA) optimization especially for issues of high dimensionality [3]. GA based selection has demonstrated to be reasonably capable and quick among many suboptimal search algorithms like sequential forward and backward selections [27]. GA theory is based on the survival of the fittest solutions from the entire potential solutions for a given issue [28]. Accordingly the latest generations formed from the surviving solutions are estimated to offer better accurateness to the best possible solution. The solutions match to chromosomes that are programmed with a proper alphabet. The fitness value of each chromosome is defined by a fitness function. New generations are generated by means of genetic operators i.e. crossover and mutation, with definite probabilities on the fittest members of the entire set. The primary set can be defined arbitrarily or manually. Population size, number of generations, probability of crossover and mutation are defined empirically. GS technique is simple and helpful and the chromosome length is equivalent to the dimension of a full feature set. The chromosomes are encoded as {0, 1} binary alphabet. In a chromosome, the indices denoted as "1" specify the chosen features, whereas "0" refers to the features not selected ones. For example, a chromosome defined as;

{ 1 0 1 0 1 1 0 0 0 1} implies that the 1st, 3rd, 5th, 6th, and 10th features are chosen and the remaining are eliminated. The fitness value related to a chromosome is defined by a specific success factor that is generated with the chosen features. A few instances of genetic feature selection research are presented in papers [27, 29, 30].

III. Contemporary Affirmation of the Literature About text Classification, Feature Selection and Filtering Strategies

a) *The Feature Selection Techniques or Methods*

This paper concentrates on filter methods because; i) they are comparatively more scalable to huge collections and ii) their objectives considerations are diverse from those of classifiers. There are four filter methods core, variant, combined and redundancy reducing methods;

i. *Core Methods*

We incorporated a few feature selection methods; (DF) document frequency (just count the number of documents including the feature), (IG) information gain (number of bits of information collected for category prediction for a particular feature) and (CHI) (measuring the absence of independence between a term and the category) [2]. Also the binary version of information gain (IG2) was included because it is widely used. Mutual information due to its poor performance was excluded.

ii. *Variant methods*

- Term frequency is used as a substitute for a binary value for every document counted in the scores (such variants would be recognized by TF in the results; because not one of these methods were amid the top three performers, they are not shown on the graphs.)
- The methods having one value per type (IG2, CHI, IG), we used average and also the maximum value as the score. (Identified by AVG, MAX)
- The methods, IG and CHI, were also tested with their generalized versions (cumulating evidence from all classes) are recognized by GEN.
- Also the rare words are eliminated (DF $\leq$ 5) (identified by "cut")

iii. *Combined methods*

We analyzed the correlation among some of the best performing methods and observed that a few (like the multiclass version of IG and CHI MAX) have minimum negative correlation, indicating a promising performance gain in combination. The two methods were combined by first normalizing the scores for all word and next selecting the higher of the two scores (thus giving an OR with equal weights to the two methods to be combined).

iv. *Redundancy Reducing Methods*

We executed a variant of the μ co-occurrence method as in [31] that utilizes the other filter feature

selection methods as a starting point. With the use of a tunable, arbitrary constant-size pool, the complexity analysis in [31] is seen to improve and we believe that using a percentage of the vocabulary is more suitable as the size of the vocabulary may vary broadly with collections. We executed a variant of this method, with a percentage-based initial pool (1% instead of 5 terms), smooth weighting in place of collection-dependent thresholding on the coocurrence and the multi-class version in place of 2-class.

As discussed above, the total number of features, variants as well as combinations is well over 100 where each of the core methods has an average of approximately 3 variants, and the cca. 15 resulting methods were combined in pairs.

b) Feature Selection Metrics - Evaluation and Exploration

In this study, we discuss criteria defining the feature selection metrics that have demonstrated excellent performance in text categorization. The five widely used feature selection metrics are: Document Frequency thresholding (DF), Term frequency-Inverse document frequency (TF-IDF), Chi-Square statistics (CHI), Information Gain (IG), Accuracy2 (Acc2);

i. *Document Frequency Threshold (DF)*

Document frequency is a metric used for remove the rare terms that are non-informative and confusing for classification. It is a very basic and accepted method that determines the number of documents in which the term appears without class labels [4, 32, 33]. It is based on the theory that a term belonging to a less number of documents is not an excellent feature for the classification task [34]. So, only the words that are present in a number of documents more than a defined threshold are chosen. This threshold can be calculated using a training set. Given a term t_k , this condition can be computed globally on the collection $(DFG(t_k))$ or on each category c_i $(DFL(t_k, c_i))$

$$DFL(t_k, c_i) = P(t \mid c) \approx \frac{A}{A+C}$$

$$DFL(t_k, c_i) = P(t \mid c) \approx \frac{A}{A+C}$$

One common technique to use this method is removing all the words which are present in less than x documents, x varying between 1 and 3 [34, 35, 36]. Commonly, this procedure is used with another feature selection method. The terms with low or high document frequency are frequently referred to as rare or common terms, in that order. The FS method discussed here is based on the first basic measurement that the terms with higher document frequency are more helpful for classification. However this supposition fails in giving any information sometimes. For instance the stop words (e.g., the, a, an) have very high DF scores, but hardly ever add to classification. More specifically, this uncomplicated method shoes good performance in few topic-based classification tasks (Yang and Pedersen, 1997).

ii. *Term Frequency-inverse Document Frequency (TF-IDF)*

The $tf - idf$ feature selection method is based on selecting the words with the highest $tf - idf$ scores. This method gives the highest scores to the words that are present in some documents with a high frequency meaning that it is more discriminative and if it is present in max number of the documents and then it is less discriminative for the content.

In $tf - idf$ [25], tf represents the term frequency of a term in a document. idf is defined as the inverse document frequency, i.e., the ratio of the total number of documents present in a dataset to the number of documents a given term appears in. A higher idf of a term implies that the term appears in relatively few documents and may be more significant at some stage in the process of text classification. *tfidf* is mostly used for term weighing in the field of information retrieval and is also used in text classification. The *tfidf* of a term t_k in document d_i is defined using;

$$tfidf(t_k, d_t) = tf(t_{,k}\, d_t) \log \frac{|D|}{df(t_k)}$$

Where $|D|$ refers to the total number of documents in a dataset; $tf(t_k, d_t)$ is the term frequency of a term t_k in document d_i; and $df(t_k)$ refers to the number of documents in which term t_k appears

iii. *Chi-square Statistics (CHI)*

Chi-square (2) statistics is a method commonly used in text categorization [4, 8, 32, 33], is a relevant measure, effective in text classification applications (Sebastiani 2002) [5] to measure the independence of two random variables (Liu and Setiono 1995) [37]. In text categorization, the two random variables are occurrence of term t_k and occurrence of class c_t and chi-square statistics measures the independence between t_k and c_t . The formula for chi-square score is:

$$CHI(t_k, c_t) = N \times \frac{[P(t_k, c_t)P(\overline{t_k}, \overline{c_t}) - P(\overline{t_k}, c_t)P(t_k, \overline{c_t})]^2}{P(t_k)P(\overline{t_k})P(c_t)P(\overline{c_t})}$$

where $P(t_k)$ is the percentage of documents in which term t_k occurs, $P(\overline{t_k})$ is the percentage of documents in which term t_k does not occur, $P(c_t)$ is the percentage of documents belonging to class c_t, $P(\overline{c_t})$ is the percentage of documents not belonging to class c_t, $P(t_k, c_t)$, is the percentage of documents belonging to class c_t in which term t_k occurs, $P(\overline{t_k}, \overline{c_i})$, is the percentage of documents not belonging to class c_t in which term t_k does not occur, $P(\overline{t_k}, c_i)$ is the percentage of documents belonging to class c_t in which term t_k does not occur and $P(t_k, \overline{c_t})$ is the percentage of documents not belonging to class c_t in which term t_k occurs. If chi-square score of a term t_k is of low value, this means t_k is independent from the class c_t and if chi-square score of a term t_k is of high value, this means t_k is dependent of the class c_t. Thus the chi-square feature selection method selects the terms with the highest chi-square score which are more informative for classification.

Due to the presence of words that rarely occurs and also due to limited number of positive training instances irregular behavior for very small expected counts, common in text classification, is observed.

iv. *Information Gain (IG)*

An accepted feature selection method in text categorization, information gain (IG) [4, 33, 38, 39] measures how much information the occurrence or nonexistence of a term helps to decide the correct classification criteria for any class [2,4, 40]. The terms with scores of highest information gain has maximum information about the classes. Here class membership and the presence/absence of a specific term in a certain category are seen as random variables; one computes how much information about the class membership is gained by knowing the presence/absence statistics. If the class membership is defined as a random variable c with two values, positive (c) and negative $(\bar{c})$, and a term is likewise seen as a random variable t with two values, present (t) and absent $(\bar{t})$, then information gain is calculated as;

$$IG(t_k, c_t) = \sum_{c \in [c_t, \overline{c_t}]} \sum_{t \in [t_k, \overline{t_k}]} P(t/c) \log \frac{P(t/c)}{P(t)P(c)}$$

5. Accuracy2 (Acc2)

Accuracy2 has showed better efficiency in comparison to other feature selection metrics in the earlier studies [4, 32]. In this metric, only the number of documents in which the term occurs is considered and not the number of actual documents. It measures the difference between the documents belonging to a class with a distributed term in the documents not belonging to t_k that class. Thus, the term that never occurs in a class c_t can be selected as a feature for c_t. Below is the formula for calculation of accuracy2 score:

$$Acc2(t_k, c_t) = \left| P(t_k, c_t) - P(t_k, \overline{c_t}) \right|$$

c) Feature Classification Strategies

The classification approaches for categorizing the selected features is performed by using 3 methods;

(i) Binary Classification,
(ii) Multi-Class Classification, and
(iii) Hierarchical Classification.

i. *Binary Classification*

Binary or binomial classification is categorizing the components of a given set into two sets based on specified classification rule. Binary domain tasks are regularly used and also as a subroutine to address maximum types of multi-class tasks.

Some usual binary classification tasks are (i) the effectiveness to the user in distinguishing spam email from good email. (ii) a "pass or fail" test method or quality control in factories; i.e. deciding if a specification has or has not been met: a Go/no go classification; (iii) an item may have a Qualitative property; it does or does not have a specified characteristic information retrieval, namely deciding whether a page or an article should be in the result set of a search or not – the classification property is the relevance of the article. (iv) to decide in medical testing if a patient has a specific disease or not – the classification property is the presence of the disease.

The two groups are not symmetric and this is observed in many practical binary classification instances. The focus is on the relative proportion of varied types of errors rather than on the overall accuracy. For example, in medical testing, a false positive (detecting a disease when it is not present) is considered differently from a false negative (not detecting a disease when it is present).

ii. *Multi-Class Classification*

There are two main types of multi-class classification:

(i) Single-label $(1-of-n)$ classification, where every individual case belongs to exactly one of the n classes. In the single-label case, many induction algorithms function by decomposing the problem into n binary tasks and then arriving at a final

decision by some sort of voting. Here also, feature selection can be optimized separately for each binary subtask. However, some $1-of-n$ induction algorithms do not execute binary decompositions, and require multi-class feature selection to choose a single set of features that perform well for the many classes.

(ii) Multi-label $(m-of-n)$ classification, where every individual case may belong to several, none, or even all classes. In the multi-label case, the difficulty is logically decomposed into n binary classification tasks: $class_i.vs.notclass_i$. These Individual binary tasks are solved independently where each may comprise its own feature selection to enhance its precision. And also a few $(m-of-n)$ applications programmable de novo require multi-class feature selection for performance and scalability reasons.

Other $(1-of-n)$ induction algorithms carry out a good deal of binary decomposition, e.g. algorithms finding optimal splitting hierarchies, or error-correcting code classifiers based on $o(n^2)$ dichotomies. In case of problems of this type, possibly we may carry out one multi-class feature selection rather than a separate binary feature selection for each dichotomy.

Theoretically all multi-class tasks could be performed with binary decompositions eliminating the requirement for multi-class feature selection. However, in reality lots of excellent software products APIs and libraries suppose the conversion of text into numerical feature vectors to be executed as a pre-processing step, and devoid of any capability for injecting feature selection into the inner loops, where the decompositions occur.

For instance, a centralized server task to classify millions of items on the network into multiple, orthogonal taxonomies, may be performed with more efficiency to establish a single, plausible sized feature vector to send through the network rather than sending individually to all the large documents.

For an application [41], of huge database of unstructured, multi-field (technical support) cases has memory by a cached, limited size feature vector representation for quick interactive examination, classification and labeling into multiple $(1-of-n)$ and $(m-of-n)$ taxonomies, where the classifiers are from time to time retrained in real time. It would be unfeasible to re-extract features for every binary decomposition or union of all the features into a exceptionally long feature vector that would be requested by all the binary feature selection sub tasks.

From various schemes of multi-class feature selection a few methods such as Chi-squared logically scale to multiple classes. However they face an underlying problem that is; an instance of a multi-class topic recognition case, with one of the classes holding all German texts. Now the German class will create many exceptionally predictive words. Almost all feature selection methods favor the stronger features and limit other classes for features. Similarly, if one class is mainly complicated, multi-class feature selectors will be inclined to disregard it, in view of the fact that it presents no strong features. These difficult classes require more features rather than fewer features.

A way out to this dilemma is to execute feature selection separately for each class through binary decompositions, and then to decide the final ranking of features using a round-robin algorithm where each class gets to vote its most preferred features in turn [41]. Since few classes are simpler to recognize than others this enhances performance even for well-balanced research benchmarks, however the difference results in most feature selection methods to be ignored, the very features that require most help. The aim of this scheme is to advance strength in atypical situations that arise only sporadically in practice that affects the average performance.

iii. *Hierarchical Classifications*

Hierarchy is one of the most predominant strategies for organizing abstractions. Hierarchical classification involves multiple tasks with the aim to classify items into a set of classes for organizing into a tree or directed acyclic graph, such as the Yahoo web directory. Here for some settings, the task is a single label problem to select $(1-of-n)$ nodes—or even limited to the leaf classes in the case of a 'virtual hierarchy.' For some other settings, the problem is of a multi-label task to select multiple interior nodes, optionally including all super-classes along the paths to the root.

Regardless of the given hierarchy of the classes, such issues are occasionally considered simply as flat multi-class tasks, either accommodating training examples up the tree structure for each class or a top-down hierarchy of classifiers may be generated to match the class hierarchy. The training set for each step down the tree is composed of all the training instances under each child subtree, optionally including a set of items positioned at the interior node itself, which terminates the recursion. Although this decomposition of classes is different from a flat treatment of the problem, in either decomposition, the same single-label or multi-label feature selection methods apply to the many sub-problems. It has been proposed that each internal hierarchical classifier may be quicker because dependency of each can be only for a few features (selected by feature selection), and can be further accurate because it only takes into account cases within a limited framework.

For instance an interior node concerning recycling with subtopics for glass recycling and can recycling there might be separate classifier intended for cases involving recycling. In this approach the training sets of every interior classifier are more balanced compared to a flat treatment of the problem.

iv. *Benchmarking Feature Selection Strategies*

The table 2.4 outlines latest research evaluating attributes selection techniques. The main research findings were,

(i) the filter method when implemented, information gain and chi-square have shown reasonably excellent performance (Yang and Pedersen (1997) [2] and the wrapper technique may have given even better performance if time had permitted in comparison to other similar measures.

(ii) Debole and Sebastiani (2003) [8] stated that gain ratio and chi-square outperformed information gain, iii) Forman (2003) [4] reported that information gain performed better than 10 other attribute selection methods in most experiments,

(iii) Halland Holmes 2003 [42] stated that wrapper method is very expensive for large datasets consisting of huge number of attributes.

Table 2.4 : Prior Studies on Attribute Selection Methods

References	attribute selection method	Outcome
Debole and Sebastiani 2003	χ^2, information gain, gr	GR and χ^2 outperformed Information Gain
Forman (2003)	accuracy, accuracy balanced, χ^2, document frequency, F1 measure, information gain, odds ratio numerator, odds ratio, pow, pr, rand	Information Gain outperformed other methods in most situations
Hall and Holmes (2003)	Correlation-based Feature Selection, information gain, wrapper, relief, consistency based, principle component	Wrapper was not applicable on the dataset with 1557 attributes due to time limitation. Wrapper and Correlation-Based Feature Selection outperformed other methods on the dataset with 293 attributes by NB.
Lewis and Ringuette	information gain	Both propBayes and DT-MIN 10 provided reasonable performance
Liu(2004)	information gain, mutual information, χ^2, odds ratio, simplified-chi-square	Information Gain an d χ^2 were most effective for NB. No benefit for SVM was found.
McCallum and Nigam (1998)	information gain	MNB outperformed NB in large attributes set
Madenic (1994)	information gain, odds ratio, word frequency, rand	Odds Ratio outperformed other methods
Ribone (2002)	information gain, word frequency, document frequency	Information Gain>Word Frequency>Document Frequency
Rogati and Yang (2002)	Document frequency, information gain χ^2	χ^2 outperformed other methods
Joachims (1996)	information gain	SVM outperformed other classifiers
Liu(2002)	χ^2, Correlation-based Feature Selection, mit correlation, entropy,	Entropy was the best, followed by χ^2, on the datasets. Correlation-based feature selection outperformed others on the ovarian cancer dataset.
Sebastiani (2002)		Summary of previous studies as {Odds Ratio, NGL coefficient, SS}>{ χ^2, Information Gain}>Mutual Information
Yang and Pedersen (1997)	Document frequency, information gain, mutual information, χ^2, term strength	Information Gain and χ^2 were most effective. Performance improved after attribute selection

Note 1: Abbreviations of attribute selection methods: ACC—Accuracy; ACC2—Accuracy balanced; BNS—Bi-Normal Separation; CFS—Correlation-based Feature Selection; χ2—chi-square; CNS Consistency-based; DF—document frequency; F1—F1 Measure; GSS—GSS coefficient (simplified chi-square); IG—information gain; MI—mutual information; NGL—NGL coefficient; ODDN—odds ratio numerator; OR—odds ratios; PC—Principal Components; POW—Power; PR—Probability Ratio; RAND—Random; RLF—Relief; TS—term strength; WF—word frequency; WRP—Wrapper.

Note 2: Abbreviations of classification methods: C4.5—decision tree; DT-min10—decision tree; kNN—k-Nearest Neighbors; LLSF—Linear Least Squares Fit; LR—Logistic Regression; NN—Neural Network; NB—Naïve Bayes; MNB—Multinomial Naïve Bayes; PCL—Prediction by Collective Likelihood; Prop Bayes—Bayesian classifier; SVM—Support Vector Machine.
Note 3: "$>$" means "performed better than".

d) Combination of Feature Selection Methods

Several researches have been done to enhance the efficiency of feature selection strategies on text categorization; however they generally are about improving the performance of the individual feature selection methods. The success of a feature selection method is determined by various variables and it is very difficult to understand which method is better performer than others though several selection methods are prevalent. So the combination of distinct feature selection methodologies gives more efficiency in text classification. The output of classifiers combination is a potential strategy and it is being studied extensively in the area of information retrieval. This section covers strategies for combining the outputs of the individual feature selections metrics.

i. *The Common Combining Strategies*

There are several strategies of combining the outputs of the different FS methods. Some combination strategies or most popular approaches to combine various outputs are;

1. Linear Combining Methods such as
   (i) Averaging and
   (ii) Weighted Averaging
2. Non-Linear Combining Methods such as
   (i) Ranking and
   (ii) Voting

Averaging (Tumer and Ghosh, 1999) [43], a Linear Combining method is the most frequently used combining strategy. Fox and Shaw, 1994, significantly state that most excellent combining strategy depends on adding the outputs of the algorithms that is similar to averaging on comparing six combining strategies [44]. Also Hull et al., 1996, [45] show that the accuracy of the simple averaging strategy is far better than the complex combinations of the classifiers.

With respect to the above studies and considering the results, we decide to apply the averaging strategy in two ways: *(1-a) Score Combination and (1-b) Rank Combination.*

In the Score and Rank Combination Strategy, the preceding research finds that the efficiency of the combination and the number of feature selection methods considered in combination are inversely related and vice versa. Also it is reported that maximum efficiency is achieved essentially by the combination of two feature selection methods [33, 46]. So we only choose combining two distinct feature selection methods. In the paper, we assess the performance of all potential binary-combinations (2-combinations) of five different feature selection methods: DF, TF-IDF, CHI, IG, Acc2 and metrics which are TF-IDF & CHI, TF-IDF & IG, TF-IDF & Acc2, TF-IDF & DF, CHI & IG, CHI & Acc2, CHI & DF, IG & Acc2, IG & DF and Acc2& DF.

The strategy of the feature selection methods is of two steps. In the first step score is given to the terms that are more relevant for classification and in the second step is of selecting the terms with highest score. In scoring stage, the different feature selection methods and their scores of each term are normalized using the maximum and minimum scores according to the below formula:

Score $= (s1, s2, \ldots, sn)$ where si score of the i^{th} term, n is the total number of term.

$$score^{1} = \frac{Score - \min(Score)}{\max(Score) - \min(Score)}$$

By normalization, the scores fall in the same range [0, 1] and scores of the terms from the different feature selection methods are represented equally which facilitates efficient comparisons between the methods.

(1-a) Score Combination

Score combination is averaging the normalized term scores of the different feature selection methods.

$$c_{score} = \sum_{i-1}^{M} \frac{Score_i^{'}}{M}$$

where M is the number of feature selection methods which is 2 for this study.

(1-b) Rank Combination

Rank Combination is averaging the term ranks collected from the term scores of the different feature selection methods.

Rank $= (r1, r2, \ldots, rn)$ where ri rank of the i^{th} term, n is the total number of term.

$$c_{rank} = \sum_{i=1}^{M} \frac{Rank_i}{M}$$

There are several modes for determining rankings like i) Standard Competition Ranking, ii) Modified Competition Ranking, iii) Dense Ranking, iv) Ordinal Ranking and v) Fractional Ranking. We rank the terms in our research as per the descending order of their scores and with standard competition ranking strategy. Here in competition ranking ("1, 2, 2, 4" ranking), terms with similar score are assigned the same ranking number and next a gap is given in the ranking numbers.

ii. *Proposed Combinations of Feature Selection Methods*

We present seven latest methods for the efficiency in combining the different metrics for feature selection and compare them as discussed below;

In the seven proposed combinations the first and second combinations are comparable to score combination and are a variation of the score combination. The remaining five combinations are a product combination of both score and rank value of the terms.

a. *C1 Combination -- Logarithmic Combination*

The first method, the logarithmic combination is based on the score combination that is simply averaging the term scores of the feature selection methods. The principle applied is the same where in the proposed combination the logarithmic scores of the terms are averaged in place of term scores. The principal involved is increasing the interval between the highest and the lowest scores, taking benefit of the logarithm. The interval between the scores increases as the scores decrease and it decreases as the scores increase. The natural logarithm is used as a logarithm function where if the value of the score is zero, then it will substitute the value with 0.00001 for computing the logarithm. The calculation of the C1 Combination is given by the following formula:

$$C_1 = \sum_{i=1}^{M} \frac{\ln(score_i^{'})}{M}$$

b. *C2 Combination – Square Combination*

In the second method, the Square Combination proposed is also based on the score combination, however the term scores are squared and averaged rather than only averaging the term scores. The interval between the scores exponentially increases rather than the interval between the highest scores and lowest scores that remains unchanged with the increases of scores. The difference between the scores increases as the scores increase and decreases as the scores decrease in contrast to preceding methods. Thus, if the term is considered essential for classification, this method elevates the importance of the term compared to the remaining terms prior to combination. The formula used is:

$$C_2 = \sum_{i=1}^{M} \frac{(Score_i^{'})^2}{M}$$

c. *C3 Combination – Product Combination with Fraction*

The third method, the product combination with fraction (C3) initially the rank of the term is multiplied with the scores of the terms. Next the outputs of the multiplication of the individual feature selection methods are added and divided with one for combining the outputs of the feature selection metrics. In case the value of score is equal to 1, then it will substitute the value with 0.99999 to prevent division by zero. The highest value terms are selected in feature selection step. The formula is:

$$C_3 = \frac{1}{\sum_{i=1}^{M} Rank_i^{x}(1 - Score_i^{'})}$$

When we assess the effectiveness of the combinations in classification we see that both score and rank combinations of the feature selection methods enhance the performance of the individual methods. This led us to consider using both rank and score values in the same function. The central idea of the "product combinations" in 1, 2, 3, 4, 5, 6 and 7 is to take benefit of the score and rank of the terms while identifying the most effective terms. Here both rank of the term and the score of the term have equal weight in the combination. Based on this principle, if the scores of terms of different feature selection methods are equal then greater importance is given to the term with highest rank. Different versions of the product combination are derived and are described below.

d. *C4 Combination – Product Combination with Fraction*

The fourth method, the product combination with fraction (C4) is different from the above proposed third method. In this method the logarithm of the rank is multiplied with the square of the score of the term. The formula of the C4 Combination is given as below:

$$C_4 = \frac{1}{\sum_{i=1}^{M} \ln(Rank_i) x (1 - Score_i^{'})^2}$$

e. *C5 Combination - Product Combination with Fraction*

The fifth method, the product combination with fraction, is different from the third method discussed above. In this method the square root of the rank is multiplied with the square of the score of the term. The formula is:

$$C_5 = \frac{1}{\sum_{i=1}^{M} (Rank_i)^{1/2} \times (1 - Score_i^{'})^2}$$

f. *C6 Combination - Logarithmic Product Combination*

The sixth method, the logarithmic product combination is sum of the products of each terms logarithm of the rank and the logarithm of the score of each feature selection method. The formula is:

$$C_6 = \sum_{i=1}^{M} \ln(Rank_i) x \ln(Score_i^{'})$$

g. *C7 Combination - Product Combination*

The seventh method, the product combination or C7 Combination is the last method that is the multiplication of the square root of the rank with the logarithm of the score of the term and addition of the outputs. The formula is:

$$C_7 = \sum_{i=1}^{M} (Rank_i)^{1/2} x \ln(Score_i')$$

IV. Conclusion

Feature Selection [47] continually handles huge and complex datasets and faces typical problems of feature space, which if very high, increases computational costs and also the training time.

Feature selection relevance has decreased with the constant developments in accuracy and scalability of core machine learning algorithms. An algorithm for instance, when considering more than 800,000 text cases was developed by Joachims that is an innovative linear SVM classifier. Using a 3.6GHz PC processor [18], it may be trained with approximately 50,000 word features in less than 3 minutes. As in some cases where feature selection does not improve the accuracy of the training sets, researchers interested in approaches other than feature selection can avoid the problems associated with feature selection and go for an input representation that is fixed and conveniently replicable.

However a case where a researcher of data mining is provided with a training set for generating an excellent possible classifier should definitely consider feature selection. The method can improve certainly Accuracy for certain datasets or at least give slight improvements on average. Thus, feature selection still has a role to play for those who seek to maximize Accuracy, e.g. industrial practitioners, application programmers and contestants in data-mining competitions.

References Références Referencias

1. F. Sebastiani, "Text categorization", Alessandro Zanasi (ed.) Text Mining and its Applications, WIT Press, Southampton, UK, pp. 109-129, 2005.
2. Wei Zhao "A New Feature Selection Algorithm in Text Categorization "International Symposium on Computer, Communication, Control and Automation 2010.
3. Y. Yang, J.O. Pedersen, "A comparative study on feature selection in text categorization", Proceedings of the 14th International Conference on Machine Learning, pp. 412–420, 1997.
4. G. Forman, "An extensive empirical study of feature selection metrics for text classification", Journal of Machine Learning Research, Vol. 3, pp. 1289–1305, 2003
5. Sebastiani F (2002). Machine Learning in Automated Text Categorization, ACM Computing Surveys, 34(1): 1-47.
6. Jensen R (2005). Combining rough and fuzzy sets for feature selection, PhD Thesis, University of Edinburgh, UK.
7. KjerstiAas and Line Eikvil "Text Categorization: A Survey" Report No. 941. ISBN 82-539-0425-8. , June, 1999.
8. Debole, F. And F. Sebastiani, "Supervised Term Weighting for Automated Text Categorization", Proceedings of SAC-03-18th ACM Symposium on Applied Computing, ACM Press, pp. 784–788, 2003.
9. Wanas, N., D. A. Said, N. H. Hegazy and N. M. Darwish, "A Study of Local and Global Thresholding Techniques in Text Categorization", Proc. Fifth Australasian Data Mining Conference, 2006.
10. D. Mladenic and M. Globelnik. Word sequences as features in text learning. In Proceedings of the 17th Electrotechnical and Computer Sci-ence Conference (ERK98), Ljubljana, Slovenia, pages 145–148, 1998.
11. 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, TR WS-98-05, pages 41–48. AAAI Press, 1998.
12. S. Das. Filters, wrappers and a boosting-based hybrid for feature selection. In International Conference on Machine Learning, 2001.
13. George A. Miller and Edwin B. Newman. Tests of a statistical explanation of the rank-frequency relation for words in written English. American Journal of Psychology, 71:209-218, 1958.
14. Avrim, L.B., and Pat, L. "Selection of Relevant Features and Examples in Machine Learning.," Artificial Intelligence (97:1-2) 1997, pp 245-271.
15. D. Koller and M. Sahami. Toward optimal feature selection. In International Conference on Machine Learning, pages 284–292, 1996.
16. DunjaMladenic and Marko Grobelnik. Feature Selection for Unbalanced Class Distribution and Naïve Bayes. In Proceedings of the Sixteenth International Conference on Machine Learning (ICML), pages 258-267, 1999.
17. Yiming Yang and Xin Liu. A Re-examination of Text Categorization Methods. In Proceedings of the Twenty-Second International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), pages 42-49, 1999.
18. T. Joachims. Training linear SVMs in linear time. In Proc. of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 217–226, 2006.
19. I. Guyon and E. Elisseef, A. Special issue on variable and feature selection. J. of Machine Learning Research, 3:1157–1461, 2003.
20. Nejad MB, Attarzadeh I, Hosseinzadeh M (2013). An Efficient Method for Automatic Text Categorization",

International Journal of Mechatronics, Electrical and Computer Technology, 3(9): 314-329.

21. Su-JeongKo and Jung-Hyun Lee "Feature Selection Using Association Word Mining for Classification "H.C. Mayr et al. (Eds.): DEXA 2001, LNCS 2113, pp. 211–220, 2001.
22. AnirbanDasgupta "Feature Selection Methods for Text Classification "KDD'07, August 12–15, 2007.
23. C.D. Manning, P. Raghavan, H. Sch¨utze, Introduction to Information Retrieval, Cambridge, Cambridge University Press, 2008.
24. Yang,Y. (1999).An evaluation of statistical approaches to text categorization. Information Retrieval, 1(1–2), 69–90.
25. Chih, H.,&Kulathuramaiyer,N. (2004). An empirical study of feature selection for text Categorization based on term weightage. In Proceedings of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence (pp. 599–602).Washington, DC: IEEE.
26. Dave K (2011). Study of feature selection algorithms for text-categorization, University of Nevada, Las Vegas, UNLV Theses/Dissertations/Professional Papers/ Capstones, Paper 1380.
27. S. Gunal, O.N. Gerek, D.G. Ece, R. Edizkan, "The search for optimal feature set in power quality event classification", Expert Systems with Applications, Vol. 36, pp. 10266–10273, 2009.
28. D.E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Reading, Massachusetts, Addison-Wesley, 1989.
29. C.L. Huang, C.J. Wang, "A GA-based feature selection and parameters optimization for support vector machines", Expert Systems with Applications, Vol. 31, pp. 231–240, 2006.
30. J. Yang, V. Honavar, "Feature subset selection using a genetic algorithm", IEEE Intelligent Systems, Vol. 13, pp. 44–49, 1998
31. P. Soucy and P. Mineau. A simple feature selection method for text classification. In Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, pages 897–902, 2001.
32. Tasci, S., "An evaluation of existing and new feature selection metrics in text categorization", Computer Engineering, Bogaziçi University, 2006
33. Li, Y., D. F. Hsu and S. M. Chung "Combining Multiple Feature Selection Methods for Text Categorization by Using Rank-Score Characteristics", International Conference on Tools with Artificial Intelligence - ICTAI, pp. 508-517, 2009
34. E. Wiener, J. O. Pedersen, and A. S. Weigend. A neural network approach to topic spotting. In SDAIR'95: Proceedings of the 4th Symposium on Document Analysis and Information Retrieval, pages 317–332, 1995.
35. S. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In CIKM'98: Proceedings of the 7th international conference on Information and knowledge management, pages 148–155, New York, NY, USA, 1998. ACM.
36. Y. H. Li and A. K. Jain. Classification of text documents. The Computer Journal, 41:537–546, 1998.
37. Liu, H., and Setiono, R. "Chi2: Feature Selection and Discretization of Numeric Attributes," proceedings of the Seventh International Conference on Tools with Artificial Intelligence, 1995.
38. M. F. Caropreso, S. Matwin, and F. Sebastiani. A learner-independent evaluation of the usefulness of statistical phrases for automated text categorization. In A. G. Chin, editor, Text Databases and Document Management: Theory and Practice, pages 78–102. Idea Group Publishing, Hershey, US, 2001.
39. C. Lee, G.G. Lee, "Information gain and divergence-based feature selection for machine learning-based text categorization", Information Processing and Management, Vol. 42, pp. 155–165, 2006.
40. Quinlan, J. (1986). Induction of decision trees. Machine Learning, 1(1), 81–106.
41. G. Forman. A pitfall and solution in multi-class feature selection for text classification. In ICML '04: Proc. of the 21st Int'l Conf. on Machine learning, pages 297–304. ACM Press, 2004.
42. Hall, M.A., and Holmes, G. "Benchmarking Attribute Selection Techniques for Discrete Class Data Mining," IEEE Transactions on Knowledge and Data Engineering (15:6) 2003, pp 1437-1447.
43. Tumer, K. and J. Ghosh, "Linear and order statistics combiners for pattern classification", Combining Artificial Neural Networks, Springer Verlag, pp. 127–162, 1999.
44. Fox, E. and J. Shaw, "Combination of multiple searches", Proceedings of the 2nd Text Retrieval Conference (TREC-2), National Institute of Standards and Technology Special Publication 500-215, pp. 243–252, 1994
45. Hull, D., J.Pedersen and H. Schutze, "Method combination for document filtering", Proceedings of the 19th ACM SIGIR Conference on Researchand Development in Information Retrieval, pp.279–287, 1996.
46. Olsson, J. S. and D. W. Oard, "Combining Feature Selectors for Text Classification", CIKM'06, 2006.
47. A. Dasgupta, P. Drineas, B. Harb, "Feature Selection Methods for Text Classification", KDD'07, ACM, 2007.
48. Jensen R (2005) Combining rough and fuzzy sets for feature selection, PhD thesis for University of Edinburgh UK.