Received 1 August 2015
Revised 10 February 2016
Accepted 13 July 2016

A novel committee selection mechanism for combining classifiers to detect unsolicited emails

Shrawan Kumar Trivedi
Department of Information Systems and Business Analytics, School of Management, BML Munjal University, Gurgaon, India, and
Shubhamoy Dey
Department of Information Systems, Indian Institute of Management Indore, Indore, India

Abstract

Purpose – The email is an important medium for sharing information rapidly. However, spam, being a nuisance in such communication, motivates the building of a robust filtering system with high classification accuracy and good sensitivity towards false positives. In that context, this paper aims to present a combined classifier technique using a committee selection mechanism where the main objective is to identify a set of classifiers so that their individual decisions can be combined by a committee selection procedure for accurate detection of spam.

Design/methodology/approach – For training and testing of the relevant machine learning classifiers, text mining approaches are used in this research. Three data sets (Enron, SpamAssassin and LingSpam) have been used to test the classifiers. Initially, pre-processing is performed to extract the features associated with the email files. In the next step, the extracted features are taken through a dimensionality reduction method where non-informative features are removed. Subsequently, an informative feature subset is selected using genetic feature search. Thereafter, the proposed classifiers are tested on those informative features and the results compared with those of other classifiers.

Findings – For building the proposed combined classifier, three different studies have been performed. The first study identifies the effect of boosting algorithms on two probabilistic classifiers: Bayesian and Naïve Bayes. In that study, AdaBoost has been found to be the best algorithm for performance boosting. The second study was on the effect of different Kernel functions on support vector machine (SVM) classifier, where SVM with normalized polynomial (NP) kernel was observed to be the best. The last study was on combining classifiers with committee selection where the committee members were the best classifiers identified by the first study i.e. Bayesian and Naïve bays with AdaBoost, and the committee president was selected from the second study i.e. SVM with NP kernel. Results show that combining of the identified classifiers to form a committee machine gives excellent performance accuracy with a low false positive rate.

Research limitations/implications – This research is focused on the classification of email spams written in English language. Only body (text) parts of the emails have been used. Image spam has not been included in this work. We have restricted our work to only emails messages. None of the other types of messages like short message service or multi-media messaging service were a part of this study.

Practical implications – This research proposes a method of dealing with the issues and challenges faced by internet service providers and organizations that use email. The proposed model provides not only better classification accuracy but also a low false positive rate.

VINE Journal of Information and Knowledge Management Systems
Vol. 46 No. 4, 2016
pp. 524-548

2059-5891
DOI 10.1108/VJIKMS-07-2015-0042

Originality/value – The proposed combined classifier is a novel classifier designed for accurate classification of email spam.

Keywords SVM, Bayesian, Probabilistic classifiers, Naïve Bayes, Function-based classifiers, Kernel functions, Combining classifiers, Stacking, Committee machine

Paper type Technical paper

1. Introduction

In the present automated world, sharing information is important to be competitive and sustainable in business. Email is a rapid and inexpensive medium of communication. It is a popular medium of interaction between people and has become a part of life itself (Whittaker and Moody, 2005). However, spam (unsolicited bulk email) has become a nuisance in such communication. Recently, interest of researchers has increased in the spam classification domain as its bulk is increasing day by day. A study observes that 66 per cent of all business emails are spam (Kaspersky Spam Statistics, 2014). This rapid growth leads to serious problems such as unnecessary filling of users' mailboxes, engulfing of important emails, consuming storage space and bandwidth and requiring time to sort them.

Legal and other simplistic methods like blacklisting, keyword-based filtering, etc., have shown limited effect in countering such problems. However, content-based filtering using machine learning methods is reported to be promising in literature.

Nowadays, spam classification is becoming a challenging area due to the complex nature of the spam. Complexity is defined as the modifications of content, such as tokenization (modifying words such as "free" being written as f r 3 3) and obfuscation (which hides feature by adding HTML or some other codes such as "free" coded as fr&#101xe or FR3E) (Heydari *et al.*, 2015; Goodman *et al.*, 2007), etc., to change the information of features so as to create barriers in distinguishing spam from legitimate emails. Many machine learning classifiers have been tested in an attempt to tackle these problems. Some of them, such as probabilistic classifiers [Bayesian (Koller and Sahami, 1997a; Jatana and Sharma, 2014) and Naïve Bayes (NB) (Farid *et al.*, 2014; Lewis and Gale, 1994)] and support vector machine (SVM) (Drucker *et al.*, 1999), have been found to be good performers in literature. Significantly good accuracy has been reported even in the presence of the complexity discussed above. The Bayesian technique is well known, as it has the interesting concept of finding the informative features/words with the help of deviation from the mean.

The first part of this research is on probabilistic classifiers (Bayesian and NB) and the concept of boosting [Bagging, Boosting (with re-sampling) and AdaBoost] to improve data sampling for better learning during training. Boosting methods use voting mechanism where a single classifier is formulated as the linear combination of many weak classifiers.

For SVM classifiers, a good choice of kernel function is important and at the same time difficult. A good kernel function provides efficient learning to the SVM during training. In the second part of our study, a number of different kernel functions have been compared.

In the final part of this research, a method for combining classifiers with committee selection has been proposed where the individual classification decisions of the good classifiers are combined. The committee members are chosen from the best performers

of the first part of the study, and the committee president is selected from the second part of the study.

Genetic feature search method is used to select an informative feature subset. The concerned classifiers are tested on the informative feature subsets extracted from three different publicly available data sets (Enron, SpamAssassin and Lingspam).

2. Related work

A number of researches have been conducted in the area of classification. However, this research concentrates only on spam classification that has turned into a critical area of research in recent years. This study considers machine learning classifiers and possible ways of improving the learning capability of classifiers in a supervised learning environment. A variety of research has been reported in literature where different classifiers have been tested on different email data sets. Classifiers such as probabilistic classifiers (i.e. Bayesian and Naive Bayes), Boosting algorithms (i.e. Bagging, Boosting and AdaBoost), function-based classifiers (i.e. SVM with different Kernel function) and combining of classifiers with committee selection have been considered in this research.

The Bayesian classifier was introduced by Lewis (1998). This method involves traditional text mining approaches based on content and domain knowledge about the documents. Androutsopoulos *et al.* (2000a, 2000b, 2000c) have done a series of studies to justify the use of an extended version of NB classifier that was initially proposed by Sahami *et al.* (1996). These researches show the consequence of using different number of features and the size of training set on the classifier performance. They have done a comparative study of NB with memory-based classifiers. Their results have established the strength of probabilistic classifiers.

A common problem of misclassification seen in machine learning classifiers happens due to poor sampling of the data set during training stage. Boosting algorithms attempt to solve this problem with the help of a voting mechanism and have therefore found a prominent place in the literature (Heydari *et al.*, 2015; Carreras and Màrquez, 2001; Zhang *et al.*, 2004; Korada *et al.*, 2012).

SVM has been identified to be a strong classifier and has a respectable place in classification literature. Drucker *et al.* (1999) have done a comparative study where the performance of SVM was compared with various machine learning classifiers. The result of this study was in favour of SVM and boosted decision trees in terms of accuracy and speed. However, the training time of boosted decision tree was longer than that of SVM. Another study conducted by Woitaszek *et al.* (2003) uses a simple SVM and a personalized dictionary for detecting commercial emails. Microsoft Outlook XP has integrated an SVM-based spam detection system plug-in for its users.

Recently, the interest of researchers has moved towards ensemble based and combined classifiers to improve sampling of the data set and learning of the classifiers. The concept of stacking, which is a way of combining classifiers' decisions, was developed by Wolpert (1992). In a combined classifier, the individual decisions of multiple classifiers are combined together to get the final classification decision (Dietterich, 1997). Committee selection is one of the methods of combining classifiers, where multiple classifiers participate as members of the committee and a single classifier works as the president of that committee. This method was suggested by Sakkis *et al.* (2001). In their work, NB and *k*NN classifiers were the committee members, and a second *k*NN was the president. Their experiment demonstrates that members

made very dissimilar errors, and in less than 2 per cent instances, both members were simultaneously wrong. In a situation where one of the members was wrong, the president could take the decision by selecting or ignoring the members' decisions.

3. Structure of the spam filter

3.1 Data sets

This study uses three publicly available data sets, taken from three different sources. Our main analysis is done with "Enron email" data set, and thereafter "SpamAssassin" and "LingSpam" data sets are used for validation of the results. A description of each data set is given below.

3.1.1 Enron email data set. In this study, of the six existing versions of Enron email data set (Enron, 2004), Enron versions 3, 4, 5 and 6 have been selected to create 6,000 legitimate (ham) and 6,000 unsolicited (spam) files by simple random sampling. The spam files have been produced from Enron versions 4, 5 and 6. To obtain the same number of ham files, Enron version 3 has also been used along with versions 4, 5 and 6. The rationale of selecting only these versions was that the complexities present in the spam files of those versions would help in finding the actual strength of the classifiers against different types of attacks by spammers.

3.1.2 SpamAssassin. SpamAssassin (2005) data set is another data set used in this research. This data set includes some older as well as some relatively recent unsolicited emails (spam) developed by some non-spam-trap sources. Of the entire set of spam emails, 2,350 spam files have been sampled for this research. In addition, this data set has some easy to identify and some difficult to identify legitimate (ham) emails. Both easy and difficult emails are sampled to generate 2,350 ham files. Easy emails can be identified and classified by straightforward techniques, whereas difficult ham emails are affected by some attacks such as use of HTML, unusual HTML mark-up, coloured text, "spammish-sounding" phrases, etc., making it difficult to distinguish them from spam emails.

3.1.3 LingSpam. This third data set has been built from the LingSpam corpus (Ling-Spam, 2000; Androutsopoulos, 2000a). This corpus includes four different version of email files, i.e. bare (Lemmatiser disabled, stop-list disabled), lemm (Lemmatiser enabled, stop-list disabled), lemm_stop (Lemmatiser enabled, stop-list enabled) and stop (Lemmatiser disabled, stop-list enabled). This study makes use of 478 unsolicited (spam) email files and an equal number, 478 legitimate (ham) email files gathered from all versions of the corpus. The detail of these emails are as follows:

- *478 Linguist Ham emails* – gathered by randomly downloading digests from the archives, separating their messages and removing text added by the list's server; and
- *478 Spam emails* – attachments, HTML tags and duplicate spam messages were not included in the files.

3.2 Pre-processing of the data set

An email file is modeled as a collection of feature vectors a_k^i, defined as the weight of word i belonging to the document k (Aas and Eikvil, 1999). These feature vectors and email files together constitute a matrix, called the term-document matrix (TDM), for representing the feature space. This matrix appears high dimensional and sparse in nature due to the presence of a large number of features in the representation. However,

this problem is tackled by "dimensionality reduction" that is done before the classification and with the use of "feature selection" or "feature extraction" processes. Dimensionality reduction includes "stop-word" (terms that contain no information such as pronouns, prepositions and conjunctions) removal (Aas and Eikvil, 1999) and "lemmatisation" (grouping the terms that contain same information such as "combine, combined, and combining", etc.).

3.3 Feature selection

Feature selection is applied after dimensionality reduction of the matrix. A number of feature selection methods have been discussed in the literature. In general, these methods help to find informative features from the set of available features. For cost sensitive evaluation, this research uses a evolutionary feature subset search method known as "Genetic subset search" (Trivedi and Dey, 2014; Haleh and Imam, 1994) to identify a small subset of informative features.

3.3.1 Genetic feature search method. This study focuses on cost sensitive evaluation where genetic subset search method plays a crucial role for selecting a small numbers of informative features. This algorithm initially introduced by Holland (1975) works on an inductive learning approach. It works in a way similar to the genetic models of the natural systems and is therefore called Genetic algorithm. In the beginning, this method starts with a population of individuals to search a given space. Each individual of the population is evaluated for their fitness. New individuals, called "offspring", are produced by selecting best performing individuals (Oreski and Oreski, 2014). The offspring retains the features of their parents and generate a population with improved fitness. This process is performed by two genetic operators, i.e. "Crossover" and "Mutation". Crossover operates by random selection of a point in two-parent gene structures and develops two new individuals by exchanging the remaining part of parents. Hence, this operator generates two new offspring combining the two parent individuals. The Mutation operator creates a new individual by arbitrarily altering some gene component of an old individual. The work of this operator is similar to a population perturbation operator which amounts to introducing new information in the population. This operator may also help to avoid stagnation which can arise during the search process.

3.4 Machine learning classifiers

3.4.1 Boosting algorithms. The idea of boosting (Heydari *et al.*, 2015; Duda *et al.*, 2001; Efron, 1982) was developed from bootstrapping. The basic phenomenon of bootstrapping is to re-assess the accuracy of some estimate. It is a statistical sample-based method where samples are taken by drawing randomly with replacement from a data set. Some boosting algorithms have been shown to strengthen the accuracy of classifiers.

3.4.1.1 Bagging. Bagging technique (Heydari *et al.*, 2015; Duda *et al.*, 2001; Breiman, 1996; Hastie, 2001) is also based on the concept of bootstrapping which helps in improving the learning of classifiers during training. The main idea of this technique is to take an aggregation of bootstraps. Let us assume the training set $T_r = t_1, t_2, t_3 \ldots t_n$ with $t_i = (x^i, y^i)$ where $x^i \in R_n$ and $y^i \in \{+1, -1\}$ which is defined as the particular class for i^{th} training sample. In this research, $+1$ is denoted as legitimate emails (ham) and -1 is considered as unsolicited email (spam). This algorithm fits the regression model

which formulates a prediction f^x at input x. It averages prediction over the collection of bootstraps for reducing variance and increasing accuracy. This algorithm fits our model with the given prediction f^x_{bag} for each bootstrap sample, where $b \in 1, 2, 3, \ldots B$. Bagging estimation can be shown as:

$$f^x_{bag} = \frac{1}{B}\sum_{b=1}^{B} f^x_b \tag{1}$$

Algorithm for classification

Input: Training set $T_r = t_1, t_2, t_3 \ldots t_n$ with $t_i = (x^i, y^i)$. Number of sample version of training set B.
Output: An appropriate classifier G^x_t for above training set.

For $n = 1, 2, 3, \ldots B$ (where n is the number of classifiers for the ensemble):

- Draw $K_x \leq N_x$ samples *with* replacement from the training outset T_r and to obtain n^{th} sample T^n_r.
- Take the classifier G^n_t to train it from each training sample T^n_r in each iteration $n = 1,2, 3, \ldots B$.
- Build up the final classifier G^n_t with number of ensemble classifier ($n = 1, 2, 3, \ldots B$). The final classifier is the combined aggregate result of all classifiers that participated in ensemble.

$$G^x_t = sign\left(\sum_{n=1}^{B} G^n_t\right) \tag{2}$$

3.4.1.2 Boosting (Boost with re-sample). Boosting technique works in the same way as bootstrapping and bagging. The only difference is in the sample selection, while bootstrapping and bagging perform sampling with replacement, boosting performs sampling without replacement. This technique was proposed by Schapire (1989).

Algorithm for classification

Input: Training set $T_r = t_1, t_2, t_3 \ldots t_n$ with $t_i = (x^i, y^i)$. Number of sample version of training set B.
Output: An appropriate classifier G^x_t for above training set.

- Draw $K^1_x < N_x$ samples *without* replacement from the training outset T_r and obtaining training sample T^1_r. Take weak classifier G^1_t and train it for each sample T^1_r. Each individual classifier G^1_t is trained with the help of training sample T^1_r that is draw at random without replacement and hence G^1_t is to be considered as weak classifier.
- Select $K^2_x < N_x$ samples from the training outset T_r which include half misclassified samples by weak classifier G^1_t. Train weak classifier G^2_t on new samples.
- Finally, select remaining samples misclassified by G^1_t and G^2_t. Train weak classifier G^3_t on remaining samples.
- Build up the final classifier as a vote of weak classifiers.

$$G_t^x = sign\left(\sum_{n=1}^{3} G_t^n\right) \tag{3}$$

3.4.1.3 Adaptive boosting (AdaBoost). Adaptive boosting (AdaBoost) (Heydari *et al.*, 2015; Hastie, 2001; Freund and Schapire, 1996) works differently from other boosting methods and uses re-weight technique rather than simple random sampling. This technique is based on the idea of building an ensemble of classifiers for performance boosting with better learning. AdaBoost learns with the set of outputs M_x of weak classifiers $G_t^{m_x}$ and then combines the decision for the final classifier G_t^x.

Algorithm for classification

Input: Training set $T_r = t_1, t_2, t_3 \ldots t_n$ with $t_i = (x^i, y^i)$. Number of sample version of training set B.

Output: An appropriate classifier for the training set G_t^x.

(1) Initialize the weights $w_i^t = 1/N, i \in 1, 2, 3, \ldots N$, where N is the number of examples in the training set, chosen according to the information gain calculated for the each feature and w_i^t is a randomly assigned weight assigned to each of the examples.

(2) From $m = 1, 2, 3, \ldots M_x$, where M_x is the number of individual classifiers that participated in ensemble generation.

- Train the weak classifier $G_t^{m_x}$ on the training sample that is taken from the training outset using weights w_i^t.
- Calculate the error term $E_{rror}^m = \Sigma_{i=1}^N w_i^t I(y_i \neq G_t^{m_{xi}}) / \Sigma_{i=1}^N w_i^t$, which is defined as the percentage error calculated after the classification with the individual weak classifier.
- Calculate weight contribution $\theta_m = 0.5\log(1 - E_{rror}^m / E_{rror}^m)$, it is calculated to reassign the value to the instances for the successive iterations.
- Substitute $w_i^t \leftarrow w_i^t Exp(-\theta_{(m)} I(y_i \neq G_t^{m_{xi}}))$ then re-normalize $\Sigma_i w_i^t = 1$.

(3) The final classifier is the combined decision of all weak classifiers and represented as:

$$G_t^x = \theta_m sign\left(\sum_{m=1}^{M_x} G_t^{m_x}\right) \tag{4}$$

3.4.2 Probabilistic classifiers. This idea was proposed by Lewis (1998), who introduced the term $P(c_i/d_j)$ and defined it as the probability of a document represented by a vector $d_j = w_j^1, w_j^2, \ldots, w_j^n$ of words falling within a certain category c_i. This probability is calculated by the Bayes theorem:

$$P\left(\frac{c_i}{d_j}\right) = \frac{P(c_i) * P\left(\frac{d_j}{c_i}\right)}{P(d_j)} \tag{5}$$

where $P(d_j)$ symbolizes the probability of arbitrarily selected documents represented by the documents vector d_j, and $P(c_i)$ is the probability of arbitrary selected documents d_j falling in a particular class c_i. This classification method is usually known as "Bayesian Classification".

Bayesian is a popular technique, but it is challenging in the case of high dimension of the data vector d_j. This challenge is tackled by developing an assumption that any two arbitrarily selected coordinates of document vector d_j (tokens) are independent to each other. This assumption is represented by the given equation:

$$P\left(\frac{d_j}{c_i}\right) = \prod_{l=1}^{n} P\left(\frac{w_j^l}{c_i}\right) \tag{6}$$

This above assumption is the foundation of "Naive Bayes" classifier which is popular in the area of the text mining (Heydari *et al.*, 2015; Joachims, 1998; Koller and Sahami, 1997b; Larkey and Croft, 1996).

3.4.3 Support vector machine. SVM is another popular machine learning classifier. It takes inspiration from statistical learning theory and structural minimization principle (Vapnik, 1995). It is one of the best accepted classifiers due to its strength in dealing with high dimensional data with unique kernel function. A set of data is said to be high dimensional when each entity (i.e. document) is represented by a vector with a large number of dimensions (i.e. features/terms in the context of text representations). In such situations, determining the correct classes for the document vector can become both difficult and computationally inefficient.

The basic concept of SVM is to separate the classes (i.e. positive and negative) by the use of maximum margin produced by a hyperplane. Let us take a training sample $X = x^i, y^i$, where $x^i \in R_n$ and $y^i \in +1, -1$, which is defined as the particular class for i^{th} training sample. In this research, $+1$ is denoted as legitimate emails (ham) and -1 is unsolicited emails (spam). Final output of the classifier is determined by the following equation:

$$y = w \,.\, x - b \tag{7}$$

where y indicates final output of classifier, w termed as normal vector analogous to those in the feature vector x and b is the bias parameter that is determined by the training procedure. The following optimization function is used to maximize the separation between classes:

$$\text{minimize} \quad \frac{1}{2}\|w\|^2 \tag{8}$$

$$\text{subject to} \quad y_i(w \,.\, x - b) \geq 1, \forall i \tag{9}$$

3.4.3.1 Kernel functions. Sometimes SVM classifiers are unable to separate the input data into specific classes due to the poor sampling of the data set. This problem is resolved by transforming the high dimensional input data using some non-linear transformation functions. This process helps to separate the input data in such a manner that a linear separable plane can be revealed in the transformed space. On the other hand, high dimensionality of the feature space makes the computation of

the inner product of two transformed vectors practically infeasible. For resolving this problem, "Kernel Functions" are introduced that are used in place of the inner product of two transformed data vectors in the feature space (Trivedi and Dey, 2013a). For viable operations, the computational effort is reduced by the use of appropriate kernel functions.

3.4.3.2 Kernel selection. An appropriate selection of Kernel Function is crucial for applications of SVM-based classification. A good choice of Kernel Function assures good learning of SVM. A variety of Kernel Functions has been discussed in the literature. Our research uses four Kernel Functions, shown in Table I.

The SVM classifier with a strong kernel function is an important part of the second study of this research where a good kernel with SVM has been identified and used in the proposed combined classifier. This classifier is used as the committee president of the combined classifier committee.

3.4.4 Combining classifiers. In this procedure, classifiers' committee is formed where a number of classifiers are selected as members, and another classifier is selected as the president of the committee. This multilevel classification committee is termed as "Stacking" (Wolpert, 1992; Sakkis *et al.*, 2001). Each fresh document is classified by the members of the committee, and thereafter president considers the output of committee members and selects the best one. The final decision is made by considering the members' individual decisions together with president's decision. The advantage of this approach is that though members frequently make mistakes (i.e. misclassify), the final decision of the committee is rarely incorrect (i.e. misclassifications are far fewer).

In this research, Boosted Probabilistic classifiers, i.e. Boosted Bayesian and Boosted NB are the members of the committee and a function-based classifier, i.e. SVM with Normalized polynomial Kernel (NP), is the president of the committee.

The members and the president classifiers are chosen based on the results of the first and second study, respectively. In the first study, effect of various boosting techniques has been tested for performance boosting of probabilistic classifiers where Bayesian and NB classifiers with AdaBoost were the best and were therefore selected as a members of the committee. In the second study, four different kernels have been tested for better learning of the SVM classifier, and NP kernel with SVM was found to be the best. Therefore, SVM with NP Kernel was selected as president for last part of the study.

Table I.
Kernel functions

Kernels	Full name	Functions
NP	Normalized polynomial kernel	$K^r(x_i, y_j) = \frac{(x_i^T . y_j + 1)^P}{sqrt(x_i^{(T+1)} + y_j^{(T+1)})}$
PK	Polynomial kernel	$K^r(x_i, y_j) = (x_i^T . y_j + 1)^P$
PUK	Pearson VII function-based universal kernel	$K^r(x_i, y_j) = \frac{1}{\left[1+\left(\frac{2 * sqrt(\|x_i - y_j\|^2 sqrt(2^{\left(\frac{1}{\omega}\right)}-1))}{\sigma}\right)^2\right]^\omega}$
RBF	Radial basis function kernel	$K^r(x_i, y_j) = \exp(-\gamma\|x_i - y_j\|^2)$

3.5 Description of the study

In this study, we have used JAVA and MATLAB environment on Window 7 operating system for testing the concerned classifiers. The dimensionality reduction step identifies subsets with varying numbers of most informative features, from the three different data sets. Thereafter, data splitting is performed such that 66 per cent data can be used for training, and remaining 34 per cent data can be kept aside for testing of concerned classifiers.

This study uses some well-known performance measures for evaluation and analysis purposes. The most intuitively appealing measure for comparing classifiers is the Classifier's Accuracy (A) and is defined as the percentage of accurately classified emails. This measure has its own disadvantage, which is inability to distinguish between false positives (FP) and false negatives. For accurate measurement, FP rate must be considered. That is because sometimes legitimate (ham) emails, which mostly carry important information, can be misclassified as spam emails. Minimum FP value for the Ham ensures the maximum number of accurately classified Ham emails. In this study, FP rate for ham emails and all emails have been considered. FP rate for all emails indicates the actual capability of the spam classification model and evaluates the capacity of classifier to judge the correct classes. F-value (F) which is defined as the harmonic mean of Precision (P), the fraction of retrieved classified emails that are relevant, and Recall (R), the fraction of accurate classified emails that are retrieved, is also an informative indicator of accurate classification.

In Table II, related formulae of the performance measures have been shown, where $N_{Ham\to c}$ denotes the total number of correctly classified Ham emails, $N_{Ham\to m}$ is the number of misclassified Ham emails, $N_{Spam\to c}$ is the correctly classified Spam emails and $N_{Spam\to m}$ is the total number of misclassified Spam emails.

Measures	Related formulas	
Accuracy	$A = \frac{N_{Ham\to c} + N_{Spam\to c}}{N_{Ham\to c} + N_{Ham\to m} + N_{Spam\to c} + N_{Spam\to m}}$	
Precision	For ham emails	For spam emails
	$HP = \frac{N_{Ham\to c}}{N_{Spam\to m} + N_{Ham\to c}}$	$SP = \frac{N_{Spam\to c}}{N_{Ham\to m} + N_{Spam\to c}}$
Recall	For ham emails	For spam emails
	$HR = \frac{N_{Ham\to c}}{N_{Ham\to m} + N_{Ham\to c}}$	$SR = \frac{N_{Spam\to c}}{N_{Spam\to m} + N_{Spam\to c}}$
F-value	$F = \frac{2 * P * R}{P + R}$	
False positive rate	For ham emails	For all emails
	$HFP = \frac{N_{Ham\to m}}{N_{Ham\to m} + N_{Ham\to c}}$	$FP^{H,S} = \frac{N_{H,S\to m}}{N_{H,S\to m} + N_{H,S\to c}}$

Table II. Instruments for performance measurement

3.6 Experiments and evaluation

The objective of this work is to use different combinations of classifiers for effective classification of spam and ham email. The testing of classifiers was divided into three parts: the first part performed the testing of "Probabilistic Classifiers" (i.e. Bayesian and NB), without/with "Boosting Algorithms" (i.e. Bagging, Boosting with Re-sampling and AdaBoost), thereafter "Function Based Classifiers" (i.e. SVM) was tested with different "Kernel Functions" [i.e. NP Kernel, polynomial kernel (PK), radial basis function kernel (RBF) and Pearson VII function-based universal kernel (PUK)] and in the last part the proposed "Combined Classifier" was constructed by using the best performing classifiers from the previous two tests. It was then tested on most informative features selected by "Genetic Feature Search Algorithm" for the three different datasets mentioned above.

4. Results and analysis

This research focuses on performance improvement of the classifiers where the main task is improving the classification accuracy and minimizing FP rate with the use of least number of most informative features. Three different studies have been conducted for achieving optimum classification accuracy by testing various classifiers on three different data sets (i.e. Enron email, SpamAssassin, LingSpam). Different numbers of most informative features (i.e. 375 features for Enron, 89 features for SpamAssassin and 63 features for LingSpam data set) have been selected for classifying different number of email files (i.e. 12,000 email files for Enron, 4,700 email files for SpamAssassin and 956 email files for LingSpam data set), with 50 per cent Spam rate for each data set. Enron email data set was chosen as the main data set of this research. The other two data sets (i.e. SpamAssassin and LingSpam) have been used for validation of the results obtained for the first data set.

The measures listed in Table II were used to compare and analyze the performance of the individual classifiers and the proposed combined classifier.

4.1 First study: boosting of the probabilistic classifiers

In this study, two Probabilistic classifiers (Bayesian and NB) have been tested on the three data sets with/without the help of boosting algorithms [i.e. Bagging, Boosting (with re-sampling) and AdaBoost]. The motivation behind this study was lack of learning ability of the classifiers at the training stage. Boosting algorithms strengthen the learning ability of classifiers for improving the classification accuracy.

4.1.1 Test on Enron email data set. Table III and Figure 1 demonstrate the results of this study. Without boosting algorithms, Probabilistic classifiers tested on the Enron data set show poor performance. For individual performance, Bayesian classifier gives better accuracy, i.e. 88.8 per cent, compared to NB, i.e. 88.0 per cent. Results also show that boosting algorithms improve the performance of Probabilistic classifiers. With boosting, Bayesian classifier again performs better than NB with accuracy 89.1 to 92.9 per cent (whereas for NB, it is from 88.4 to 91.7 per cent). Among all boosting algorithms, Boosting (with re-sample) gives best performance with accuracy 92.7 per cent for Bayesian and 91.7 per cent for NB. AdaBoost results were very close to the best. In this case, accuracy was 92.4 per cent for Bayesian and 91.2 per cent for NB. Bagging appears as the worst performer among all boosting algorithms with accuracy 89.1 per cent for Bayesian and 88.4 per cent for NB.

Table III.
Results of first study: accuracy and *F* value of probabilistic classifiers (with/without boosting)

	Probabilistic classifiers											
	Enron				SpamAssassin				LingSpam			
	BayesNet (BN)		NaiveBayes (NB)		BayesNet (BN)		NaiveBayes (NB)		BayesNet (BN)		NaiveBayes (NB)	
In (%)	Acc	*F*-value	Acc	*F*-value	Acc	*F*-value	Acc	*F*-value	Acc	*F*-value	Acc	*F*-value
Boosting algorithms												
Without boosting	88.8	88.7	88	88.1	87.3	87.2	85.6	85.5	92.3	92.3	91.4	91.3
Bagging (Bag)	89.2	89.1	88.4	88.4	87.5	87.4	85.6	85.5	92.3	92.3	91.7	91.7
Boosting with Re-sampling (Boost)	92.9	92.9	91.7	91.7	96.7	96.7	96.2	96.2	92.9	92.9	93.8	93.8
AdaBoost (ABoost)	92.4	92.3	91.2	91.2	97.1	97.1	96.9	96.9	94.5	94.5	93.2	93.2

Table IV and Figure 2 demonstrate FP rate of the concerned classifiers. In this research, we calculate the FP rate for Ham emails and thereafter for the total emails. Results confirm that Boosting with AdaBoost leads to the best classification. It gives accurate classification with low FP rate, i.e. 5.4 per cent for Ham and 7.8 per cent for total emails and hence, it can be said that AdaBoost is the best boosting algorithm.

4.1.2 Test on SpamAssassin data set. Results of classifiers tested on SpamAssassin data set strongly support the observations from the Enron Data set. Table III and Figure 3 reveal that, for SpamAssassin data set also, with/without boosting, Bayesian classifier performs better than NB. In this case, accuracy of Bayesian classifier was 87.3 to 97.1 per cent and for NB, it was 85.6 to 96.9 per cent. On the other hand, AdaBoost algorithm shows excellent performance boosting of probabilistic classifiers with accuracy 97.1 per cent for Bayesian classifier and 96.9 per cent for NB. Bagging algorithm has again proved to be the worst booster with performance accuracy 87.5 per cent for Bayesian and 85.6 per cent for NB (Figures 3 and 4).

In SpamAssassin data set, FP rate for Ham emails was less for Bayesian classifier with Boosting with Re-Sampling method, i.e. 2.2 per cent, but the result of AdaBoost was close to the best one, i.e. 2.8 per cent, whereas for the NB, AdaBoost gives low FP rate, i.e. 2.8 per cent. Also, with AdaBoost, FP rate of total emails was lower, i.e. 2.9 per cent for Bayesian and 3.1 per cent for NB.

4.1.3 Test on LingSpam data set. Same classifiers tested on LingSpam data set strongly validate the results of Enron and SpamAssassin data sets. Again Bayesian classifier proves its worth with accuracy 92.3 to 94.5 per cent, whereas for NB, it is 91.4 to 93.2 per cent. AdaBoost method again demonstrates its strength with accuracy 94.5 per cent for Bayesian Classifier and 93.2 per cent for NB. Bagging method continues to be disappointing by its poor boosting, with performance accuracy 92.3 per cent for Bayesian and 91.7 per cent for NB (Figures 5 and 6).

FP rate for LingSpam data set again comes lower, i.e. 2.4 per cent for Ham emails and 5.6 per cent for overall classification, for the Bayesian classifier with AdaBoost. This supports the results from the other two data sets. For ham emails without boosting, FP rate is lower, but considering together with accuracy, Bayesian with AdaBoost appears to be the best classifier.

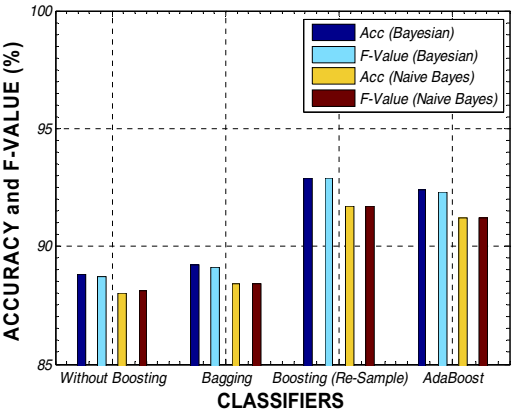

Figure 1.
Accuracy and *F*-value of probabilistic classifier with/without boosting (Enron data set)

Table IV.
Results of first study: False positive rate (Ham and all emails) of probabilistic classifiers (with/without boosting)

	Probabilistic classifiers											
	Enron				SpamAssassin				LingSpam			
	BayesNet (BN)		NaiveBayes (NB)		BayesNet (BN)		NaiveBayes (NB)		BayesNet (BN)		NaiveBayes (NB)	
In (%)	FP (H)	FP (H + S)	FP (H)	FP (H + S)	FP (H)	FP (H + S)	FP (H)	FP (H + S)	FP (H)	FP (H + S)	FP (H)	FP (H + S)
Boosting algorithms												
Without boosting	2.4	11.7	10.2	12	18.6	12.8	21.9	14.6	1.8	7.9	2.4	8.8
Bagging (Bag)	2.5	11.3	9.7	11.7	18.4	12.6	21.9	14.6	1.8	7.9	2.4	8.5
Boosting with Re-sampling (Boost)	10.7	6.9	7.4	8.4	2.2	3.2	3.4	3.7	4.2	7.2	4.8	6.2
AdaBoost (ABoost)	5.4	7.8	8.2	8.8	2.8	2.9	2.8	3.1	2.4	5.6	3.6	6.9

4.2 Second study: kernel selection for support vector machine

This study used SVM as the machine learning classifier for classifying email files of three different data sets. The motivation behind the study was to find a good choice of Kernel function for SVM classifier which had already proven its strength in previous studies reported in literature. Four different Kernel functions, i.e. NP Kernel, PK, PUK and RBF have been tested in this study (Tables V and VI).

4.2.1 Test on Enron email data set. Table V and Figure 7 demonstrate the performance of SVM with different kernel functions. Results show that NP is the best Kernel function among all with accuracy 94.4 per cent, whereas PK, PUK and RBF come in second, third and fourth positions with 93.8, 92.9 and 92.6 per cent accuracy, respectively. However, results of all kernels are very close to each other, but in this study, NP appears to be the best performing Kernel function (Figures 7 and 8).

SVM with NP Kernel proves its worth, not only in accuracy but also in the sensitive classification. In this case, FP rate comes lowest, i.e. 7.1 per cent for Ham emails and 5.5 per cent for all emails, among all.

4.2.2 Test on SpamAssassin data set. Results of SpamAssassin data set validate the observations of the first study. Once again, the performance of SVM with NP kernel function is the best with 98.6 per cent. Although results of other Kernels were more or

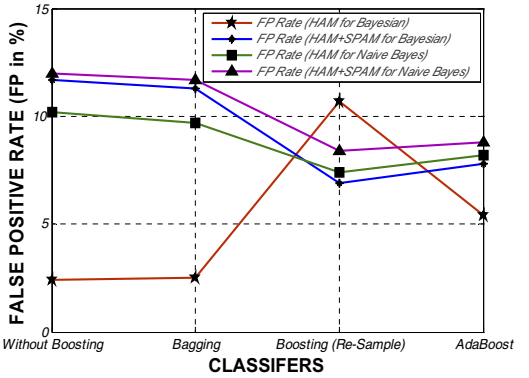

Figure 2.
FP rate (Ham and all emails) of probabilistic classifier with/without boosting (Enron data set)

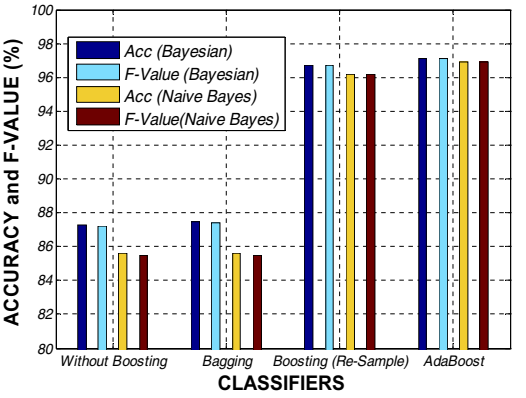

Figure 3.
Accuracy and *F*-value of probabilistic classifier with/without boosting (SpamAssassin)

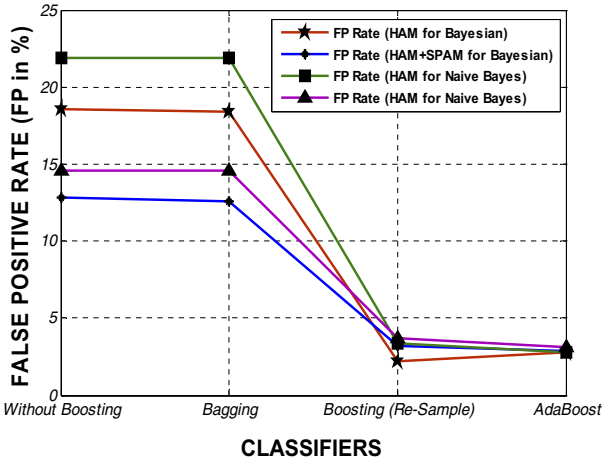

Figure 4.
FP rate (Ham and all emails) of probabilistic classifier with/without boosting (SpamAssassin)

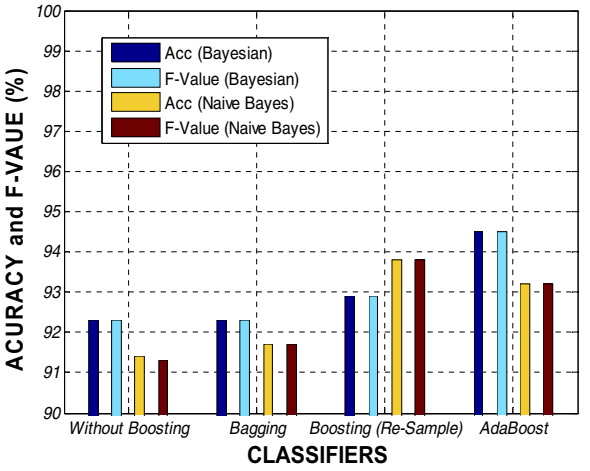

Figure 5.
Accuracy and *F*-value of probabilistic classifier with/without boosting (LingSpam)

less same, i.e. 98.3 per cent for PK, 98.1 for PUK and 98.1 per cent for RBF, again NP was the best one (Figures 9 and 10).

Table VII and Figure 10 validate the observations of the first data set. In this case, again SVM with NP kernel is found best with lowest FP rate, i.e. 2.0 per cent for Ham emails and 1.4 per cent for all emails.

4.2.3 Test on LingSpam data set. Classifiers of this study tested on LingSpam data set again validate the results of the first two data sets. In this case, SVM with Kernel NP is observed as the best classifier with 95.7 per cent accuracy among all the kernels, with PK, PUK and RBF having 95.4, 94.1, 92.9 per cent accuracy, respectively (Figures 11 and 12).

FP rate for classifiers of this study, tested on LingSpam data set, also shows that SVM with NP kernel is the best classifier amongst all with lowest FP rate: 7.3 per cent for Ham emails and 4.2 per cent for all emails.

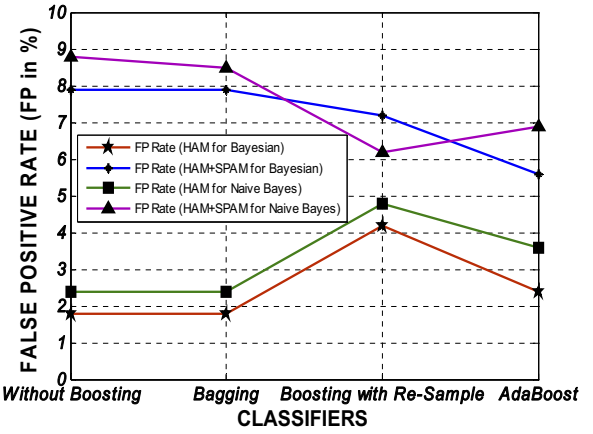

Figure 6.
FP rate (Ham and All emails) of probabilistic classifier with/without boosting (LingSpam)

Table V.
Results of second study: accuracy and *F*-value of SVM with different kernels

	Support vector machine					
	Enron		SpamAssassin		LingSpam	
In %	Acc	*F*-value	Acc	*F*-value	Acc	*F*-value
NP	94.4	94.4	98.6	98.6	95.7	95.7
PK	93.8	93.8	98.3	98.2	95.3	95.4
PUK	92.9	93.0	98.1	98.2	94.1	94.2
RBF	92.6	92.6	98.1	98.1	92.9	92.9

Table VI.
Results of second study: false positive rate (Ham and all emails) of SVM with different kernels

	Support vector machine					
	Enron		SpamAssassin		LingSpam	
In %	FP (H)	FP (H + S)	FP (H)	FP (H + S)	FP (H)	FP (H + S)
NP	7.1	5.5	2.0	1.4	7.3	4.2
PK	8.1	6.0	2.2	1.8	7.9	4.5
PUK	10.3	6.8	3.3	1.8	4.8	5.9
RBF	12.1	7.1	3.5	1.9	12.7	6.9

This study reveals that SVM with NP Kernel is the best compared to the other Kernels. When the compared with the results of the first study SVM with NP kernel turned out to be the overall best. It is for this reason that it is chosen to be the committee president for the third study.

4.3 Third study: combining classifier with committee selection. This is the final study which builds a combined classifier from the best combination of classifiers selected from the first and the second studies, for achieving accurate classification of spam and ham emails. In this case, a committee is formed which includes two different good classifiers selected from the first study (Bayesian with AdaBoost and NB with AdaBoost), that work as committee members, while a classifier is selected from the second study (SVM with NP kernel), that becomes the committee president.

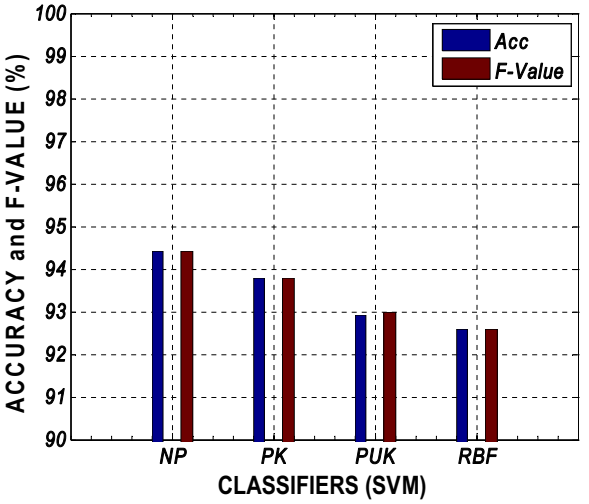

Figure 7.
Accuracy and *F*-value of SVM with different Kernels (Enron data set)

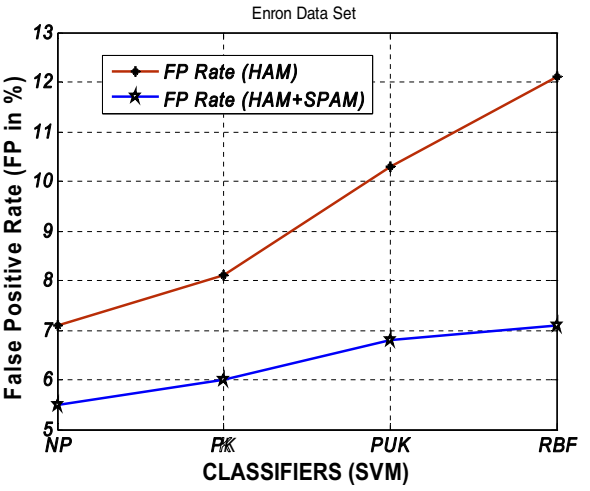

Figure 8.
FP rate (Ham and all emails) of SVM with different Kernels (Enron data set)

4.3.1 Test on Enron email data set. Table VIII and Figure 13 depict the performance of the combined classifier with committee selection tested on Enron email data set. It is observed from the results that combining of the classifiers with committee selection shows excellent performance with 95.6 per cent accuracy and is observed to be the best classifier among all (Figures 13 and 14).

Results of the combined classifier tested on Enron email data set also reveals an outstanding performance in terms of low FP rate, i.e. 3.4 per cent for Ham emails and 4.7 for all emails.

4.3.2 Test on SpamAssassin data set. Results of the same study repeated on SpamAssassin data set validates the observations from the first data set. In this case,

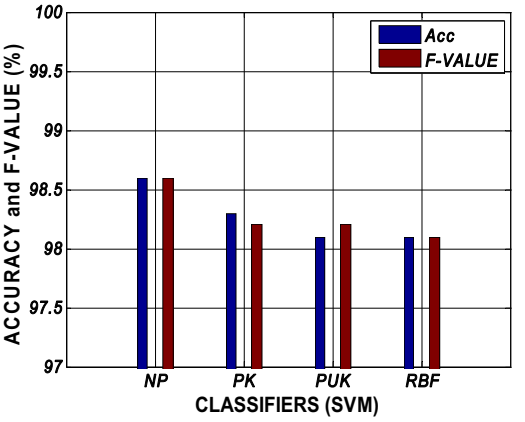

Figure 9.
Accuracy and *F*-value of SVM with different Kernels (SpamAssassin)

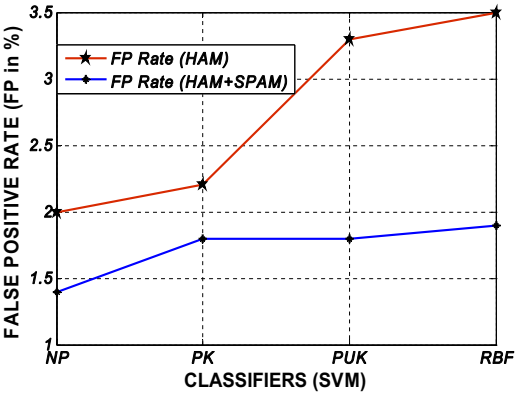

Figure 10.
FP rate (Ham and all emails) of SVM with different Kernels (SpamAssassin)

Table VII.
Results of second study: accuracy and *F*-value of combine classifier with committee selection

	Enron		SpamAssassin		LingSpam	
In (%)	Acc	*F*-value	Acc	*F*-value	Acc	*F*-value
Boost + NB	91.2	91.2	96.9	96.9	93.2	93.2
Boost + B	92.4	92.3	97.1	97.1	94.5	94.5
SVM	94.4	94.4	98.6	98.6	95.7	95.7
Combine	95.6	95.6	98.6	98.6	98.8	98.8

before combining the classifier, SVM itself has given best accuracy, i.e. 98.6 per cent and was observed to be the best performer at that stage; so no further studies were considered necessary (Figures 15 and 16).

The results of the third study performed on SpamAssassin data set validates the results of Enron data set where committee president, i.e. SVM with NP kernel, is found to be the best classifier. In this case, SVM has shown the lowest FP rate (i.e. 2.0 per cent for Ham emails and 1.4 per cent for all emails).

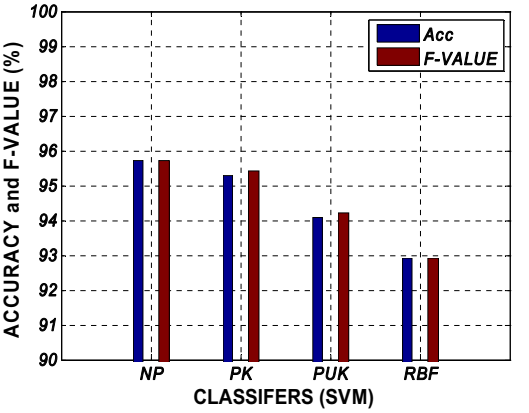

Figure 11.
Accuracy and *F*-value of SVM with different Kernels (LingSpam)

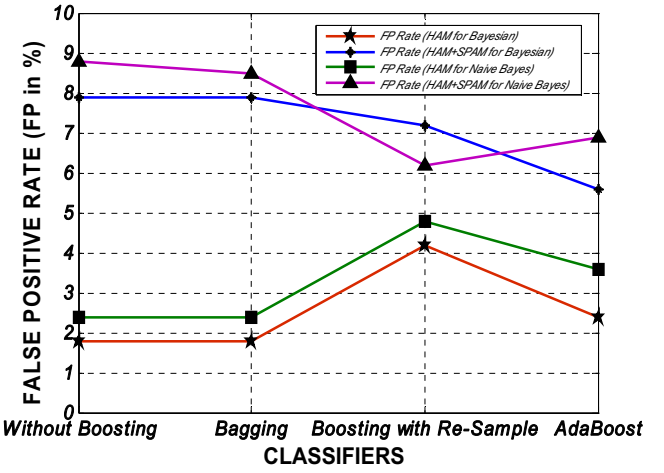

Figure 12.
FP rate (Ham and All emails) of SVM with different Kernels (LingSpam)

Table VIII.
Results of second study: false positive rate (Ham and all emails) of combine classifier with committee selection

	Enron		SpamAssassin		LingSpam	
In (%)	FP (H)	FP (H + S)	FP (H)	FP (H + S)	FP (H)	FP (H + S)
Boost + NB	8.2	8.8	2.8	3.1	3.6	6.9
Boost + B	5.4	7.8	2.8	2.9	2.4	5.6
SVM	7.1	5.5	2	1.4	7.3	4.2
Combine	3	4.7	2	2	1.3	1.2

4.3.3 Test on LingSpam data set. The combined classifier tested on LingSpam validates the results obtained from the other two data sets: the combined classifier proves its strength and gives the best accuracy, i.e. 98.8 per cent (Figures 17 and 18).

Test on LingSpam data confirms the results of Enron and SpamAssassin data sets. In this case too, the combined classifier proves to be the best in terms of low FP rate (1.3 per cent for Ham email and 1.2 per cent for all emails).

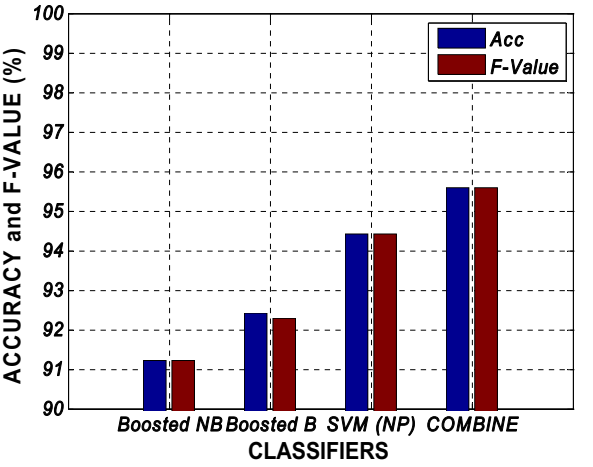

Figure 13.
Accuracy and *F*-value of combine classifier with committee selection (Enron data set)

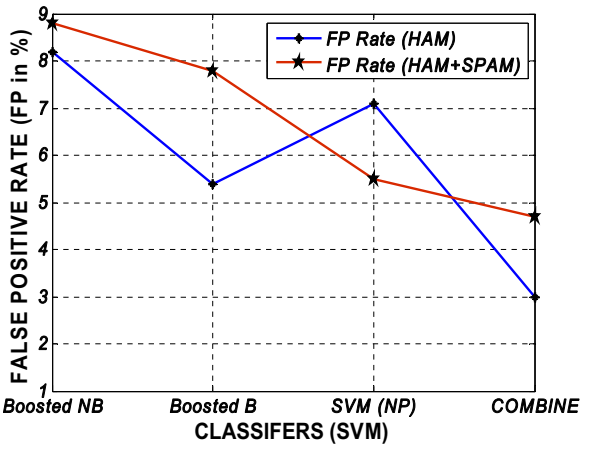

Figure 14.
FP rate (Ham and all emails) of combine classifier with committee selection (Enron data set)

5. Conclusion

After an intense three-part study, this research concludes that stacking of the classifiers identified in this research with committee selection mechanism is a promising approach in the spam classification domain. The aim of this study was to find good classifiers to be stacked for making a combined classifier that had high accuracy as well as low FP rates and could work with a small subset of informative features. The purpose of this research was successfully achieved. The first part of the study identified that probabilistic classifiers performed satisfactorily with boosting algorithms, and that AdaBoost was the best boosting algorithm. The classifiers identified in the first part of the study were selected as the committee members in the final part of the study. The second part of the research focused on finding the best kernel function for SVM classifier for spam email classification. The NP Kernel (NP) was found to be the best, and therefore, SVM with NP Kernel was selected to be the committee president for the final

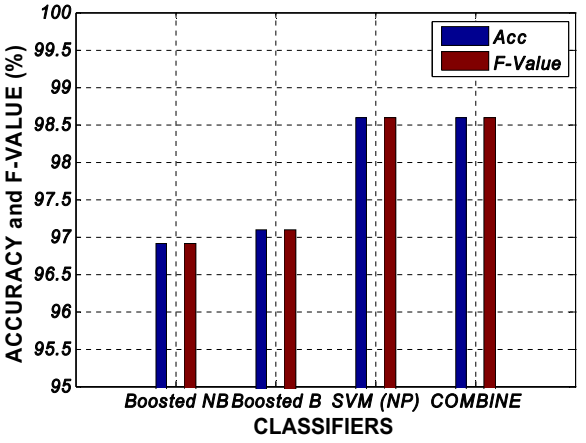

Figure 15.
Accuracy and *F*-value of combine classifier with committee selection (SpamAssassin)

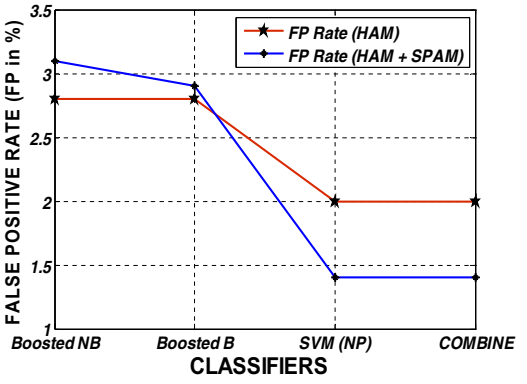

Figure 16.
FP rate (Ham and all emails) of combine classifier with committee selection (SpamAssassin)

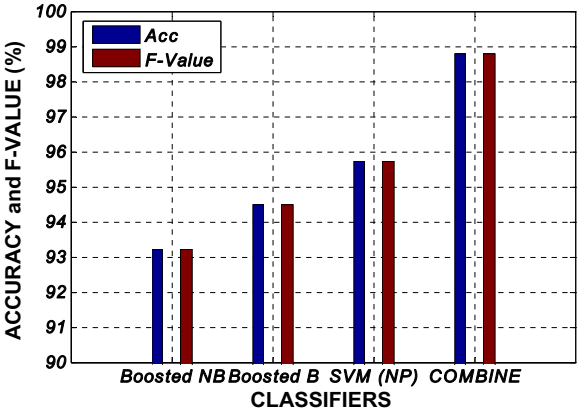

Figure 17.
Accuracy and *F*-value of combine classifier with committee selection (LingSpam)

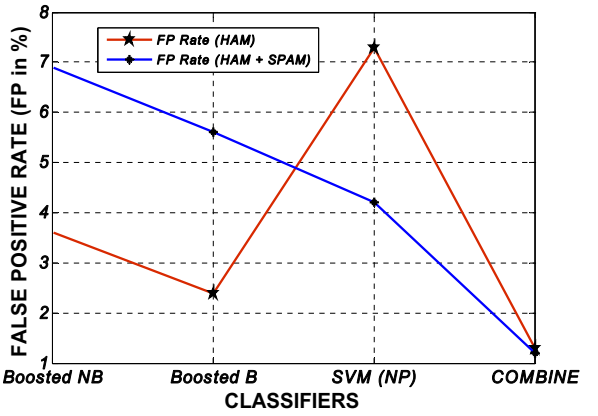

Figure 18.
FP rate (Ham and All emails) of combine classifier with committee selection (LingSpam)

part of the study. By combining the best classifiers, identified in the preceding parts of the research, using a stacking procedure, a committee was formed. The results of our experiments on three different data sets have shown that the combined classifier, thus constructed, has high accuracy as well as low FP rates.

References

Aas, K. and Eikvil, L. (1999), "Text categorisation: a survey", Technical report, Norwegian Computing Centre, Oslo.

Androutsopoulos, I., Koutsias, J., Chandrinos, K.V. and Spyropoulos, C.D. (2000c), "An experimental comparison of naive Bayesian and keyword-based anti-spam filtering with personal e-mail messages", *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and development in Information Retrieval, ACM, New York, NY*, pp. 160-167.

Androutsopoulos, J., Koutsias, K., Chandrinos, V., Paliouras, G. and Spyropoulos, C.D. (2000a), "An evaluation of Naive Bayesian anti-spam filtering", in Potamias, G., Moustakis, V. and van Someren, M. (Eds), *Proceedings of the Workshop on Machine Learning in the New Information Age, 11th European Conference on Machine Learning (ECML 2000), Barcelona*, pp. 9-17.

Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Sakkis, G., Spyropoulos, C.D. and Stamatopoulos, P. (2000b), "Learning to filter spam e-mail: a comparison of a naive bayesian and a memory-based approach", arXiv preprint cs/0009009.

Breiman, L. (1996), "Bagging predictors", *Machine Learning*, Vol. 24 No. 2, pp. 123-140.

Carreras, X. and Màrquez, L. (2001), "Boosting trees for clause splitting", *Proceedings CoNLL-2001 Shared Task, Toulouse*.

Dietterich, G.T. (1997), "Machine learning research: four current directions", *AI Magazine*, Vol. 18 No. 4, pp. 97-136.

Drucker, H., Wu, D. and Vapnik, V.N. (1999), "Support vector machines for spam categorization", *IEEE Transactions on Neural Networks*, Vol. 10 No. 5, pp. 1048-1054.

Duda, R., Hart, P. and Stork, D. (2001), *Pattern Classification*, 2nd ed., John Wiley & Sons, New York, NY.

Efron, B. (1982), "The jack-knife, the bootstrap and other re-sampling plans", *CBMS-NSF Regional Conference Series in Applied Mathematics*, Bowling Green, OH, No. 38, pp. 1-85.

Enron (2004), "Enron email dataset", available at: www.cs.cmu.edu/~enron (accessed June 2008).

Farid, D.M., Zhang, L., Rahman, C.M., Hossain, M.A. and Strachan, R. (2014), "Hybrid decision tree and naive Bayes classifiers for multi-class classification tasks", *Expert Systems with Applications*, Vol. 41 No. 4, pp. 1937-1946.

Freund, Y. and Schapire, R. (1996), "Experiments with a new boosting algorithm", *Proceeding 13th International Conference on Machine Learning (ICML)*, *Bari*, pp. 148-156.

Goodman, J., Cormack, G.V. and Heckerman, D. (2007), "Spam and the ongoing battle for the inbox", *Communications of the ACM*, Vol. 50 No. 2, pp. 24-33.

Haleh, V. and Imam, I.F. (1994), "Feature selection methods: genetic algorithms vs Greedy-like search", *Proceedings of the 3rd International Fuzzy Systems and Intelligent Control Conference*, *Louisville, KY*.

Hastie, T., Tibshirani, R. and Friedman, J. (2001), *The Elements of Statistical Learning*, 2nd ed., Springer, Berlin.

Heydari, A., Ali Tavakoli, M., Salim, N. and Heydari, Z. (2015), "Detection of review spam: a survey", *Expert Systems with Applications*, Vol. 42 No. 7, pp. 3634-3642.

Holland, J.H. (1975), *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, MI.

Jatana, N. and Sharma, K. (2014), "Bayesian spam classification: time efficient radix encoded fragmented database approach", *Computing for Sustainable Global Development (INDIACom), International Conference on*, *IEEE*, pp. 939-942.

Joachims, T. (1998), "Text categorization with support vector machines: learning with many relevant features", *Proceedings 10th European Conference on Machine Learning (ECML-98)*, *Chemnitz*, pp. 137-142.

Kaspersky spam statistics (2014), available at: https://usa.kaspersky.com/internet-security-center/threats/spam-statistics-reports-data#.VaihA_nHh4s

Koller, D. and Sahami, M. (1997a), "Hierarchically classifying documents using very few words", *Proceedings 14th International Conference on Machine Learning (ICML)*, *Nashville, TN*, pp. 170-178.

Koller, D. and Sahami, M. (1997b), "Hierarchically classifying documents using very few words", in Fisher, D.H. (Ed.), *Proceedings 14th International Conference on Machine Learning (ICML)*, *Morgan Kaufmann, San Francisco*, pp. 170-178.

Korada, N.K., Pavan Kumar, N.S. and Deekshitulu, Y.V.N.H. (2012), "Implementation of Naive Bayesian classifier and Ada-Boost Algorithm using Maize expert system", *International Journal of Information Sciences and Techniques (IJIST)*, Vol. 2 No. 3, pp. 63-75.

Larkey, L.S. and Croft, W.B. (1996), "Combining classifiers in text categorization", *Proceedings 19th Annual Conference, Research and Development in Information Retrieval (SIGIR-96)*, *Zurich*, pp. 289-297.

Lewis, D.D. (1998), "Naive (Bayes) at forty: the independence assumption in information retrieval", *Proceedings of 10th European Conference on Machine Learning (ECML-98)*, *Chemnitz*, pp. 4-15.

Lewis, D.D. and Gale, W.A. (1994), "A sequential algorithm for training text classifiers", *Proceedings 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, *Dublin*, pp. 3-12.

Ling-Spam (2000), "Ling-Spam dataset", available at: www.csmining.org/index.php/ling-spam-datasets.html (accessed June 2008).

Oreski, S. and Oreski, G. (2014), "Genetic algorithm-based heuristic for feature selection in credit risk assessment", *Expert Systems with Applications*, Vol. 41 No. 4, pp. 2052-2064.

Sahami, M. (1996), "Learning limited dependence Bayesian classifiers", *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, *Palo Alto*, pp. 335-338.

Sakkis, G., Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Spy-ropoulos, C.D. and Stamatopoulos, P. (2001), "Stacking classifiers for anti-spam filtering of e-mail", *Proceedings of 6th Conference on Empirical Methods in Natural Language Processing*, Vol. 1, pp. 44-50.

Schapire, R. (1997), "Using output codes to boost multiclass learning problems", *Proceedings 14th International Conference on Machine Learning (ICML)*, *Nashville, TN*, pp. 313-321.

SpamAssassin (2005), "SpamAssassin public corpus", available at: http://spamassassin.apache.org/publiccorpus/ (accessed June 2008).

Trivedi, S.K. and Dey, S. (2013a), "Effect of various Kernels and feature selection methods on SVM performance for detecting email spams", *International Journal of Computer Applications*, Vol. 66 No. 21, pp. 18-23, March, Published by Foundation of Computer Science, New York, NY.

Trivedi, S.K. and Dey, S. (2014), "Interaction between feature subset selection techniques and machine learning classifiers for detecting unsolicited emails", *ACM SIGAPP Applied Computing Review*, Vol. 14 No. 1, pp. 53-61.

Vapnik, V. (1995), *The Nature of Statistical Learning Theory*, Springer, AT&T Bell Labs, Holmdel, NJ.

Whittaker, V.B. and Moody, P. (2005), "Introduction to this special issue on revisiting and reinventing e-mail", *Human-Computer Interaction*, Vol. 20 No. 1, pp. 1-9.

Woitaszek, M., Shaaban, M. and Czernikowski, R. (2003), "Identifying junk electronic mail in microsoft outlook with a support vector machine", *Applications and the Internet, Proceedings Symposium on*, *IEEE*, pp. 166-169.

Wolpert, D. (1992), "Stacked generalization", *Neural Networks*, Vol. 5 No. 2, pp. 241-260.

Zhang, L., Zhu, J. and Yao, T. (2004), "An evaluation of statistical spam filtering techniques", *ACM Transactions on Asian Language Information Processing (TALIP)*, Vol. 3 No. 4, pp. 243-269.

Further reading

Trivedi, S.K. and Dey, S. (2013b), "Effect of feature selection methods on machine learning classifiers for detecting email spasm", *Proceedings of the Research in Adaptive and Convergent Systems*, ACM, New York, NY, pp. 35-40.

Trivedi, S.K. and Dey, S. (2013c), "An enhanced genetic programming approach for detecting unsolicited emails", *Computational Science and Engineering (CSE), IEEE 16th International Conference on*, *IEEE*, pp. 1153-1160.

Corresponding author

Shrawan Kumar Trivedi can be contacted at: f10shrawank@iimidr.ac.in